Proceedings of the Artificial Life Conference 2016

Edited by

Carlos Gershenson, Tom Froese, Jesus M. Siqueiros, Wendy Aguilar, Eduardo J. Izquierdo and Hiroki Sayama

Proceedings of the Artificial Life Conference 2016

Edited by:

Edited by Carlos Gershenson, Tom Froese, Jesus M. Siqueiros, Wendy Aguilar, Eduardo J. Izquierdo and Hiroki Sayama

Table of Contents

iv

Preface

Preface

This volume presents the proceedings of ALife 2016, *the Fifteenth International Conference on the Synthesis and Simulation of Living Systems*, held July 4th-8th. It took place in Latin America for the first time in Cancún, México (http://xva.life).

1   The ALife 2016 Program

We received in total 174 submissions, out of which 68 were accepted as oral presentations and 41 were accepted as poster presentations, both of which are included in these proceedings. We accepted 21 late breaking abstracts for poster presentations and are available at the conference website http://xva.life.

The conference program of this year included:

- Eight keynote presentations of internationally renowned speakers within a wide variety of topics:

    - Randall Beer (Indiana University),
    - Mark Bickhard (Lehigh University),
    - Ezequiel Di Paolo (Ikerbasque),
    - Jorge M. Pacheco (Universidade do Minho, Portugal),
    - Alexandra Penn (University of Surrey),
    - Ken Rinaldo (The Ohio State University),
    - Francisco C. Santos (Instituto Superior Técnico, University of Lisbon, Portugal), and
    - Linda Smith (Indiana University, Bloomington).

- Parallel sessions on:

    - ALife and society
    - Origins of life and protocells
    - Self-organization and automation
    - Robotics
    - Genetics
    - Open-ended evolution

- – Morphology
- – Evolvability
- – Cooperation and collective behavior
- – Development
- – Learning and Memory
- – Ecology
- – Artificial societies
- – Language and culture evolution
- – Computational biology
- – Artificial chemistries
- – Living technology
- – Human-Computer interaction
- – Theory and measures

- Eight workshops:

  - – SCBCS: Synthesizing Concepts from Biology and Computer Science
    (Organizers: Emily Dolson and Charles Ofria).

  - – SLCS: Steering Living and Life-like Complex Systems
    (Organizers: Alexandra Penn, Rob Mills, and Emma Hart).

  - – SLACE: Social Learning and Cultural Evolution
    (Organizers: Chris Marriot, Peter Andras, James Borg, and Paulo Smaldino).

  - – BFE: The Biological Foundations of Enactivism (Organizers: Eran Agmon,
    Nathaniel Virgo, Tom Froese, and Matthew Egbert).

  - – OEE: Open Ended Evolution: Recent Progress
    (Organizers: Mark Bedau, Alastair Channon, and Tim Taylor).

  - – GSO: Eight International Workshop on Guided Self-Organization
    (Organizers: Mikhail Prokopenko, Carlos Gershenson, and Daniel Polani).

  - – MEW: Morphogenetic Engineering Workshop
    (Organizers: René Doursat and Hiroki Sayama).

  - – EGT: Multidisciplinary applications of evolutionary game theory
    (Organizers: Tom Lenaerts, Luis A. Martínez-Vaquero, Jelena Grujic, Francisco C. Santos, and Tom Froese.

- Five tutorials:

  - – **MABE**: An Introduction to MABE (Modular Agent Based Evolution) and
    Markov Network Brains.

  - – **AVIDA-ED**: Avida-ED, a tool for teaching a classroom research.

  - – **AEVOL**: In silico experimental evolution with the Aevol Software.

- **ISA**: Introductory Statistics for ALife Experiments: A Visual Approach.

- **NetLogo**: A low threshold/high ceiling for programming multi-agent models.

- The third ISAL Summer School.

- Art exhibition
(Ken Rinaldo (The Ohio State University), Tatsuo Unemi (Soka University), Daniel Bisig (University of Zurich), Mario García-Valdez (Instituto Tecnolgico de Tijuana), Eduardo Makoszay Mayén, and Antonio Isaac Gómez).

2 About the Editors

Carlos Gershenson is a tenured research professor at the Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas of the Universidad Nacional Autónoma de México, where he leads the Self-organizing Systems Lab. His research interests include complex systems, self-organization, urbanism, philosophy, and artificial life.

Tom Froese is Associate Professor at the Research Institute for Applied Mathematics and Systems of the National Autonomous University of Mexico, Mexico City. He is also affiliated with the Center for the Sciences of Complexity at the same university. His research interests include evolutionary robotics, origins of life, and cognitive science.

J. Mario Siqueiros is Associate Professor at the Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas of the Universidad Nacional Autónoma de México. His research interests include complex systems, computational social science, emergence and evolution of culture, social-ecological systems, and philosophy of biology.

Wendy Aguilar works as a Research Assistant at the Department of Computer Science, Intituto de Investigaciones en Matemáticas Aplicadas y en Sistemas at the Universidad Nacional Autónoma de México. She received the PhD., M.Sc., and B.Sc. degrees in Computer Science from the Universidad Nacional Autónoma de México (UNAM). Her research areas include developmental artificial intelligence and computational creativity.

Eduardo Izquierdo is an Assistant Professor in the Cognitive Science Program at Indiana University. He is also affiliated to the Program in Neuroscience, School of Informatics and Computing, the Indiana University Network Science Institute, the Center for Complex Networks and Systems Research, and the Center for the Integrative Study of Animal Behavior. His research is focused on the understanding of behavior from brain-body-environment interactions in simple model organisms.

Hiroki Sayama is Director of the Center for Collective Dynamics of Complex Systems and Associate Professor of Systems Science and Industrial Engineering, at Binghamton University, State University of New York, USA. His research areas include complex systems, dynamical networks, artificial chemistry, computational social science, and interactive evolutionary computation.

3 About the Conference Logo

In recognition of the Maya, this year's *ALife* conference logo includes a Mayan hieroglyph for soul-heart (Figure 1), which is closely related with life. It is read *o'hlis*, and refers to one of the essential animating souls or forces that is specific to the human species with its emotions, knowledge, and thought (Erik Velásquez García, pers. comm.). This force is also an embodiment of the spirit of the god that was principally responsible for the creation of humanity, the god of maize, since the Maya believed that our flesh was originally made from ground corn. The hieroglyph is surrounded by motifs related to the artificial forces of the contemporary world: mechanical and digital technologies. The three bars below *ALife* are the number fifteen in Mayan numerals (one bar represents the number 5), indicating the fifteenth edition of the conference.

Figure 1: The conference logo of ALife XV.

4 Acknowledgments

Arts committee:

Aranda Brito Leonardo
Froese Tom
Mendoza Villafuerte Iliana
Ruíz Rangel Cristina

Program committee:

Aaron Eric	Goñi Moreno Ángel	Pascalie Jonathan
Ackley Dave	Goodman Erik	Payne Joshua
Adamatzky Andy	Goodnight Charles	Penn Alexandra
Agmon Eran	Grabowski Laura	Peronard Jean-Paul
Albrecht Andreas	Grana Manuel	Pettre Julien
Altenberg Lee	Gras Robin	Pfaffmann Jeffrey
Amos Martyn	Gravato Hugo	Philippides Andrew
Andersson Claes	Haasdijk Evert	Pilat Marcin
Arita Takaya	Hamann Heiko	Polack Fiona
Aston Elizabeth	Haruna Taichi	Polani Daniel
Auerbach Joshua E	Harvey Inman	Possani Edgar
Banzhaf Wolfgang	Hashimoto Takashi	Powers Simon
Bartlett Stuart	Hassas Salima	Prokopenko Mikhail
Bedau Mark	Hauser Helmut	Ray Thomas
Bedia Manuel	Hernandez-Lemus Enrique	Reggia James
Beer Randall	Hernndez-Quiroz Francisco	Reynolds Craig
Belavkin Roman	Herrera Pérez Carlos	Risi Sebastian
Benitez Mariana	Herrmann J. Michael	Rojas Raul
Bentley Katie	Hickinbotham Simon	Roli Andrea
Bentley Peter	Hinze Thomas	Ruiz-Mirazo Kepa
Berthouze Luc	Hogeweg Paulien	Rupp Matthew
Beslon Guillaume	Hubert Julien	Salge Christoph
Bhalla Navneet	Huepe Cristian	Sanders David
Bich Leonardo	Humphreys Paul	Santos Francisco C.
Biehl Martin	Hutton Tim	Santos Jose
Bilotta Eleonora	Ichinose Genki	Santos Bruno A.
Blackwell Tim	Iizuka Hiroyuki	Sasahara Kazutoshi
Blum Christian	Ikegami Takashi	Sayama Hiroki
Boldt Joachim	Izquierdo Eduardo J.	Schillaci Guido
Bongard Josh	Jacob Christian	Schmickl Thomas
Borg James	Jaffe Klaus	Shirakawa Tomohiro
Boumaza Amine	Joachimczak Michal	Shirt-Ediss Ben
Brede Markus	Johnson Colin	Silverman Eric
Bredeche Nicolas	Juarez Martínez Genaro	Siqueiros J. Mario
Breen David	Kampis George	Spicher Antoine
Buckley Christopher	Kayama Yoshihiko	Spranger Michael
Bull Larry	Knibbe Carole	Sprevak Mark
Bullock Seth	Kuruma Yutetsu	Standish Russell
Camacho David	LambiotteRenaud	Stanley Kenneth
Cangelosi Angelo	Lehman Joel	Stano Pasquale
Carletti Timoteo	Lenaerts Tom	Stanton Adam

Carmichael Ted
Cecilia Jos M
Cenek Martin
Channon Alastair
Cho Sung-Bae
Connelly Brian D.
Correia Lus
Costa Ernesto
Cronin Leroy
Cussat-Blanc Sylvain
Dandekar Thomas
Dautenhahn Kerstin
Der Ralf
Deunnue Adolfo
Diaconescu Ada
Dodig Crnkovic Gordana
Doncieux Stphane
Dorin Alan
Doursat Rene
Dulman Stefan
Duro Richard
Egbert Matthew
Eldridge Alice
Elmenreich Wilfried
Espinal-Enriquez Jesus
Etxeberria Arantza
Fernandez Jose
Fernndez Nelson
Fogel Gary
Froese Tom
Fchslin Rudolf M.
Fuentes Miguel
Garnier Simon
Geard Nicholas
Gershenson Carlos
Ghazi-Zahedi Keyan
Giacobini Mario
Giavitto Jean-Louis
Glette Kyrre
Goldsby Heather
Gomila Antoni

Lim Soo Ling
Lints Taivo
Lizier Joseph
Lobo Daniel
Loettgers Andrea
Lones Michael
Long John
Lowe Robert
Luga Herve
Marriott Chris
Maye Alexander
Mcgregor Simon
Mckinley Philip
McMullin Barry
Mcowan Peter
J.J. Merelo
Merkle Daniel
Mills Rob
Misevic Dusan
Montagna Sara
Montanier Jean-Marc
Montebelli Alberto
Mouret Jean-Baptiste
Moya Andres
Müller Vincent C.
Murcio Roberto
Máté Nagy
Nichele Stefano
Nitschke Geoff
Nolfi Stefano
Nomura Shin-Ichiro
Nowe Ann
Nuño De La Rosa Laura
Nurzaman Surya Girinatha
OKeefe Simon
Ofria Charles
Ogai Yuta
Oka Mizuki
Olbrich Eckehard
Olson Randal
Pacheco Jorge M.

Stepney Susan
Stoy Kasper
Sullins John
Suzuki Reiji
Suzuki Yasuhiro
Suzuki Keisuke
Tanev Ivan
Tangen Uwe
Taylor Charles
Taylor Tim
Teo Jason
Teuscher Christof
Thill Serge
Torresen Jim
Trianni Vito
Tsuda Soichiro
Tuci Elio
Umerez Jon
Unemi Tatsuo
Vallejo Edgar
Vardy Andrew
Vaughan Neil
Villalobos Mario
Villani Marco
Virgo Nathaniel
Von Mammen Sebastian
Walker Sara Imari
Werfel Justin
Wheeler Michael
Whiting James
Williams Lance
Williams Hywel
Witkowski Olaf
Woodward Alexander
Wrobel Borys
Wuensche Andrew
Yang Xin-She
Yang Xin-She
Zaman Luis
Zenil Hector
Zhang Xiaoge

Finally, the unconditional support from our friends and families was essential during the whole organization process. All the support from all actors made the conference and these proceedings possible.

Part I

Introduction

Introduction

Tom Froese, J. Mario Siqueiros, Wendy Aguilar,
Eduardo J. Izquierdo, Hiroki Sayama, and Carlos Gershenson

Cancún is located in what is known as the Mayan Riviera. In the popular imagination of our contemporary world, the Maya are best known as a lost civilization whose ruins are scattered throughout the tropical jungles of Mexico and Central America. Yet, as Coe [3] (p. 11) reminds us, the Maya are hardly a vanished people: their population numbers over 7 million people, which makes them the largest single block of indigenous people north of Peru. Already at the time of the Spanish conquest, the Maya were found in an area that included all of the Yucatan Peninsula (including, of course, the area of Cancún) and parts of the states of Tabasco and Chiapas in Mexico, as well as Guatemala, Belize, and the western portion of Honduras and El Salvador.

Ideas related to the creation of life have a long history in the Old World, reaching back to the times of classical antiquity, and are expressed in a diverse range of intersecting fields jointly known as artificial life today [1]. It has been argued that the modern sciences of the artificial could learn something from these classical myths [6]. What is less well known is that ancient Mesoamerican cultures also possessed a rich corpus of myths related to what we would now refer to as artificial life. In other words, although this is the first time that an *ALife* conference takes place in Mesoamerica, in a sense the central concept of the field, i.e. the creation of life via artificial means, has already been around the region for centuries if not for over a millennium. In the following we will focus on the specific case of the Maya, from whom several myths related to this concept have survived.

According the Mayan creation account, as recorded by the Quiché Maya during colonial times in the 16th century book Popul Vuh [10] [1], all was empty in the beginning, only murmurs and ripples in an endless sea under a dark night sky. The creator gods convened to bring about the dawn of the world with all of its geological and biological features. But this world was unable to appreciate the grandeur of their efforts and the animals lacked language to express themselves properly. So the gods decided to create humans. A first human design made from mud failed mainly because of its unreliable material composition. The mud man was dissolving and only talked senselessly [2]. So the gods dismantled it.

The gods therefore tried another human design, this time based on wood. They created "manikins, woodcarvings, human in looks and human in speech. [···]. They

---

[1] Page numbers of all the quotations of the Mayan myths refer to this book.

[2] As noted by Tedlock (1996), this reference to a single man made from mud might be an allusion to the Biblical myth of Gods creation of Adam from the dust of the ground, turning this part of the story into an indirect resistance to the colonial doctrine. For the writers of the Popol Vuh, "a singular creature of mud could neither have made sense, nor walked nor multiplied" (p. 231).

came into being, they multiplied, they had daughters, they had sons, these manikins, woodcarvings" (p. 70). Nevertheless, although this design was already better, these replicating manikins had another unexpected defect, because "there was nothing in their hearts and nothing in their minds, no memory of their mason and builder" (p. 70). Although these wooden people could multiply, talk, and move around, their bodies were still too dry and deformed and they lacked an appreciation of their own existence, and "and so they fell, just an experiment and just a cutout for humankind" (ibid.).

In revenge the animal species they had been eating and the tools they had been using began to speak themselves and turned against their former masters: "Their faces were crushed by things of wood and stone. Everything spoke: their water jars, their tortilla griddles, their plates, their cooking pots, their dogs, their grinding stones, each and every thing crushed their faces" (p. 72). What is most interesting from the perspective of artificial life is that their household utensils, and even the houses, became enlivened and took issue with the people's fall from grace. Here, for example, is the lament of their grinding stones:

> "We were undone because of you. Every day, every day, in the dark, in the dawn, forever, r-r-rip, r-r-rip, r-r-rub, r-r-rub, right in our faces, because of you. This is the service we gave you at first, when you were still people, but today you will learn of our power. We shall pound and we shall grind your flesh," their grinding stones told them. (p. 72)

As their utensils rose up against them and their houses collapsed, these wooden humans with their crushed faces scattered in all directions and took refuge in the forests. Today we can still see this previous version of the human form: they are what we call monkeys. Interestingly, this shows that the Maya had no problem conceiving of non-human primates as precursors to humans, an idea which only took hold in the Western imagination following the development of evolutionary theory, in particular after the publication of Darwin's controversial book *The Descent of Man* in 1871.

According to Tedlock, the moral of this story is that we disenchant nature and over-rely on technology at our peril. This ethical concern becomes even more pressing when we consider the contemporary push toward using the tools and insights of artificial life for the creation of so-called living technology [2]. For if we were indeed able to create the conditions for genuinely autonomous, automatically self-reproducing, and open-endedly evolving examples of technology, then how could we guarantee that this living technology will remain favorably inclined towards humans rather than turn out to be just self-interested or even confrontational [4] (pp. 549-551). After all, autonomy is the logical opposite of controllability. Such a vision is still far from being realized in practice. Nevertheless, this failed creation story is a timely reminder that we must balance our efforts to improve technology with a healthy dose of humility and caution. We do not want to end up creating tools akin to the Mayan's disgruntled grinding stones that resent their users.

At this point the narrators of the *Popul Vuh* temporally leave the problem of the origins of modern humans aside in order to relate some myths about the origins of

various personified celestial movements and constellations. One of the myths is of special interest for us because it involves the construction of an artificial life form, which is explicitly conceptualized as being an artificial replica of organic life. In this myth the Four Hundred Boys, patron deities of alcoholic intoxication, scheme to kill a self-aggrandizing crocodilian earth monster, Zipacna, but end up being killed by him instead (and thus serving as a representation of the setting of the Pleiades). The Hero Twins, Hunahpu and Xbalanque, want to revenge their death and so they set a trap for Zipacna:

> It's mere fish and crabs that Zipacna looks for in the waters, but he's eating every day, going around looking for his food by day and lifting up mountains by night. Next comes the counterfeiting of a great crab by Hunahpu and Xbalanque. And they used bromeliad flowers, picked from the bromeliads of the forest. These became the forearms of the crab, and where they opened were the claws. They used a flagstone for the back of the crab, which clattered. After that they put the shell beneath an overhang, at the foot of a great mountain. (p. 84)

Then the Hero Twins talk with the hungry Zipacna and let him know that they have seen a crab that he could eat. They guide him to the bottom of the canyon:

> The crab is on her side, her shell is gleaming red there. In under the canyon wall is their contrivance. "Very good!" Zipacna is happy now. He wishes she were already in his mouth, so she could really cure his hunger. He wanted to eat her, he just wanted it face down, he wanted to enter, but since the crab got on top of him with her back down, he came back out. "You didn't reach her?" he was asked. "No indeed she was just getting on top with her back down. I just barely missed her on the first try, so perhaps I'd better enter on my back," he replied. After that he entered again, on his back. He entered all the way only his kneecaps were showing now! He gave a last sigh and was calm. The great mountain rested on his chest. He couldn't turn over now, and so Zipacna turned to stone. (p. 85) [3]

In this way Zipacna was "defeated by genius alone" (p. 85), that is, by an artificial crab that could make sounds and move around. In other words, although the imagined technology, consisting of flowers and stones, is clearly unrealistic, here we find the very idea of artificial life as such (see illustration in Figure 1). This myth is special because, in contrast, the rebelling utensils were animated by a divine wrath rather than by artifice, and the mud man and the wooden manikin were supposed to be actual living beings instead of imitations of living beings. However, the idea of the counterfeit crab, in an abstract sense, is no different to how we understand the concept of artificial life from a modern perspective: it was an artificial imitation of life rather than natural life itself.

---

[3] According to Tedlock (1996), Zipacna's struggle to wrestle his body into the right position to consummate his hunger becomes a symbolic parody of sexual intercourse (p. 35). This is confirmed by one of his Maya informants, who notes with amusement the gender reversed roles in that finally the man ends up on his back. This interpretation is also supported by the fact that in Mópan Maya the term for crab, yux, is used as a metaphor for vulva.

Figure 1: The Hero Twins make an imitation crab. Figure taken from [8] (p. 214); downloaded from the public domain.

Unfortunately, it is difficult to date the origins of this mythical crab to a specific time period of the ancient Maya, but references to the adventures of the Hero Twins are known from Late Formative times (around 300 BC) onwards [7](p. 134). For example, it has been argued that proto-Classic period stele from Izapa, a Mayan site in Chiapas, Mexico, depict early representations of the Hero Twins' defeat of Zipacna's father, the monster bird Vucub Caquix (ibid., p. 182). Future research could try to determine more precisely when the myth of the artificial crab first arose, and how it compares with the more familiar myths from European antiquity, especially from ancient Greece (Mayor, 2016).

After recounting a number of additional myths centered on the exploits of the Hero Twins and other gods, the narrators of the *Popol Vuh* are ready to return to the story of

the origins of human beings. This time the gods are prepared:

> And here is the beginning of the conception of humans, and of the search for the ingredients of the human body. So they spoke, the Bearer, Begetter, the Makers, Modelers named Sovereign Plumed Serpent: "The dawn has approached, preparations have been made, and morning has come for the provider, nurturer, born in the light, begotten in the light. Morning has come for humankind, for the people of the face of the earth," they said. It all came together as they went on thinking in the darkness, in the night, as they searched and they sifted, they thought and they wondered. And here their thoughts came out in clear light. They sought and discovered what was needed for human flesh (p. 145).

This secret ingredient of the living body turns out to be nothing other than drink and food, especially corn. What follows is a Mayan version of the saying that you are what you eat:

> And this was when they found the staple foods. And these were the ingredients for the flesh of the human work, the human design, and the water was for the blood. It became human blood, and corn was also used by the Bearer, Begetter. [· · ·] And then the yellow corn and white corn were ground, and Xmucane did the grinding nine times. Food was used, along with the water she rinsed her hands with, for the creation of grease; it became human fat when it was worked by the Bearer, Begetter, Sovereign Plumed Serpent, as they are called. After that, they put it into words: the making, the modeling of our first mother-father [4], with yellow corn, white corn alone for the flesh, food alone for the human legs and arms, for our first fathers, the four human works. It was staples alone that made up their flesh (p. 146).

Thus, in the end the creator gods had realized that constructing human beings was best done by using existing organic compounds that lend themselves to be formed appropriately and which can at the same time sustain the resulting living beings. Here we have, in a nutshell, a mythological explanation of the fact that as humans we must eat and drink. It is what sustains our existence.

Interestingly, although the Maya seem to have distinguished between creating substance (the making) and creating form (the modeling), we can see throughout these repeated trial and errors of creation a concern for their interdependence. The human form cannot be artificially imposed on just any kind of substrate; on the contrary, its embodiment requires very specific material conditions. After having found these conditions the gods celebrated their ultimate success, all the while emphasizing that the origins of the human species are an achievement of divine engineering: the first human beings were not born but made.

---

[4]The term "mother-father" is not intended to imply androgynous or dual sex. In fact, the first four persons that were created were all male. Mother-father is an expression that is used to refer to their role as lineage heads.

> They were simply made and modeled, it is said; they had no mother and
> no father. We have named the men by themselves. No woman gave birth
> to them, nor were they begotten by the builder, sculptor, Bearer, Begetter.
> By sacrifice alone, by genius alone they were made, they were modeled
> by the Maker, Modeler, Bearer, Begetter, Sovereign Plumed Serpent. And
> when they came to fruition, they came out human (p. 146).

The genius of the creator gods, i.e. to sacrifice existing life and to process and refashion its matter and form so as to give rise to new life, is reminiscent of a dominant approach in synthetic biology, which also famously boasted to have created a new form of life, in this case by assembling a new cell by reusing existing cells in combination with synthesized components [5]. The challenge now facing synthetic biology is to go a step further and, like these Mayan gods, create new life without basing it on existing individuals, but only on the processing of more basic organic compounds (i.e. to use nothing but "food alone"). This feat will require genius indeed.

There is a final caveat to this success story: it turns out that this time the creator gods have overshot their target and accidentally created super-humans. These first humans saw and knew all there was to know about the world and did so without any movement or effort. The gods were worried because these flawless individuals looked poised to become godlike themselves, thereby defeating the whole point of their creation. Accordingly, they decided to remove some of their knowledge and to diminish their sight, so that the humans would continue to worship while also being concerned with more practical earthly matters.

> "Aren't they merely 'works' and 'designs' in their very names? Yet they'll
> become as great as gods, unless they procreate, proliferate at the sowing,
> the dawning, unless they increase. Let it be this way: now we'll take them
> apart just a little, that's what we need. [···]" And such was the loss of the
> means of understanding, along with the means of knowing everything, by
> the four humans. The root was implanted (p. 148).

There are ways to overcome these inbuilt restrictions, as the humans eventually discover. And the Popol Vuh itself, as a sacred book of council, turns out to be one of them. We can make sense of this by considering that the same technology that enables literacy at the same time enables accumulation of knowledge, and it therefore allows humans to progressively better understand and to see more clearly. It is still no different in today's world: now, in addition to all kinds of information technology, we also have at our disposal a range of technologies that enable us to overcome the limitations of our senses. Yet, as Tedlock [10](p. 60) reminds us, we should be careful to leave space for mystery in our lives. The last humans that stopped wondering about the meaning of existence and became overly absorbed in exploiting nature and technology lost the essence of their humanity. They are still with us today: they are swinging through the trees.

On the other hand, the Maya creation myths do not view that which is artificial as necessarily dehumanizing; to the contrary, the creation and use of technology is shown to be part of our essential nature. In agreement with ancient and modern traditions in the philosophy of technology [9], the narrators of the *Popol Vuh* highlight that the

artificial is constitutive of our very being. After all, the first humans were made, not born. What is not clear, and what the narrators leave for their audience to reflect upon, is how we can ensure that people are empowered rather than overpowered by their unavoidable use of knowledge and technology. The difficulty of realizing this ambition is revealed by the end of the Maya civilization, which had largely collapsed even before the arrival of the Spanish conquerors, a catastrophe that is likely to have been precipitated by too much environmental degradation around the major cities. We must do our best to prevent humanity from repeating the same mistakes again, a possibility that looks increasingly likely and which, though now greatly enhanced in scale, would unfortunately be consistent with the Maya's cyclical view of temporality.

With this unresolved challenge in mind, we dedicate this year's installment of the *ALife* conference series to the theme "Artificial life and society".

References

[1] Aguilar, W., Santamaría-Bonfil, G., Froese, T., and Gershenson, C. (2014). The past, present, and future of artificial life. *Frontiers in Robotics and AI*, 1(8).

[2] Bedau, M. A., McCaskill, J. S., Packard, N. H., and Rasmussen, S. (2010). Living technology: Exploiting life's principles in technology. *Artificial Life*, 16, 89-97.

[3] Coe, M. D. (2005). *The Maya*. London, UK: Thames and Hudson.

[4] Froese, T. (2014). Bio-machine hybrid technology: A theoretical assessment and some suggestions for improved future design. *Philosophy and Technology*, 27(4), 539-560.

[5] Gibson, D. G., Glass, J. I., Lartigue, C., Noskov, V. N., Chuang, R. Y., Algire, M. A., . . . Venter, J. C. (2010). Creation of a bacterial cell controlled by a chemically synthesized genome. *Science*, 329(5987), 52-56.

[6] Mayor, A. (2016). Bio-techne: Half-human soldiers, robot servants and eagle drones - the Greeks got there first. Could an AI learn from their stories? *Aeon*. Retrieved from https://aeon.co/essays/replicants-and-robots-what-can-the-ancient-greeks-teach-us

[7] Miller, M., and Taube, K. (1993). *An Illustrated Dictionary of the Gods and Symbols of Ancient Mexico and the Maya*. London, UK: Thames and Hudson.

[8] Spence, L. (1913). *The Myths of Mexico and Peru*. New York: Thomas Y. Crowell Company.

[9] Stiegler, B. (1998). *Technics and Time, 1: The Fault of Epimetheus* (R. Beardsworth and G. Collins, Trans.). Stanford, CA: Stanford University Press.

[10] Tedlock, D. (1996). *Popol Vuh: The Mayan Book of the Dawn of Life*. New York, NY: Touchstone.

Part II

Keynotes

Autopoiesis and Enaction in the Game of Life

Randall D. Beer[1]
[1]Cognitive Science Program, Indiana University
rdbeer@indiana.edu

Over 40 years ago, the Chilean biologists Humberto Maturana and Francisco Varela put forward the notion of autopoiesis as a way to understand living systems and their phenomenology. Varela and others subsequently extended this framework to an enactive approach that places biological autonomy at the foundation of situated and embodied behavior and cognition. In this talk, I will describe an attempt to place these ideas on a firmer foundation by studying them within the context of a toy model universe, John Conway's Game of Life (GoL) cellular automata. The talk has both pedagogical and theoretical goals. Simple concrete models provide an excellent vehicle for introducing some of the core concepts of autopoiesis and enaction and explaining how these concepts fit together into a broader whole. In addition, a careful analysis of such toy models can hone our intuitions about these concepts, probe their strengths and weaknesses, and move the entire enterprise in the direction of a more mathematically rigorous theory. In particular, I will identify the primitive processes that can occur in GoL, show how these can be linked together into mutually-supporting networks, map the responses of such entities to environmental perturbations, and investigate the paths of mutual perturbation that these entities and their environments can undergo. Some of the topics that can be examined in GoL include the structure/organization distinction, organizational/operational closure, self-production, self-individuation, destructive vs. nondestructive perturbations, precariousness, cognitive domain, subjectivity, significance, sense-making, structural coupling, and enaction. I will end with some comments on the limitations of the GoL model and directions for future work.

Gilbert Simondon and the enactive conception of life and mind

Ezequiel Di Paolo[1]
[1] Ikerbasque, Basque Foundation for Science
ezequiel.dipaolo@ehu.es

The work of French philosopher Gilbert Simondon is seeing a vigorous rediscovery. His ideas have a richlargely untappedpotential for science, e.g., in origins of life studies, developmental psychology, embodied cognition, and artificial life. I summarise some key concepts of Simondons philosophy side-by-side with ideas in enactivism, an approach to life and mind based on the works of Francisco Varela, Hans Jonas, and Maurice Merleau-Ponty. I hope to show that there is much overlap between the two approaches, which is good, but also many productive complementarities, and some tensions, which is better.

Simondon encourages enactivism by making its implications more explicit. He advocates the abandonment of hylomorphic metaphysics (the conceptual separability of form and matter) for an ontology of restless and open-ended materiality, relationality, and virtuality. According to him, being and becoming are mutually co-defined. The subject, in her ongoing individuation, sustains inherently meaningful relations with her world. Physical, biological, mental, and social processes of individuation nicely complement the different kinds of precarious autonomy and sense-making elaborated by enactive theory, concepts that in turn are only implicit in Simondons work.

Individuation involves the organization that happens in a milieu capable of abundant potentialities when a process of concrete transduction occurs from more to less metastable states (crystallization is one example). Organisms are processes of individuation prevented from finishing through regulated engagements with the world in search of new sources of potentiality. This coheres with the enactive concept of life as the regulation of the tensions between self-production and self-distinction. Life and mind, for Simondon, entail the neotenic expansion of the early stages of individuation such that its termination is temporarily and progressively delayed. This makes explicit the material conditions of autonomy and introduces new elements for enactivism such as the notion of pre-individual criticality as inherent in the living body.

Simondons recurrent use of the term information may entail some tensions with enactivism, although his notion is subtle and different from the (hylomorphic) information processing metaphor of biological or cognitive functionalism.

I conclude with reflections on the relevance of Simondons philosophy of technology for artificial life, in particular the implication that any life-like artificial system must be materially embodied and embedded in concrete, open-ended relations with the world.

Artificial Life and Society: Philosophies and Tools for Experiencing, Interacting with and Managing Real World Complex Adaptive Systems

Alexandra Penn[1]

[1] Evolution and Resilience of Industrial Ecosystems Project (ERIE),
Centre for the Evaluation of Complexity across the Nexus (CECAN),
Centre for Research in Social Simulation (CRESS),
Dept. of Sociology, University of Surrey, UK
a.penn@surrey.ac.uk

Many of the grand challenges that society faces are concerned with understanding, managing and indeed creating complex living, lifelike or hybrid systems at multiple scales. Conventional approaches are often unsuccessful in dealing with these complex adaptive systems, which require management tools that interact with dynamic, self-organising processes facing perturbation and change, rather than with inert artefacts. By using interactive steering strategies which exploit CASs dynamics and self-organisation, we can attempt to manoeuvre systems to more preferable, stable states and update our interventions as they adapt.

This however, is not sufficient for real world problem solving. In systems with human involvement, key drivers are often social, political or economic. Possible interventions are limited, goals are subjective and participatory or political processes must be integrated. Not only do we need innovative ways to manage our systems which embrace their complexity, we need broadly-accepted narratives of systems as complex and adaptive to help us to shape policy and management. We must be prepared to take action with incomplete knowledge and require tools and methodologies for steering, monitoring and learning, allowing us to adapt as systems respond to intervention. All in complex adaptive systems which we for the most part experience and intuit rather than measure.

New paradigms are urgently required and our community can play a key role. Artificial Life offers tools and philosophical approaches well-matched to the nature of these systems and can provide important perspectives on how to progress. I will give an overview of the potential contribution that I believe that Alife can make and the need to connect productively with many different disciplines. In particular how ALifes technologies and approaches, combined with its inherent creativity, focus on synthetic methods and philosophy with a screwdriver ethos, provide what I believe is the perfect basis for engaging with real world complex systems.

Why development matters to (artificial) life: Lessons from human babies

Linda Smith[1]
[1]Dept. of Psychological and Brain Sciences, Indiana University
smith4@indiana.edu

Why do living forms develop? Development, like evolution and culture, is a process that creates complexity by accumulating change. At any moment, the developing agent is a product of all previous developments, and any new change begins with and must build on those previous developments. Biological systems that are flexibly smart have relatively long periods of immaturity. Why is this? This talk will consider answers to this question using evidence from the first two years of life of human infants. The core ideas are that an adaptive system that can succeed in varied and novel contexts is slow does not settle too fast; develops new mechanisms of change and learning processes over the life time; develops in a series of different environments.

Cognition and the Brain

Mark Bickhard[1]
[1]Department of Philosophy, Lehigh University
mhb0@lehigh.edu

Standard semantic information models are, arguably, conceptually incoherent and factually false about the brain. But, nevertheless, they constitute the primary frameworks for modeling cognitive processes, including in the brain. If such models are ultimately not viable, what sort of framework could model cognition in the brain? I will argue that an action based approach, in the general lineage of pragmatism, provides an alternative modeling framework. In this approach, anticipatory processes are necessary as part of the evolutionary solution to (inter-)action selection, and these yield emergent truth value possibilities of being true or false and thus ground cognition and representation in general. Such an action framework requires timing, thus oscillatory/modulatory processes, and this is in fact what we find as constituting functional processes in the brain. I will outline a micro-scale level of this model and, if time permits, a bit of a macro-scale level. This model has some superficial similarities to predictive brain models, but also fundamental and crucial differences.

In symbio biopoiesis as model of evolved Alife (400 PPM Microbiome)

Ken Rinaldo[1]
[1] Department of Art,
Art & Technology
The Ohio State University
rinaldo.2@osu.edu

Artificial life techniques are illustrative at exploring the wisdom of natural living systems. Genetic algorithms, cellular automatons are computativily complex and visually seductive with Alife in silico. Robotic artists, creating Alife installations experience design challenges more akin to works created in vivo. Robotic works function both in silico and in vivo, with the virtual spaces of computer code and unpredictable environment of the real world. With robotic Alife installations, things like; will the interactant test the system with muscle or try to defeat the code, are design challenges forcing evolution.

Interactive artists have pioneered behavior based works conceived with the current understanding of living systems, such as bottom up emergent behaviors, subsumption architectures (Autopoiesis, Fusiform Polyphony), parallel processing (Paparazzi Bots), distributed intelligences and energy autonomy (400 PPM Microbiome & Autotelematic Spider Bots).

Still, in silco / in vivo works within the Alife are islands of artifice. True living systems offer symbiotic convolutions with overlapping living systems at all scales. This is a function of their organic nature. Mitochondria and symbiogenisis are excellent examples.

In order for artificial life to further evolve and emerge artists and scientists, will need to create larger symbiotically intertwined systems. Systems moving beyond in silco and in vivo, to in symbio. Prototypical living systems will need to find and collect their own energy sources from both living and non living systems. They will find symbiotic intwining through organic interfaces to complex social systems (Augmented Fish Reality) and to bacterial cultures (Enteric Consciousness). Alife research that pioneers natural organic breakdown with bacterial cultures and sustainable practices such as aquaponics offer clear examples (The Farm Fountain).

In Symbio intertwinings of natural and inorganic electro-mechanical elements will be an important and very natural confluence and co-evolution that is necessary between living and co-evolving technological cultures, in the future of Alife.

Linking Individual to Collective Behavior in Complex Adaptive Networks

Jorge M. Pacheco[1]
[1]Departamento de Matemtica e Aplicações,
Universidade do Minho, 4710 - 057 Braga, Portugal
jmpacheco@math.uminho.pt

A long-standing and central problem in Physics is to understand how collective behavior results from a given two- or N- body fundamental interaction. Similarly, in a society, a central problem is to understand the link between individual social behavior and emergent collective phenomena (vaccination, epidemics, crowd behavior, diffusion of innovations, global governance, etc).

Here I address this problem by letting individuals engage in pair-wise interactions by means of a well- defined social dilemma (a prisoner's dilemma of cooperation). These individuals are embedded in a social network that is both complex and adaptive. Adaptation here allows individuals to manifest preferences and resolve conflicts of interest, reshaping the network accordingly. Exact Monte-Carlo simulations reveal the inadequacy of any of the tools developed to date to predict the co-evolutionary dynamics of the population at large. I will present and discuss in detail an adaptive-network-sensitive observable that is capable of predicting the collective, population-wide dynamics, given prior knowledge of the fundamental rules that govern the social interaction between 2 individuals in a social network. In this fundamental step towards linking individual behavior with population wide dynamics, I show that adaptive social networks act to change the "collective" game, from a 2-person game to a N-person game exhibiting a radically different co- evolutionary dynamics, associated with a concomitant fundamental transformation of the nature of the associated Nash equilibria.

Climate Change Governance, Cooperation and Self-organization

Francisco C. Santos[1]
[1]Instituto Superior Técnico, INESC-ID & ATP-group,
Universidade de Lisboa, Portugal
franciscocsantos@tecnico.ulisboa.pt

When attempting to avoid global warming, individuals often face a social dilemma in which, besides securing future benefits, it is also necessary to reduce the chances of future losses. Unfortunately, individuals, regions or nations may opt to be free riders, hoping to benefit from the efforts of others while choosing not to make any effort themselves. Moreover, nations and their leaders seek a collective goal that is shadowed by the uncertainty of its achievement. Such types of uncertainties have repeatedly happened throughout human history from group hunting to voluntary adoption of public health measures and other prospective choices. In this talk, I will discuss a population dynamics approach to a broad class of cooperation problems in which attempting to minimize future losses turns the risk of failure into a central issue in individual decisions. Our results suggest that global coordination for a common good should be attempted by segmenting tasks in many small to medium sized groups in which perception of risk is high. Moreover, whenever the perception of risk is low as it is presently the case we find that a polycentric approach involving multiple institutions is more effective than that associated with a single, global one, indicating that a bottom-up approach, setup at a local scale, provides a better ground on which to attempt a solution for such a complex and global dilemma. Finally, I will discuss the impact on public goods dilemmas of uncertainty in collective goals, heterogeneous political networks, obstinate players and wealth inequality, including a distribution of wealth representative of existing inequalities among nations.

Do Endothelial Cells Dream of Eclectic Shape?

Katie Bentley[1]
[1]Computational Biology Laboratory,
Centre for Vascular Biology Research,
Beth Israel Deaconess Medical Center
kbentley@bidmc.harvard.edu

Endothelial cells (ECs), which line our blood vessels, exhibit dramatic plasticity and diversity of form/behavior at the individual and collective cell level. They reorganize themselves in space and time to extend new blood vessel networks during development and during a huge array of diseases including cancer. Here we will describe, using examples from our integrated *in silico/in vitro/in vivo* research program, how the Artificial Life (ALife) perspective and approaches have been paramount in driving entirely new experimental biology understanding of the vasculature by capitalizing on the emergent, predictive capacity and testable nature of agent-based models in close combination with *in vitro* and *in vivo* experiments.

Our agent-based simulations explicitly consider the role of individual EC embodiment, active perception, heterogeneous vs homogeneous collective dynamics, pattern formation and counter-intuitive emergence from feedback in "controller" networks and many more Alife centric concepts. We recently identified *in silico* that the time it takes ECs to collectively decide who should move and who should stay during blood vessel branching morphogenesis can be varied by altering tissue environment conditions, including some changes found in tumors. By proceeding to validate these predictions *in vitro* and *in vivo* by integrating the studies in the wetlab we have been able to provide a solid new mechanism to explain the diversity of vascular network structures found across tissues and the malformations arising in disease.

There is a bright future with untapped potential for the Alife community to further contribute to understanding of animals, including humans, at the cell and tissue level, where many organizational principles of the systems behavior are still lacking. If we take care to be rigorous in how we calibrate our models to biological data and make clear experimentally testable predictions, we will show we can make real change in a experimental cell biology field, traditionally segregated from in silico research. Learning from the plight of the insightful, but ostracized, Androids in Philip K Dicks novel, overcoming our cultural differences and integrating better between the artificial and natural living systems research communities could lead to huge advantages in achieving our common goals to "understand life as it is".

Part III

Conference Presentations

Special Session on ALife and Society

Artificial Life and Society: Philosophies and Tools for Experiencing, Interacting with and Managing Real World Complex Adaptive Systems

Alexandra Penn[1]

[1] Evolution and Resilience of Industrial Ecosystems Project (ERIE), Centre for the Evaluation of Complexity across the Nexus (CECAN), Centre for Research in Social Simulation (CRESS), Dept. of Sociology, University of Surrey, UK
a.penn@surrey.ac.uk

Abstract

Many of the grand challenges that society faces are concerned with understanding, managing and indeed creating complex living, lifelike or hybrid systems at multiple scales. Conventional approaches have often been unsuccessful in dealing with the inherent non-linearity, adaptability and self-organised behaviours of these systems. In fact the underlying technologies often transform the involved organizations and society as a whole. New paradigms are clearly required and the ALife community can play a key role. Artificial Life offers both tools and philosophical approaches which are well matched to the nature of these systems and can provide important perspectives on how to move forward.

Many of the most important challenges which now face our societies involve the management of interlinked complex adaptive systems (CAS): coupled socio-economic and ecological systems composed of many interacting elements which have been created or partially created by human actions. As we explicitly wish to manage and transform these systems, engineering and design approaches have much to offer us, however they must be fundamentally modified to deal with CAS (see e.g. Penn *et al.*, 2010, Frei and Di Marzo Serugendo, 2012). These systems are not static artifacts, but dynamic, evolving and reflexive *processes* the behaviour of which is not straightforwardly predicable and which may respond in unexpected ways in response to our interventions. Additionally many of the complex systems which we would most like to influence have significant social components. Objective choices about design goals cannot be made and the integration of participatory or political processes may be required. In parallel, new technologies that exploit or emulate the unique properties of living systems from the cellular to the digital realm have great potential, but create new engineering challenges and social dilemmas which must be addressed before they can become broadly utilized.

Conventional approaches to working with both living and life-like complex adaptive systems are, for the most part, "brute force", attempting to effect control in an input- and effort-intensive manner and are often insufficient when dealing with their inherent non-linearity and complexity. Much human management of complex adaptive systems/living systems has involved simplifying their dynamics or functions via large inputs of energy or work. Ecosystems, for example, are commonly constrained and managed for a minimum set of ecosystem functions, e.g. arable agricultural systems, fisheries and flood risk management in wetlands. Doing so may involve forcing systems into states that are far from natural equilibria and hence inherently unstable and requiring significant energy to maintain. System management of this type is ineffective in low energy/resource regimes and may be vulnerable to sudden state change. It offers an illusion of predictability and control, but is vulnerable when external drivers change (Deffuant, and Gilbert, 2011). Evidently, by their very nature, CASs are dynamic, adaptive and resilient and require management tools that interact with dynamic processes rather than inert artefacts. Particularly when as now, systems face increased perturbation and change. By using "steering" strategies which recognize their dynamical nature, we can attempt to manipulate these systems to move between attractors, allowing them to remain stable in a preferable state without significant energy needed to retain it.

However, a dynamical systems perspective is not sufficient for real world problem solving. On the ground in complex adaptive systems with human components or influence, key drivers are often social, political or economic (Gilbert and Bullock 2014). Our range of possible interventions is limited. Not only do we need different ways to manage our systems which embrace their complexity, we need broadly accepted narratives of systems as complex and adaptive to help us to shape policy and management and we must be prepared to take action without full system understanding. We must accept, explicitly recognize and be able to make decisions with incomplete knowledge, and require tools and methodologies for steering, monitoring and gathering knowledge which will allow us to adapt as systems respond to our intervention in complex adaptive systems which we for the most part experience and intuit rather than measure (Rowley et al. 1997, Kay *et al.* 1999, Waltner-Toews and Kay, 2005)

Artificial Life provides unique perspectives, tools and philosophies, which can offer both technical and methodological approaches to understanding and intervening in complex systems subject to "wicked" problems. And with our creation of new living, life-like and intelligent technologies, it plays a part in constructing the complex systems of the future (See e.g.Braha *et al.*, 2006, Bedau *et al.*, 2010). The time is ripe therefore, for the Artificial Life community to collate ideas on our technologies and approaches and their possible societal contributions and impact.

I will give an overview of the potential contribution that I believe that Alife can make and the need to connect productively with many different disciplines. In particular how ALife's focus on interaction, embodiment, dynamics, enactivism and phenomenological approaches, combined with its inherent creativity, focus on synthetic methods and "philosophy with a screwdriver" provide what I believe is the perfect basis for engaging with real world complex systems.

To illustrate these connections I will discuss numerous examples. Including some from my own work in developing participatory complexity science tools for use by policy makers and system stakeholders in both regional industrial economies ("industrial ecosystems") and water catchment management in the UK (Penn *et al.*, 2013, 2014, 2016). In particular, using a "steering complex adaptive systems" approach; a continuous process involving interacting with, monitoring and learning from the system in question which combines tools from complexity science and participatory methods with whole systems design philosophy and adaptive management (Penn, *Forthcoming*). In this particular context I will describe some of the experiences and challenges of combining mathematical modelling and analysis with participatory work in the context of rapidly changing real world systems and how an Alife perspective has informed the work. I will further discuss the potential role of experiential complex systems and ask what an interactive "natural history" approach to complex adaptive systems can offer in combination with mathematical and computational tools.

References

Bedau, M., McCaskill, J., Packard, N., and Rasmussen, S. (2010) Living technology: Exploiting life's principles in technology. Artificial Life, 16:89–97.

Braha, D., Minai, A., and Bar-Yam, Y. (2006) Complex Engineered Systems: Science Meets Technology. Springer.

Deffuant, G. and Gilbert, N. *eds* (2011) Viability and Resilience of Complex Systems: Concepts, Methods and Case studies from Ecology and Society. Springer-Verlag berlin Heidelberg

Frei, R. and Di Marzo Serugendo, G. (2012) The Future of Complexity Engineering *Cent. Eur. J. Eng.* 2(2), 164-188 DOI: 10.2478/s13531-011-0071-0

Kay, J.J., Regier, H.A. Boyle, M. and G. Francis. (1999). An ecosystem approach for sustainability: addressing the challenge of complexity. *Futures* 31 (7):721-742

Gilbert, N. and Bullock, S. (2014). Complexity at the social science interface. Complexity, page doi:10.1002/cplx.21550

Penn A.S., Watson R., Kraaijeveld A., Webb J., Systems Aikido: A novel approach to managing natural systems (2010) in *12th Int. Conf. on Int. Conf. on the Synthesis and Simulation of Living Systems (Artificial Life XII)*, (Odense, Denmark), pp. 577–578

Penn A.S, Knight, CJK, Lloyd, DJB. Avitabile, D., Kok, K., Schiller, F. Woodward, A., Druckman, A. and Basson, L. (2013) Participatory development and analysis of a fuzzy cognitive map of the establishment of a bio-based economy in the Humber region. *PLoS one* 8(11) DOI: 10.1371/journal.pone.0078319

Penn, A.S. P.D. Jensen, A. Woodward, L. Basson, F. Schiller, A. Druckman (2014) Sketching a network portrait of the Humber region *Complexity*

Penn A.S, Knight, C.J.K., Chalkias, Y. ,Velenturf, A. Lloyd, D.J.B. (2016) Extending a Fuzzy Cognitive Map Using Control Nodes: a case study of influencing the development of a bio-based economy in the Humber region. In *Participatory Modeling in Environmental Decision-making: Methods, tools, and applications*. Springer. *Eds* Gray, S., Gray, S. and Jordan, R.

Penn A.S (Forthcoming) Steering a Complex Adaptive System: A complexity science design methodology applied to an industrial ecosystem in the Humber Region, UK In *Social Systems Engineering: The Design of Complexity*, John Wiley & Sons,Series in Computational and Quantitative Social Science Edited by C. Garcia-Diaz, C. Olaya

Rowley, T., G. Gallopin, D. Waltner-Toews, and E. Raez-Luna. 1997. Development and application of an integrated conceptual framework to tropical agroecosystems based on complex systems theories: Centro Internacional de Agricultura Tropical-University of Guelph Project. *Ecosystem Health* 3:154-161.

Waltner-Toews, D., and J. Kay. (2005). The evolution of an ecosystem approach: the diamond schematic and an adaptive methodology for ecosystem sustainability and health. *Ecology and Society* 10(1): 38.

Alife as a Model Discipline for Policy-Relevant Simulation Modelling: Might "Worse" Simulations Fuel a Better Science-Policy Interface?

Seth Bullock

Department of Computer Science, University of Bristol, UK
seth.bullock@bristol.ac.uk

Abstract

Policy-relevant scientific models are typically expected to make *empirically valid predictions* about policy-relevant problems. What are the consequences of shaping our science-policy interface in this way? Here, it is argued that the theoretically insecure simulation modelling pioneered within artificial life is emblematic of an important alternative approach with significance for policy-relevant modelling.

The 21st century brings a raft of significant *systemic* challenges: global finance, global climate, global technology, global security, global governance, global sustainability, etc. Meeting these challenges will involve understanding and managing complex systems that comprise many interacting parts (Gilbert and Bullock, 2014). Computer simulation is emerging as the key scientific tool for dealing with such systems (e.g., Farmer and Foley, 2009). As a consequence, simulation science is increasingly political and has the potential to be critical to our future well-being and quality of life (e.g., Edwards, 2010). To what extent do our current simulation modelling practices measure up to the responsibility that they must now shoulder?

The assumed gold standard for simulation modelling is that our models should achieve a (pseudo)-empirical status (Peck, 2004) that allows their results to be understood as secure forecasts, or valid predictions about their real-world target systems (but see Epstein, 2008). Where simulations have real-world political significance, it is expected that their predictions and forecasts can inform the decisions of policy makers, stakeholders, etc. Such models are analogous to the simulated wind tunnels within which new designs of cars or planes are trialled before deciding whether they should be constructed and sold in the real world (e.g., Bullock, 2011).

Why would we want a science-policy interface structured in this way: one in which the flow from policy to science defines "challenges" and "impact" while the flow in the other direction takes the form of predictions and forecasts? Here, the merits of an alternative science-policy interface will be considered; one in which simulation models that do not rely upon empirical validity are built and explored in order to generate *insights* into our understanding of policy-relevant target systems (Di Paolo et al., 2000).

I argue that, in the same way that *Drosophila melanogaster* and *C. elegans* have played a useful role within biology as model organisms, Artificial Life has the potential to be a *model discipline* within academia, epitomising the use of simulation to deliver *insights into what may be* as opposed to *predictions of what will be*.

There are several points in favour of this position. First, the field is explicitly located at the boundary between the artificial and the real (Silverman and Bullock, 2004); second, the research community has explicitly debated the role of simulation and its epistemological status (e.g., Wheeler et al., 2002); and third, crucially, it is generally accepted that most artificial life models lack theoretical security in the following sense. Theoretically secure models are underwritten by mature and consensually agreed upon theory (e.g., the Navier-Stokes equations for fluid flow). Insecure models do not benefit from such a mature theoretical underpinning, but rather, are exploratory attempts to generate or progress such theory. Perhaps uniquely, the field of artificial life deliberately courts insecurity of this kind, by tackling counterfactual questions ("life-as-it-could-be") for which there is, almost by definition, little theoretical basis. Similarly, the related field of complexity science can be regarded as an effort to address questions concerning emergent phenomena that lie outside the security provided by reductionist science.

Research into the policy-relevant systemic questions with which this paper opened currently tends to be deeply insecure in the sense described above. It may not always be this way since systemic problems are not necessarily inherently insecure. Indeed, it is to be hoped that progress on the problems of climate change, financial stability, etc., might provide more secure foundations for policy making in the near future. However, presently, it is not only more facts and data pertaining to these problems that is required but more and better theory with which to marshall and make sense of these data. Consequently, it is important to learn lessons from modelling and simulation efforts that may appear "worse" in terms of empirical validity, but may be "better" in terms of generating the insights and understandings

that fuel new and better theory.

In outlining this style of simulation, I will draw on an analysis of Charles Babbage's simulation model of miracles, the first example of a simulation model (Bullock, 2000), and its subsequent impact on attempts to build machines capable of automatic economic reasoning (Bullock, 2008), and analysis of Levins' position on the trade-offs inherent in modelling living systems (Levins, 1966; Bullock, 2014).

As a concrete example of this type of model, consider Schelling's (1971) well-known model of urban segregation, which demonstrates that an initially well-mixed population of city dwellers of various types, each with a tendency to relocate at random unless they are adjacent to at least a few similar individuals, tends over time to become strongly spatially segregated by type. The insight here is that real-world segregated populations may not be made up of individuals with explicit preferences for segregation; the "micromotives" of the agents may not be reflected straightforwardly in their aggregate, population-level patterns. Schelling's model is not empirically valid, makes no forecasts, and is not intended to reresent a specific city or population. Nevertheless it delivers an insight that is compelling, transparent and, consequently, capable of helping to shape better social policy.

Conversely, one might imagine many more sophisticated and more data-driven models of, say, Chicago's economy (including representations of its housing market, land prices, geography, etc.) and population (including accurate demographic information, and analogues of relevant psychological and social processes, etc.). Such models might be able to achieve a degree of empirical validity in the sense of being able to "hind-cast" Chicago's historical patterns of segregation to some degree of accuracy. Nevertheless, despite the ready ability of such models to generate specific forecasts and point predictions with the potential to strongly determine associated policy decisions, they would (given our relatively poor understanding of the fundamental systems involved) be both insecure and opaque, and, consequently, problematic as policy tools.

The groundwork outlined above enables us to engage with three key questions: Might a re-examination of simulation modelling enable a science-policy interface to be restructured in order to allow insights to pass across it? To what extent are accountability and democracy impacted by the current character of the science-policy interface? Could epistemically insecure simulation models of the kind pioneered within artificial life research offer a paradigm within which an effective reassessment of impact-driven science takes place? To which the answers argued for are, respectively: "yes", "badly", and "perhaps".

References

Bullock, S. (2000). What can we learn from the first evolutionary simulation model? In Bedau, M. A., McCaskill, J. S., Packard, N., and Rasmussen, S., editors, *Proceedings of the Seventh International Conference on Artificial Life*, pages 477–486. MIT Press, Cambridge, MA.

Bullock, S. (2008). Charles Babbage and the emergence of automated reason. In Husbands, P., Holland, O., and Wheeler, M., editors, *The Mechanical Mind in History*, pages 19–39. MIT Press, Cambridge, MA.

Bullock, S. (2011). Prospects for large-scale financial systems simulation. Technical report, Foresight Programme of the UK's Government Office for Science.

Bullock, S. (2014). Levins and the lure of artificial worlds. *The Monist*, 97(3):301–320.

Di Paolo, E. A., Noble, J., and Bullock, S. (2000). Simulation models as opaque thought experiments. In Bedau, M. A., McCaskill, J. S., Packard, N., and Rasmussen, S., editors, *Proceedings of the Seventh International Conference on Artificial Life*, pages 497–506. MIT Press, Cambridge, MA.

Edwards, P. N. (2010). *A Vast Machine: Computer Models, Climate Data, and the Politics of Global Warming*. MIT Press.

Epstein, J. M. (2008). Why model? *Journal of Artificial Societies and Social Simulation*, 11(4):12.

Farmer, J. D. and Foley, D. (2009). The economy needs agent-based modelling. *Nature*, 460:685–686.

Gilbert, N. and Bullock, S. (2014). Complexity at the social science interface. *Complexity*, 19:1–4.

Levins, R. (1966). The strategy of model building in population biology. *American Scientist*, 54(4):421–431.

Peck, S. L. (2004). Simulation as experiment: A philosophical reassessment for biological modeling. *Trends in Ecology and Evolution*, 19(10):530–534.

Schelling, T. C. (1971). Dynamic models of segregation. *Journal of Mathematical Sociology*, 1:143–186.

Silverman, E. and Bullock, S. (2004). Empiricism in artificial life. In Pollack, J., Bedau, M., Husbands, P., Ikegami, T., and Watson, R. A., editors, *Proceedings of the Ninth International Conference on Artificial Life*, pages 534–539. MIT Press, Cambridge, MA.

Wheeler, M., Bullock, S., Di Paolo, E., Noble, J., Bedau, M., Husbands, P., Kirby, S., and Seth, A. (2002). The view from elsewhere: Perspectives on alife modelling. *Artificial Life*, 8(1):87–100.

The institutional approach for modeling the evolution of human societies

Simon T. Powers

School of Computing, Edinburgh Napier University, Edinburgh, UK.
S.Powers@napier.ac.uk

Abstract

Artificial Life is concerned with understanding the dynamics of human societies. A defining feature of any human society is its institutions. However, defining exactly what an institution is has proven difficult, with authors often talking past each other. This paper presents a dynamic model of institutions, which views institutions as political game forms that generate the rules of a group's economic interactions. Unlike much prior work, the framework presented in this paper allows for the construction of explicit models of the evolution of institutional rules. It takes account of the fact that group members are likely to try to actively create institutional rules that benefit themselves at the expense of others. The paper finishes with an explicit example of how a model of the evolution of institutional rewards and punishment for promoting cooperation can be created. It is intended that this framework will allow Artificial Life researchers to address how human groups can create conditions that support cooperation. This will help to both provide a better understanding of historical human social evolution, and help in understanding the resolution of pressing public goods problems such as climate change.

Introduction

Artificial Life is concerned with the simulation and synthesis of living systems. One key type of living system that Artificial Life seeks to understand through simulation and synthesis is human social organization. The goals behind this are many and varied, from wanting to better understand the ecological and social pressures that historically transformed human groups from egalitarian hunter-gatherers to hierarchical chiefdoms and states, to being able to devise incentive schemes to prevent climate change, to being able to engineer artificial systems that autonomously adapt their social organization to changing conditions. All of these efforts lie at the interface with a number of other disciplines that are concerned with understanding human social organization, including anthropology, archeology, artificial intelligence, economics, evolutionary biology, primatology, political science, and psychology.

In this paper, I review the different approaches that have been used to model the cultural evolution of human societies. I argue for the merits of an institutional approach. Following Hurwicz (1996), I define institutions as political game forms that generate the rules of a group's economic interactions. This is in contrast to other work that has tended to define institutions either as equilibrium behavior within a society, or as the rules of the economic interactions themselves. Instead, I show that by viewing institutions as political game forms that generate these rules, we can develop dynamic models of how societies change over time, allowing us to better address the goals of Artificial Life researchers.

Two big questions about human societies

When we look at human societies, two big features stand out as being is in particular need of explanation. The first is the high level of cooperation and coordination between unrelated individuals. Compared to other primates, humans are unique in depending upon exchange with other individuals for nearly all of their vital resources. In economics, this high degree of interdependency is known as catallaxy, and contrasts heavily with the autarky and self-reliance of other primates. Strikingly, the degree of interdependence has increased over time from the first hunter-gatherers through to modern day states (North, 1990). For hundreds of thousands of years, humans lived as hunter-gatherers, obtaining resources by hunting large animals and gathering plant materials (Marlowe, 2005). Studies of extant hunter-gatherer groups imply that ancient hunter-gatherer groups practiced extensive food sharing between camp members (Boehm, 1999), and that there was a marked division of labor between males who hunted large animals, providing protein, and females who gathered plants, providing carbohydrates (Marlowe, 2007). With the Neolithic origin of agriculture that began circa 10,000 years ago, division of labor further increased, with some individuals specializing entirely in tasks unrelated to food production, such as producing crafts (Oka & Kusimba, 2008). Where we see such high levels of specialization elsewhere in the biological world, it is only in cases where there is a very high genetic relatedness between group members, as exemplified by eusocial insect colonies. In such cases, the division of labor is coordinated by means of a common genetic program carried by each individual. But in human societies, division of labor and exchange occurs between unrelated individuals that may never meet again, what Paul Seabright (2010) calls "A company of strangers". This creates all kinds of opportunities for one party to cheat on an exchange (North, 1990), while the fact that interactions in modern societies are between unrelated individuals who may never meet again is problematic for traditional explanations for cooperation based upon kinship and reciprocity.

The second key feature of human societies is their transition between egalitarian and hierarchical modes of social organization (Currie *et al.*, 2010). Both anthropological and

archeological evidence imply that the first human social groups were egalitarian hunter-gatherers. Anthropological studies of modern hunter-gatherer groups show that decisions are invariably reached by a group consensus being formed, with each individual being allowed to voice its opinion in a group-wide discussion (Boehm, 1999). While such groups do have leaders, the role of leaders is not to coerce others or monopolize the discussion, but rather to facilitate turn-taking and help the group reach a consensus. Archeological evidence of burial sites similarly reveals little status differentiation when individuals were buried (Price, 1995).

By contrast, the transition to agriculture was accompanied by a shift to hierarchical social organization, with a small number of individuals exhibiting high status. Evidence from burial sites shows that leaders started to be buried with valuable grave goods such as obsidian, and were not buried alongside other group members as had occurred previously (Price, 1995). Hierarchy was manifested both in resource inequality, and in inequality in decision-making, with leaders at the top of the hierarchy coercing the rest of the group to follow their decisions. The archeological evidence points to the first hierarchical societies being chiefdoms, with a single level of hierarchy, i.e. a chief presiding over commoners. The origin of states around 4000 years ago is defined in terms of a shift to multiple levels of hierarchy, with rulers creating specialized administrative positions between themselves and the commoners (Spencer, 2010). This represents a new form of division of labor and specialization, where some individuals specialize in administering the group.

What we see in human evolution, then, is a gradual increase both in hierarchical organization, and in the degree of division of labor and specialization. These co-occur with an increase in group size. Hunter-gatherer bands would have numbered no more than the hundreds. Cemetery evidence shows that the origin of agriculture brought about a massive increase in fertility (Bocquet-Appel, 2011), while further evidence suggests that the population density of early agriculturalists may have been up to 40 times larger than that of hunter-gatherers (Hassan & Sengel, 1973). This is supported by evidence that the first cities arose during this period. Finally, in modern states economic interactions occur between millions of individuals. To understand societies, what Artificial Life needs is a dynamic model of how cooperation, hierarchy, and group size co-evolve. In the next section, I introduce the critical role that institutions play in this.

Institutions

What do economic interactions within groups look like? In modern groups, individuals take part in a range of interactions, from bilateral exchange through to the production and maintenance of public goods upon which the whole group depends, such as clean air. These have traditionally been modeled as pairwise reciprocity, and *N*-player public goods games, respectively. However, these models abstract away from the fact that human economic interactions are universally governed by rules. These rules change what the optimal economic behavior for self-interested individuals is. The rules are created by institutions, and are referred to here as institutional rules. Institutions, in turn, are the processes that create the economic rules.

Institutions and institutional rules are not an invention of modern society; they exist even in hunter-gatherer groups (Kaplan *et al*., 2005). For example, extant hunter-gatherer groups have rules specifying who may take part in hunting an
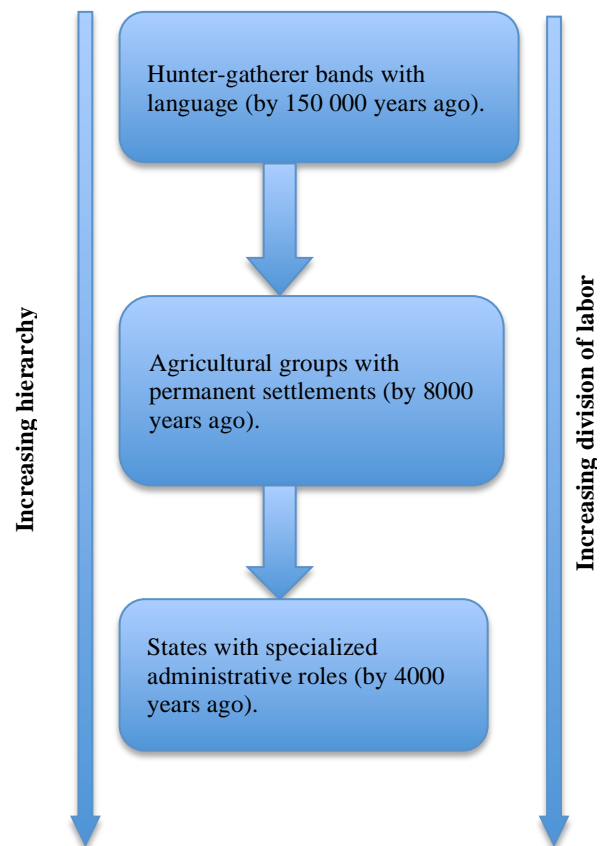
Figure 1: Human social evolution (adapted from Powers, van Schaik & Lehmann, 2016).

animal, who gets to keep which part of the kill, how the food will be shared back at the camp, *et cetera* (Hill, 2009). Similarly, the origin of agriculture necessitated the creation of rules of property rights, to prevent one individual from simply having their crops taken by another (Bowles & Choi, 2013). Agriculture would also have required rules to regulate the construction and usage of new collective goods such as irrigation systems; such rules are seen in extant small-scale farming villages (Ostrom, 1990). Finally, trade in the medieval period required rules to allow a trader to ascertain the reputation of new trade partners, as in the Law Merchant system in Europe (Greif, 2006). With regard to the present, it has been argued that institutions are the main determinant of whether whole nations succeed or fail (North, 1990; Acemoglu & Robinson, 2011).

The processes by which institutional rules are created has also changed over the course of human social evolution. Although institutional rules typically change slowly, over many generations, they are nevertheless not the result of random drift-like processes, but instead are actively shaped by group members pursuing their own interests. Specifically, we should expect each group member to try to create institutional rules that will benefit itself and its kin. In extant hunter-

gatherer groups, institutional rules are routinely discussed by all group members around the camp fire (Boehm, 1999). By contrast, with the rise of agriculture leaders started to dominate the creation of institutional rules, creating rules that benefitted themselves (e.g. by reinforcing inequality) at the expense of the rest of the group.

The story of human social evolution, then, is a story about how institutions and institutional rules have changed over time (Powers, van Schaik & Lehmann, 2016). How have institutional rules been created that allow for successful trade between individuals who may never meet again (North, 1990)? Why have some groups been able to create institutional rules that move their economic game form away from the Tragedy of the Commons when sharing common resources such as irrigation systems or fisheries (Ostrom, 1990)? And why did the processes that create a group's institutional rules change from egalitarian in hunter-gatherers, to extremely hierarchical in the first states?

Where institutional rules are included in models, they usually take the form of rewards for cooperative behavior, or punishment for uncooperative behavior. But this is often done by assuming that each individual alone makes a unilateral decision about whether to punish or reward another group member (so-called "peer-punishment" and "peer-rewarding"), and pays a cost on its own for doing so. However, in reality rewards and punishment follow agreed rules and are done in a coordinated by the whole group, so that no one individual bears the cost alone (Baumard, 2010; Guala 2012; Powers & Lehmann 2013).

The important question is then, how are the institutional rules formed? Very few models have actually looked at this question. The few models that have looked at coordinated rewards and punishment have often assumed that the reward or punishment scheme is determined exogenously by processes outside of the model. While this approach is useful for looking at the effects of various institutional rules, it cannot address how or why institutional rules change over time. What we need is a model of the evolution of institutional rules, a dynamic model that accounts for how institutional rules adapt to changing ecological conditions (Ostrom, 2005).

A framework for modeling the creation of institutional rules

Hurwicz (1996) provides a general model for this. Hurwicz defines an institution as a political game form, which sets the rules for a subsequent economic game form. In game theory, a game form consists of the set of allowed strategies plus the mapping between strategies and outcomes. A game then consists of the game form plus the individual preferences over outcomes, i.e. the player's utility functions. Separating the game form from the game is useful because the game form represents the parts that can be changed by institutional rules, i.e. the parts that are malleable to human intervention (Hurwicz, 1996). In the political game form, the individual strategies consist of messages, and the outcomes consist of rules. The material payoffs that individuals earn are then determined by playing an economic game form, such as a public goods game, that is governed by these rules. For example, the political game form may consist of individuals agreeing that each group member should contribute a certain amount to the public good, and that any individual that

contributes less than this will be punished by an agreed amount. Material payoffs are then assigned by playing the public goods game with these rules (Figure 2).

In the presence of an institution then, individuals engage in two stages of social interactions, where the first (political) sets
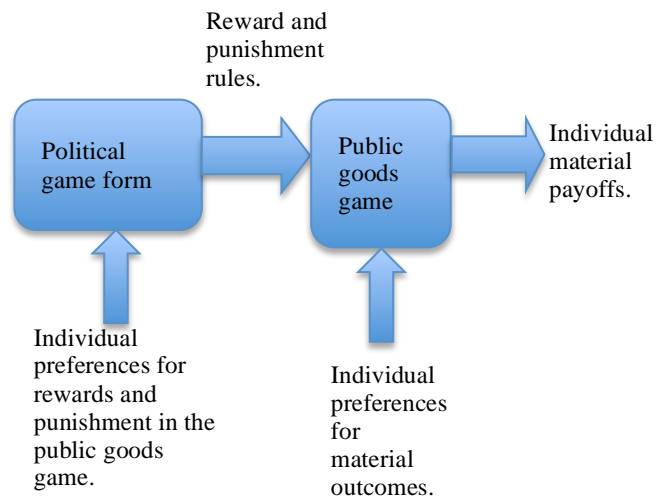
Figure 2: An institution is represented by a political game form, which determines the rules for subsequent economic interactions.

the rules for the second (economic). Different sets of institutional rules generated in the political game form will change the way that self-interested individuals will behave in the economic game form. In other words, the results of the political game form will determine whether cooperation is favored or not.

What might the political game form look like? In hunter-gatherer groups, it is typically of an egalitarian nature where the preferences of all group members are taken account of (Boehm, 1999). From a modeling point of view, this could be operationalized by forming institutional rules by taking some aggregate of the preferences of each group member for the rules. By contrast, with the origin of agriculture, and subsequently the first states, political game forms became much less egalitarian (Price, 1995; Earle, 1997). Through unequal access to resources, leaders became able to dominate the political game form and create rules that benefitted themselves at the expense of others. An example of this is institutional rules that determine how the surpluses resulting from agriculture are distributed within groups. In hunter-gatherers, institutional rules meant that food was shared relatively equally within groups (Boehm, 1999). With the transition to agriculture, however, despotic leaders created rules of distribution in which most resources went to themselves and their kin (Powers & Lehmann, 2014). In these cases, models of the political game form should give weight to the amount of resource that a group member has, in contrast to the egalitarian political game form appropriate for modeling hunter-gatherer groups.

This raises an important question: how do political game forms change over time? The political game form itself has rules, determining how the preferences of individuals result in rules for economic interactions. These rules can themselves change. In the general model of Hurwicz, the rules of the political game form are set by a preceding game form, which can be thought of as a "constitutional game form" (Ostrom, 2005). The constitutional game form might model, for example, a transition between egalitarian and hierarchical interactions within groups. Of course, the rules for the constitutional game form themselves have to come from somewhere, and they may be set by another preceding game form. However, there will not be an infinite regress of game forms, because eventually the rules will be given by unchangeable aspects of the environment, such as the total amount of resources available to individuals, and the laws of physics (Hurwicz, 1996; Ostrom, 2005).

One criticism of the Hurwicz model might be that in reality institutions change very slowly, and that institutional evolution is highly path dependent. The model presented here can take account of this, however. In particular, the political game form does not have to be played on the same timescale as the economic game form. For example, the economic game form may be played many times over the course of a generation, while the political game form may only be played once every several generations. Further, the political game form takes account of path dependence because it is constrained by rules set by the constitutional game form, which will typically be played even less frequently. In this way the model combines intentional change, where self-interested actors actively try to create rules to benefit themselves, with historical contingencies. The balance between the effect of historical contingencies and the effect of intentional action is an empirical question that can only be determined by examining the institutions in question.

Comparison with other approaches to modeling institutions

The two main approaches in the literature have been to view institutions either as the rules of the economic interactions themselves (e.g. North, 1990; Ostrom, 1990), or to view institutions as equilibrium patterns of behavior within groups (e.g. Richerson & Henrich, 2012). The problem with both of these approaches is that they struggle to explain institutional change. Viewing institutions as rules recognizes that they can be produced by intentional action. In other words, it recognizes that institutions are the means by which humans create their economic interactions (North, 1990). However, we also need a model for the processes that generate the rules. Following Hurwicz (1996), it is argued here that the essence of an institution is a political game form that generates rules, as well as the rules themselves.

In cultural evolution models, it is common to view institutions as equilibria (see e.g. Richerson & Henrich, 2012). The idea here is that different social groups reach different stable equilibria (for example as modeled by Boyd & Richerson, 1990), i.e. settle on different institutions. This is compatible with the model presented here to the extent that different institutional rules, i.e. different outcomes of the political game form, will lead to different equilibria in the economic game form. However, the two approaches make

very different predictions about the processes by which groups move between equilibria. In the "institutions as equilibria" model, institutional change is a result of random drift-like processes followed by competition between groups. This is commonly referred to as cultural group selection (Richerson & Boyd, 2005), and is inherently a slow process because variation is only selected at the group level.

Moreover, the change of institutions by cultural group selection is expected to be discontinuous, with long periods of stasis interspersed by sudden and large change when between-group competition events occur and groups suddenly jump to a new and previously unreachable equilibrium. Between-group competition must typically take an extreme form in order to shift another group to a new equilibrium, for example, the extinction of whole groups and the recolonization of their sites by members of other groups, as modeled by Boyd & Richerson (1990), for example. However, the sudden and complete change of behavioral equilibria predicted by these models is at odds with empirical observations of institutions. Rather, most institutional change is gradual (North, 1990; Ostrom, 1990). For example, the reliable enforcement of exchange contracts by state courts in Europe followed from the informal enforcement mechanism of the Law Merchant courts for traders in medieval Europe (North, 1990). Similarly, the institutional rules that provided for cooperative use of the *huerta* irrigation systems in southern Spain described by Ostrom (1990) developed gradually by trial-and-error tinkering of rules over a 1000 year period. Indeed, the empirical work of Ostrom suggests that sudden imposition of different institutional rules by those outside of the group is likely to lead to a reduction in cooperation. This is because what works in well in the particular environmental conditions of one group will typically not work well in another environment, even if both groups face a similar problem such as managing an irrigation system (Ostrom, 1990; Baumard, 2010). It is also because social groups operate with norms and other informal constraints that cannot simply be changed by fiat (North, 1990).

By contrast, the "institutions as political game forms" model presented here allows institutional rules to change as a result of the intentional action of agents over shorter timescales. This fits well with the cognitive skills of humans, including language and shared intentionality (Tomasello & Carpenter, 2007). It accounts for the fact that self-interested individuals should be expected to try and craft institutional rules that benefit themselves in economic interactions. While cultural group selection posits that between-group interactions are the driving force in institutional change, the model here assumes that institutional rules are affected by the within group processes of bargaining and negotiation between self-interested individuals (the political game form). Institutional rules are predicted to typically change gradually, and to be increments of the preceding rules. The cause of change is that one or more individuals estimate that the cost to themselves of changing the rules is more than offset by the subsequent gains that they will receive under a new economic game form. When institutional rules change, the direction of that change depends upon the preferences of individual group members, and the corresponding bargaining strength of the individuals in the political game form (North, 1990; Reiter, 1996).

The fundamental difference between North and Ostrom is the type of cooperative interaction that they focus on. North focuses on the dyadic exchange of private resources, i.e. trade. As he stresses, the reason that cooperation is not a problem in the neoclassical model of exchange is that both parties to the exchange are assumed to have perfect information, and the exchange is assumed to happen simultaneously and with perfect enforcement of contracts. In reality these conditions are never perfectly met. Asymmetries in information mean that one party may know more about the goods to be exchanged than the other, or may be able to exploit the lack of perfect enforcement, leading to a Prisoner's Dilemma situation (North, 1990). North is interested in how institutional rules can avoid this from happening, and hence how the neoclassical gains from dyadic trade can be realized. He is quick to point out, however, that there are just as many institutional rules which do not promote cooperation as there are rules that do. He contrasts the effects of institutional rules in Third World economies with those of the West in terms of their different effects on economic growth. Ostrom, on the other hand, is concerned with the exploitation of common resources such as irrigation systems and fisheries. Her empirical work shows how the right kind of institutional rules can avert the Tragedy of the Commons (Ostrom, 1990). It also shows that institutional rules can fail to avert the Tragedy. This implies that if we are going to make policy interventions to try and increase cooperation, then we need a dynamic model of the evolution of institutional rules in order to understand what kinds of changes might or might not promote cooperation.

Other work has looked at the effect of various institutional rules on the evolution of human cooperation (e.g. Sasaki *et al.* 2012; Chen *et al.*, 2014). A key question this work has addressed is whether rewards or punishment are more effective at promoting cooperation in public goods games. While this work has examined the effect of varying the magnitude of rewards or punishment, it has treated the amount of reward or punishment as an exogenous parameter of the model. Consequently, these models have not addressed why groups would actually settle on different reward and punishment schemes. Essentially, the models have looked at the effects of varying the outcome of the political game form, but have not actually modeled the political game form itself and so have not addressed how the institutional rules actually evolve.

The next section provides an example of how the general Hurwicz model can be instantiated as a dynamic model of the evolution of institutional rules.

A simulation model of the evolution of institutional rewards and punishment

The model presented here is largely based upon that presented in Powers & Lehmann (2013), but modified to allow groups to reward cooperators as well as punish defectors. Individuals carry three cultural traits that are passed from parent to offspring subject to a mutation rate, μ. The first trait determines whether individuals cooperate and produce B units of public good at a cost of C to themselves, or whether they defect and produce no public good, and hence pay no

cost. Mutation on this trait involves changing to the other type. The second trait is a preference, h, (range 0 to 1, inclusive) for the proportion of the group's public good that should be used for helping, i.e. distributed between all group members to increase their payoff. The remaining proportion of the public good is then used to pay for institutional rewards and punishment. How this is divided up between reward and punishment is determined by the third trait that individuals carry. Specifically, individuals have a preference for what proportion, r, (range 0 to 1 inclusive) of the remaining public good should be used to reward cooperators as opposed to punish defectors. Mutation on these preference traits is done by adding a small random number drawn from a normal distribution with mean 0.

Unlike Powers & Lehmann (2013), which modeled structured populations, here individuals interact in randomly drawn groups of size n (without replacement from the global population). Groups are reformed every generation. Within groups, individuals play a political game form followed by an economic game form. The political game form determines H, the proportion of a group's public good that is used for helping. It also determines R, the proportion of the remaining public good that is used to reward cooperators as opposed to punish defectors. The model assumes an egalitarian political game form in which each group member's preference is weighted equally. H and R are then set by taking the mean of each group member's preference (without regard to whether the individual is a cooperator or a defector). This is then followed by the economic game form, which is modeled as a linear public goods game. Cooperators contribute to the public good, and may be rewarded for doing so, depending upon the outcome of the political game form. Defectors do not contribute and may be punished for this. The fitness of cooperators (w_c) and defectors (w_d) is then given by:

$w_c = (HBn_c)/n - C + (1-H)RBE,$
$w_d = (HBn_c)/n - [(1-H)(1-R)BEn_c]/n_d.$

E is the efficiency of the institution, i.e. the rate at which public good is converted into rewards or punishment, n_c is the number of cooperators in the group, and n_d the number of defectors in the group. The term $(1-H)RBE$ represents the rewards to cooperators given by the institutional rules decided by the group members in the preceding political game form. Similarly, the term $[(1-H)(1-R)BEn_c]/n_d$ represents the punishment given to defectors according to the agreed institutional rules. Crucially, because H and R depend on individual traits h and r, these institutional rules themselves evolve by individual selection.

After the public goods game has taken place and fitness determined, all individuals in the global population compete to form a new population of size N by fitness proportionate selection, i.e. individuals leave descendent offspring in proportion to their fitness. Generations are non-overlapping.

Results

A full analysis of the model will be presented elsewhere. The purpose here is to illustrate how institutions as political game forms can be modeled in simulation.

The results (Figure 3) demonstrate that cooperation-promoting institutions can result from a political game form, even when each individual taking part in the political game form is following its own self-interest. In the absence of

rewards or punishment, cooperation would not be stable, and would have a long-run frequency close to 0, using the parameters in Figure 3 (since it is a standard result that in such cases cooperation would only be stable when $B/n > C$, such that the actor's share of the benefit it produces is greater than the cost). However, we see that while individuals evolve to invest most of their public good in the benefit of helping, they do invest enough into institutional reward and punishment to maintain cooperation. Occasionally the average h-trait becomes very close to 1, meaning that little is invested in rewards or punishment. In these cases cooperation collapses. However, cooperation is quickly recovered once the h-traits start to become slightly smaller again, creating sufficient rewards such that cooperation again pays more than defection. Previous work (Powers & Lehmann, 2013) suggests that these fluctuations may not happen in structured populations.

One important finding is that individuals evolve to create institutions that mainly use rewards rather than punishment to support cooperation. Chen et al. (2014) argued that the optimal strategy should be for groups to use rewards when cooperation is rare, but then switch to punishment once cooperation is common. Such a policy minimizes the expenditure necessary to favor cooperation. However, the results presented here suggest that while this policy may be the optimum, this does not mean that the evolution of institutional rules will necessarily settle upon it. The model here suggests that when individual preferences for rewards or punishment are evolving, they may tend to favour rewards even when cooperation is common. This highlights the importance of explicitly modeling the process by which institutional rules are generated within groups.

Discussion

Institutions can be defined as political game forms that generate the rules, and hence incentives, for economic interactions (Hurwicz, 1996). Taking this view allows us to produce dynamic models of institutional evolution. This allows us to explore why some groups have historically managed to create institutional rules that foster cooperation, and why others have failed (North, 1990; Acemoglu & Robinson, 2011). Applications to this include understanding the rise of hierarchy and states, and addressing pressing public goods problems such as climate change.

Cultural group selection models have traditionally viewed institutions as equilibria. These models suggest that institutional rules change by a slow process of random drift and between-group competition. However, individuals should be expected to try to craft institutional rules that benefit themselves. This means that institutional rules can also change as a result of within-group processes, often on much faster timescales.

Future work needs to model political game forms in more detail. There is a need for more realistic models of the bargaining and negotiation processes that go on within groups to generate institutional rules. How we can best model the bargaining process between individuals with different preferences for institutional rules? The processes by which political game forms themselves change also need to be modeled. When are political game forms likely to move between egalitarianism and despotism, as happened, for example, with the transition from a hunter-gatherer to agricultural lifestyle 10,000 years ago?

In summary, a framework for modeling institutional evolution has been presented here. An application of the
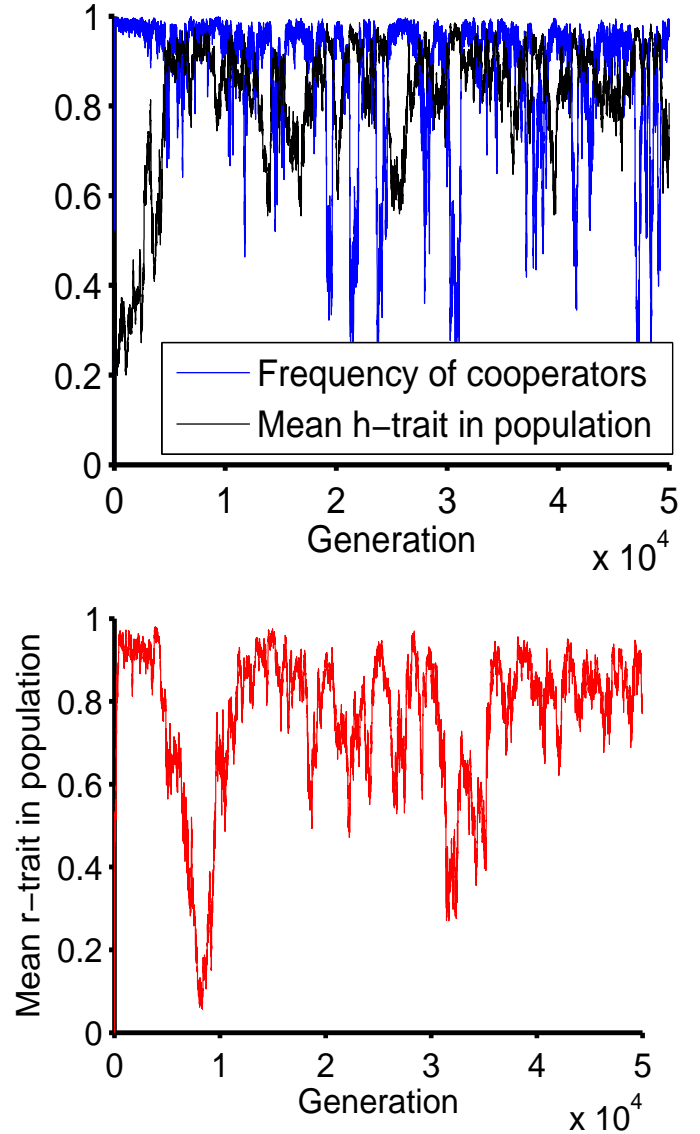
Figure 3: Co-evolution of institutional rules for rewards and punishment alongside individual strategies in the economic game form. Parameters: $n = 15, N = 750, B = 0.9, C = 0.1, E = 0.9$.

framework was illustrated using a simple model of the co-evolution of individual social behaviors, with individual preferences for whether groups should reward cooperators, or punish defectors. The political game form was modeled as an egalitarian process in which the preferences of all group members were aggregated. It is intended that this framework will allow Artificial Life researchers to address how groups can create conditions that support cooperation. In the final section of this paper, I turn to discuss how the institutional

modeling approach might help Artificial Life to address pressing social issues in modern societies.

What can the institutional approach offer to our understanding of societal challenges?

The problem of cooperation in modern societies manifests itself in two forms. The first is in exchange of resources between agents, i.e. trade. Trade may be between individuals at a village market, between firms within a nation, or between nations. The second form of cooperation is in the provision and usage of collective goods, ranging from the management of a local inshore fishery, through to a global reduction in carbon emissions to prevent climate change.

In all of these cases, what determines whether or not a society achieves cooperation is whether or not its institutional rules provide the right incentives to the agents in that society. Do the institutional rules move the economic game form away from a single-shot Prisoner's Dilemma? The agents could be, for example, single individuals, firms, or governments.

As Ostrom (1990) notes, policy prescriptions by economists and other social scientists have traditionally involved externally imposing a solution to a cooperation problem on a society. For trading, this might involve suggesting that a society copy the market rules of a more successful society. For collective goods, suggested policies might include either dividing the good into private shares, or assigning a state body to monitor and enforce rewards and punishments (Ostrom, 1990). But as Ostrom stresses, these imposed mechanisms of institutional change have repeatedly failed. Essentially, this is because what works well in one local environment need not necessarily work well in another. This is both because local environments will tend to differ in ways that affect the economic game form, and because different societies have different local norms and customs. Transplanting institutional rules into a society in which they are not compatible with the norms and beliefs held by the agents within that society is unlikely to work. Furthermore, norms and beliefs typically change very slowly, hence why economics tends to explain changes in behavior in terms of changes in relative prices rather than by changes in tastes (North, 1990).

This suggests that to make successful policy prescriptions we need a bottom-up understanding of how institutional rules change within societies. Traditional models in economics have focused on equilibrium conditions. But such models, along with cultural group selection models, are ill suited to capture the dynamics of institutional evolution, because institutions typically change through many small and gradual changes. And while the Hurwicz framework and similar approaches (e.g. Reiter, 1996) have been proposed in economics, they have not been instantiated in a fully dynamic form that fits particular empirical scenarios.

This is where Artificial Life, and the related field of agent-based economics, comes in. At its very core, Artificial Life is concerned with producing the bottom-up generation of behavior. This is exactly what is needed to understand how agent behavior and institutional rules co-evolve. To date, a convincing theory of institutional change has been lacking. A convincing model of institutional change needs to both allow institutional rules to change as a result of individual agent behavior, and to allow for the fact that individual agents are not perfectly rational and have incomplete information about their environment. These are both traditional strengths of Artificial Life.

Artificial Life researchers are also used to dealing with complex systems in which small perturbations can sometimes cause large and unexpected shocks. This is quite likely to occur with institutional evolution, where small changes in the political game form may lead to large changes in the economic game form. Again, the toolkit of bottom-up modeling is well equipped to highlight this.

By using Artificial Life simulation techniques, we can begin to get a handle on the effect that changing institutional rules is likely to have on economic game forms, and on how these changes in the economic game form feed back into changed individual preferences in the political game form. We can also start to appreciate the effect of different political and constitutional game forms on this process. This has previously all lied outside of the scope of static equilibrium models, which has limited the ability of analysts to foresee the implications of policy changes.

References

Acemoglu, D. and Robinson (2011). *Why Nations Fail: The Origins of Power, Prosperity and Poverty*. Crown Publishers, New York.

Baumard, N. (2010). Has punishment played a role in the evolution of cooperation? A critical review. *Mind & Society*, 9(2):171-192.

Bocquet-Appel, J.-P. (2011). When the world's population took off: The springboard of the Neolithic Demographic Transition. *Science*, 333(6042):560-561.

Boehm, C. (1999). *Hierarchy in the Forest: The Evolution of Egalitarian Behavior*. Harvard University Press.

Bowles, S. and Choi, J.-K. (2013). Coevolution of farming and private property during the early Holocene. *Proceedings of the National Academy of Sciences of the United States of America*, 110(22):8830-8835.

Boyd, R. and Richerson, P. J. (1990). Group selection among alternative evolutionarily stable strategies. *Journal of Theoretical Biology*, 145:331-342.

Chen, X., Sasaki, T., Brännström, Ã., and Dieckmann, U. (2015). First carrot, then stick: how the adaptive hybridization of

incentives promotes cooperation. *Journal of The Royal Society Interface*, 12(102):20140935+.

Currie, T. E., Greenhill, S. J., Gray, R. D., Hasegawa, T., and Mace, R. (2010). Rise and fall of political complexity in island South-East Asia and the Pacific. *Nature*, 467(7317):801-804.

Earle, T. (1997). *How Chiefs Come to Power: The Political Economy in Prehistory*. Stanford University Press, Stanford, CA.

Greif, A. (2006). *Institutions and the Path to the Modern Economy: Lessons from Medieval Trade*. Cambridge University Press, Cambridge, UK.

Guala, F. (2012). Reciprocity: Weak or strong? What punishment experiments do (and do not) demonstrate. *Behavioral and Brain Sciences*, 35(01):1-15.

Hassan, F. A. and Sengel, R. A. (1973). On mechanisms of population growth during the Neolithic. *Current Anthropology*, 14(5):535-542.

Hill, K. (2009). Animal "culture"? In Laland, K. N. and Galef, B. G., editors, *The Question of Animal Culture*, pages 269-287. Harvard University Press, Cambridge, MA.

Hurwicz, L. (1996). Institutions as families of game forms. *The Japanese Economic Review*, 47(2):113-132.

Kaplan, H., Gurven, M., Hill, K., and Hurtado, A. M. (2005). The natural history of human food sharing and cooperation: A review and a new multi-individual approach to the negotiation of norms. In Gintis, H., Bowles, S., Boyd, R., and Fehr, E., editors, *Moral sentiments and Material Interests: The Foundations of Cooperation in Economic Life*, pages 75-113. MIT Press.

Marlowe, F. W. (2005). Hunter-gatherers and human evolution. *Evolutionary Anthropology: Issues, News, and Reviews*, 14(2):54-67.

Marlowe, F. W. (2007). Hunting and gathering: The human sexual division of foraging labor. *Cross-Cultural Research*, 41(2):170-195.

North, D. C. (1990). *Institutions, Institutional Change and Economic Performance (Political Economy of Institutions and Decisions)*. Cambridge University Press.

Oka, R. and Kusimba, C. (2008). The archaeology of trading systems, part 1: Towards a new trade synthesis. *Journal of Archaeological Research*, 16(4):339-395.

Ostrom, E. (1990). *Governing the Commons : The Evolution of Institutions for Collective Action*. Cambridge University Press.

Ostrom, E. (2005). *Understanding Institutional Diversity*. Princeton University Press, Princeton, NJ.

Powers, S. T. and Lehmann, L. (2013). The co-evolution of social institutions, demography, and large-scale human cooperation. *Ecology Letters*, 16(11):1356-1364.

Powers, S. T. and Lehmann, L. (2014). An evolutionary model explaining the Neolithic transition from egalitarianism to leadership and despotism. *Proceedings of the Royal Society B: Biological Sciences*, 281(1791):20141349.

Powers, S. T., van Schaik, C. P., and Lehmann, L. How institutions shaped the last major evolutionary transition to large-scale human societies. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 371.

Price, T. D. (1995). Social inequality at the origins of agriculture. In Price, T. D. and Feinman, G. M., editors, *Foundations of Social Inequality*, pages 129-151. Plenum Press, New York.

Reiter, S. (1996). On endogenous economic regulation. In *Economic design*, volume 2, pages 211-243. Kluwer Academic Publishers.

Richerson, P. J. and Boyd, R. (2005). *Not by Genes Alone: How Culture Transformed Human Evolution*. University of Chicago Press.

Richerson, P. and Henrich, J. (2012). Tribal social instincts and the cultural evolution of institutions to solve collective action problems. *Cliodynamics*, 3(1):38-80.

Sasaki, T., Brännström, Ã., Dieckmann, U., and Sigmund, K. (2012). The take-it-or-leave-it option allows small penalties to overcome social dilemmas. *Proceedings of the National Academy of Sciences of the United States of America*, 109(4):1165-1169.

Seabright, P. (2010). *The Company of Strangers*. Princeton University Press, Princeton, NJ.

Spencer, C. S. (2010). Territorial expansion and primary state formation. *Proceedings of the National Academy of Sciences of the United States of America*, 107(16):7119-7126.

Tomasello, M. and Carpenter, M. (2007). Shared intentionality. *Developmental Science*, 10(1):121-125.

Human Crowd Simulation: What Can We Learn From ALife?

Rui Filipe Antunes[1,2], Nadia Magnenat-Thalmann[1]

[1]MIRALab, Université de Genève
[2]BioISI, Faculdade de Ciências, Universidade de Lisboa
rfantunes@fc.ul.pt

Abstract

One of the key components of the suspension of disbelief in real-time 3D simulations is the apparent authenticity of actions and gestures played by the individuals of the virtual population. This paper addresses this aspect of simulation, by investigating ways to improve the behavioral realism of virtual humanoid characters in groups and small multitudes. We look at the framework of ALife, identifying and analyzing existing bio-mimicking techniques that can be used in this context and contribute towards the improvement of the plausibility from the generated simulations. By looking at the literature, we identify some of the key elements from ALife that are being progressively incorporated in the simulations of groups and crowds. Then, we discuss a generative model for spontaneity and heterogeneity where bio-inspired agents are individualized with DNA-like strings and appear organized hierarchically exchanging token units of energy, mass, and resources. The result is a generative population of agents that self-organize and interact autonomously, exhibiting interesting social dynamics based on biological tenets and an economy of resources. We analyze this simulation quantitatively with the purpose of studying the impact of each of the previously identified techniques.

Figure 1: Global view of the environment of the experimental setting created for this paper.

Introduction

Animation of crowds in the historical site of Pompeii (Maïm et al., 2007), or visitors in theme parks (Shao and Terzopoulus, 2006) are good examples of a developing area of research that looks at modeling virtual spaces inhabited by communities of humanoid avatars that self-organize and interact autonomously. Commercial video games, such as The Sims (Electronic Arts, 2015), Assassins Creed 4 (UbiSoft, 2014) and Grand Theft Auto (Rockstar Games, 2015) also share similar goals with great success in terms of mainstream appeal. Traditionally, the algorithms for modeling crowds attempt to simulate the realistic behaviors of the crowd at the macro-level, including the features of its spatial flow (Helbing, 1992; Hughes, 2003). Recently, more attention has been put into the micro-level, centered on individual behavior within a multitude (Weizi and Allbeck, 2011; Park et al., 2012). The challenge is to be able to create complex scenes in real-time, with generative populations of virtual humans, interacting autonomously, where behaviors resemble the variety-rich feel of the real world.

Human biology, psychology, social organizations, and relationships form complex networks within which behaviors occur. In this complex matrix, the cognitive and biological systems act as elementary forces in generating and shaping motivations. Physiological and psychological processes are dynamic, and two individuals sharing similar initial conditions may act in different ways when facing identical stimuli according to their past experiences and environmental context. In that sense, groups of humans can be described as complex adaptive systems since they act as a form of decentralized, distributed processing, where their internal states and the environment interoperate in feedback loops. Influenced by dynamic variations in individual motivations one to one interactions occurring at the local scale lead to changes in the observed patterns emerging higher at the group and population levels.

One area of knowledge sharing the interest in related themes of complex phenomena is Artificial Life (ALife), a discipline characterized by the study of complex processes observed in organisms and communities. We are interested

in investigating how the framework of ALife can benefit the field of crowd simulation, namely for its emphasis on the phenomenon of Emergence and Self-organization. Bio-inspired systems, known as Computational Ecosystems, are part of this framework. These are multi-agent systems where individuals appear organized in a hierarchical way (in the form of a food-chain), and traditionally agents have their motivations based on their self-sustenance and the perpetuation of their genetic patrimony. The different internal states of each of the agents during their regular activity (search for energy, fight, eat, etc.), generates individual differentiation at the local scale of the community and permanently changes global patterns and flows.

We have built on this type of systems to develop an agency model for generative populations of humanoid characters with social dynamics based on biological tenets and an economy of resources. We further developed a simulation using this model with the objective of analyzing its overall behavior quantitatively. The purpose of this study is to understand the impact and benefits of techniques from the framework of ALife implemented in the context of behavioral simulation of humans in groups and multitudes.

We have organized the paper as follows. First, we discuss the objectives and contextualize this work with related work in crowds and group simulation. Then, in the section Methodology, we provide details of the agency model, and its implementation in a population of autonomous trading agents (Figs. 1 and 2). In the section Results, we discuss the outcome of this experiment, bringing up the advantages and disadvantages of this model to conclude suggesting future possibilities for research and development drawing on this approach.

State of the art

Attention to metabolic functions has been, traditionally, at the core of ALife practice with multi-agent systems (Yaeger, 1994; Taylor and Hallam, 1997; Dorin, 2009). Traditionally, agents require a permanent input of token units of some sort from an external source. Usually, these tokens are identified as 'energy.' This energy is then converted into useful activity when agents spend it performing their regular activities. This simple mechanism provides an intrinsic motivation to act upon in the world. We see this motivational strategy progressively appearing in a growing number of simulation of humans in an empirical observation of Abraham Maslow predicates of metabolic functions preceding other human activity (Maslow, 1943). Sevin and Thalmman describe an agent that needs to satisfy its hungriness and thirstiness as well as its tiredness consequently needing to eat, drink, rest and sleep (Sevin and Thalmann, 2004). However, this was implemented as a single agent model.

Later, these functions began to be relatively more frequent within groups and multitudes of virtual humans. For instance, we can see these reappearing in Navarro *et al.*

(Navarro et al., 2015), or in Silverman *et al.* (Silverman et al., 2005) and Cassenti's (Cassenti, 2009) work. In AStar, Navarro and colleagues equally implement hungriness and thirstiness as bottom level motivational factors for agency (Navarro et al., 2015). In PMFServ, Silverman's agents act motivated by their energetic requirements, their stress levels, and they also suffer fatigue requiring some resting time to sleep and recover (Silverman et al., 2005). The correlation between stress and metabolic functions causes a corresponding degree of fallibility, in a similar way as their human counterparts.

One of the interesting aspects of metabolic oriented agents such as the above is that they become individually differentiated as their internal states progress differently from each other. We can see a similar approach in Trescak and colleagues (Trescak et al., 2014) where agents similarly have basic metabolic needs with changing levels of hungriness, thirstiness, fatigue and comfort. Since objects have annotated functions, such as 'fish-cook', or 'fish-catch', agents interact with them differently according to their internal states. Moreover, these authors went a step further in their appropriation from ALife's framework. In their simulation of the Babylonian city of Uruk, they represented individuals using a DNA like string, which encodes the visual representation and the social roles played by the agent in the world. Antunes and Leymarie (Antunes and Leymarie, 2013) also explore the DNA mechanism to define hierarchical classes of individuals that establish the social relationships. Another technique these later authors have borrowed from ALife is the one of reproduction. In their simulation, of gregarious humanoids, they have included states of birth and death to create dynamic changes in the population density. The internal states of the agents are also used to trigger differentiated gestural animations depending on the type and outcome of the interactions.

Given the growing of attention that this framework is gaining in the field of group and crowd simulation, it seems opportune to address now this topic of research, and identify and study the impact of its techniques.

Methodology

We defined our experimental setting as a community of autonomous humanoid characters that was built based on a model of agency that draws upon ALife's predicates. The objective is to be able to measure and study its properties in terms of the generated patterns of behavior namely, their diversity and spontaneity.

The model of agency

We got inspired by earlier work on societies of agents from the domain of ALife (Holland, 1996; Yaeger, 1994; McCormack, 2001). The structure of our population was similarly arranged in hierarchical layers organized in a way

Figure 2: Close up of two individuals interacting.

to promote interaction and resources exchange. Permanent changes of patterns and flow characterize these type of multi-agent communities since to satisfy their goals individuals need to move to find interaction partners. We found useful to include a similar and relatively simple motivational layer, where agents need to exchange token units of energy and resources, motivated by their survival and the perpetuation of their genetic patrimony.

DNA based individuals

Individuals in the virtual population are identified by a DNA-like code, which functions as a blueprint defining its particular features. The sixteen binary digits of the DNA structure encode various phenotypic aspects such as *a)* the type of resources produced and those required to perform metabolic functions, *b)* initial parametrization of the psychological component, *c)* the definition of procreation compatibility, as well as *d)* the biological component of the personality. These features establish a class distinction between individuals which allows us to define functional hierarchies organized around the trade of resources and reproduction.

Eq. 1 describes the blueprint, where each of the characters from *a* to *q* in the DNA-set stands for one, or part of one parameter.

$$DNA = \{a, b, \ldots, o, q\}, a, b, \ldots, o, q \in \{0, 1\} \qquad (1)$$

Reproduction

Agents emulate a life cycle, including birth, death (by the lack of energy), and reproduction. When individuals multiply, their progeny inherits half of each of the parent's genetic blueprints using a crossover operation with associated mutation. The mutation operator flips each allele-bit probabilistically (10%). When individuals are born, they appear

in the animation from a predefined arbitrary building. Similarly, when they die, they move to another one before they get removed from the system. This technique allows us to have continuous fluctuations of population density, as well as a discontinuous diversity of functions and roles.

Metabolism

Resources occupy a central role in this scheme. They have a dual function: firstly, they are required to generate energy; secondly, the agent recycles them to produce other types of resources. One hypothetical example, the metabolic function of individual *i* uses one unit of resource type *a* and one unit of resource type *b* to generate one unit of resource type *c* and 100 units of energy.

Each possible action performed by the agent has an associated cost expressed as token units of energy. For instance, the action 'move' will require one token of energy per meter whereas 'run' will use two instead. As a consequence, agents need a regular input of energy. To generate this, each of the agents needs to gather resources and trade the ones he owns by those he needs. This fact gives them an intrinsic motivation to act in the world.

Algorithm 1 describes the metabolic function, where e_t is the value of energy at time *t*, {a,b,c} are the resources owned by the agent and required for its metabolism, and *d* stands for the resources produced, and *k* is a constant value.

Algorithm 1 the metabolic function

1: **for all** time t **do**
2:     **if** $digestionTimeAND\{a, b, c\} \geq 0$ **then**
3:         $e_t \leftarrow e_{t-1} + k * 3$
4:         $\{a, b, c\} \leftarrow \{a - k, b - k, c - k\}$
5:         $d_t \leftarrow d_{t-1} + k$
6:     **end if**
7: **end for**

Psychology

Individuals have their personality and emotional temperament. This is defined with a three-layered psychological model, integrating: *i)* short-term emotions, which result from goal achievement; and a temperamental factor, combining *ii)* a long-term mood, which is the accumulated memory of these emotions; and a *iii)* biological imprint, which is a genetically determined component of the personality of the agent. Mehrabian's PAD (Mehrabian, 1996) is used to represent these personality traits. PAD describes psychological features as three-dimensional vectors of Pleasure, Arousal, and Dominance, where each dimension uses the bipolar space [-1,1].

Eq. 2 describes the biological component of personality, *k1, k2* and *k3* are constant values determined by the DNA-blueprint.

$$personality \leftarrow \{k1, k2, k3\}, k1, k2, k3 \in [-1, 1] \quad (2)$$

Eq. 3 describes the mood component at the time *t*, which is a combination of three vectors: the first with the previous mood at the time *t-1*, a second with the last emotional state, and a third with the agent's personality. α, β, γ are weight coefficients.

$$\vec{mood}_t \leftarrow \alpha\vec{mood}_{t-1} + \beta\vec{emotion} + \gamma\vec{personality} \quad (3)$$

PAD dimensions are also used to define the emotional vector. These are given from affective appraisals of the interactions, as described next.

Reinforcement learning

We have implemented the behaviors using a Markov chain *Fig. 3*. The probabilities associated with each state transition are updated after each transition of state, and more importantly, after every interaction. Rewards of state-transitions have an affective dimension. They incorporate context dependent utilitarian and emotional components. The appraisal of a situation not only updates the weights of the transition states in the chain, but it also originates new emotions. When we calculate the reward of the state transition, we do it accessing its dimensions of Pleasure, Arousal, and Dominance. This appraisal gives origin to a vector with the current emotion. As we have described above, in *Eq. 3*, emotions have a direct impact on the current mood. This, in turn, is also used to determine the outcome of interactions.
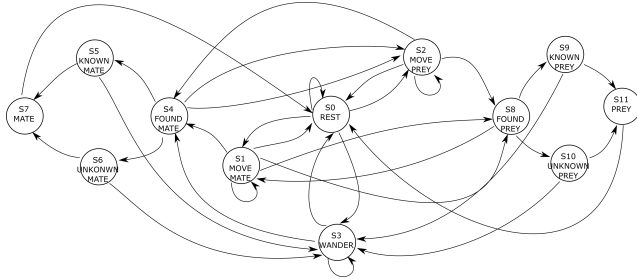
Figure 3: Diagram of the Markov chain. Circles indicate states: S0-Rest; S1-Move to potential mate; S2-Move to potential prey; S3-Wander; S4-Found mate; S5-Previously known mate; S6-Next to an unknown mate; S7-Mate; S8-Found Prey; S9-Next to known prey; S10-Next to an unknown prey; and S11-Attack. Arrows indicate state transitions.

Interactions

Agents interact based on their metabolic and reproductive needs. Sensors for the energetic deficit, hungriness, and sexual arousal drive the desire to connect with others. Interactions are then established based on the DNA definition of the

classes of individuals. As such, when hungry, one individual will only search for potential preys, avoiding all the others individuals. In this context, potential preys are individuals that can provide the resources this one needs and are simultaneously interested in the ones it can supply. The outcome of the interactions depends on *a)* the mood of both partners, and *b)* the utility for both intervenients of this particular interaction.

Algorithm 2 describes one of the possible interactions, the transference of resources {a,b,c} from agent *j* to agent *i*, and the reciprocal transference of {d,e,f} from *i* to *j*, in a cooperative interaction of type 'eat' between the two agents, with energy *e* decremented by the associated cost *k*. α, β, γ are weight coefficients.

Algorithm 2 the eating interaction

$\{a_i, b_i, c_i\} \leftarrow \{a_i + a_j\alpha, b_i + b_j\beta, c_i + c_j\gamma\}$
2: $\{a_j, b_j, c_j\} \leftarrow \{a_j - a_j\alpha, b_j - b_j\beta, c_j - c_j\gamma\}$
$\{d_j, e_j, f_j\} \leftarrow \{d_j + d_i\delta, e_j + e_i\varepsilon, f_j + f_i\zeta\}$
4: $\{d_i, e_i, f_i\} \leftarrow \{d_i - d_i\delta, e_i - e_i\varepsilon, f_i - f_i\zeta\}$
$e_i \leftarrow e_{i-1} - k$
6: $e_j \leftarrow e_{j-1} - k$

A video of the system running is available at `https://youtu.be/_W0KEz52Ksw`. To further analyze quantitatively the simulation we set it running for one hour and we have captured portraits of the population at every minute. The next section presents and discusses some of these results.

Results and Discussion

We analyze the behavior of this population in quantitative terms. Our interest is on putting an emphasis in aspects provided by techniques from the ALife's framework previously identified in the State of the Art: *a)* the DNA blueprint, *b)* Reproduction, *c)* Metabolism, *d)* Psychology, and *e)* Reinforcement Learning.

We have departed from the framework presented by Antunes (Antunes, 2013). This author presents a discussion on diversity and heterogeneity, from which we borrow some of the methodological tools. First and foremost we look at the DNA blueprint as a mechanism that allows control over the diversity of the population. Antunes suggests Shannon and Pielou indexes of diversity and evenness to measure the population level of diverseness and entropy (Antunes, 2013).

$$J^{'} \leftarrow \frac{H^{'}}{H^{'}max} \quad (4)$$

$$H^{'} \leftarrow \sum_{i=1}^{R} pi * ln(pi) \quad (5)$$

In ecological studies, these indexes allow us to determine how distributed the species are (Mulder et al., 2004). For instance, consider a population composed of five foxes and one thousand rabbits. This community is far from even. *Eq. 4* describes this relation, where J' is the evenness factor, ranging in the interval between 0 and 1; the higher the value obtained, the less variability there will be between the species. H' denotes a number derived from the Shannon diversity index and H' max is the maximum value of H' equal to *ln*S (Mulder et al., 2004). Shannon's index of diversity (Shannon, 1948) is used to measure a population's heterogeneity. Claude Shannon introduced this index to measure the entropy in strings of text. *Eq. 5* describes Shannon's index, where the richness of biodiversity in a community H' is a function of the total of individuals *R* and the proportion of individuals *pi* belonging to the *ith* species. The Shannon index increases as both the richness and the evenness of the community increase. Typically, values are between 1.5 and 3.5 in most ecological studies, and the index is rarely greater than 4. In our simulation, we have defined speciation according to the three binary digits of mating criteria in the DNA (eight possible species). The evenness of this run was 0.936. Shannon's index was determined to be 1.946. These results indicate this population being rich, diverse and evenly distributed in genetic terms.

This result points out that communities can have phenotypes evenly distributed and be relatively varied. The DNA technique seems to be useful in the framework of crowd simulation's developers as it introduces an easily implementable mechanism to define and establish hierarchical classes and relationships.

The second aspect under scrutiny was the population density. *Fig. 4* shows the evolution with time of the population density. We can see a rapid increase of the population density from the initial 50 to about 65 and then an even sharper boost to about 140. Then, numbers fluctuate within a variation of about 20 individuals but with a tendentious decline.

The fluctuations observed are an important aspect that we must retain with regards to simulations of humans since this mechanism introduces naturally occurring variations in the number of characters that are simultaneously present in a scene, a situation that is common in reality scenarios involving humans.

The third aspect under scrutiny was the spatial distribution of the population, and the activities performed at each moment of time. *Fig. 5* shows two distinct moments of the run. We took snapshots of the location in space of each of the individuals in the population individual at intervals of one minute each. We then overlapped these snapshots in intervals of ten minutes. *Fig. 5-Top* shows the first period of the run and *Fig. 5-Bottom* the second period. The graph indicates that the population moves, as expected, between the four interest points minimizing their effort using the shortest paths. However, far from a uniform distribution, the popula-
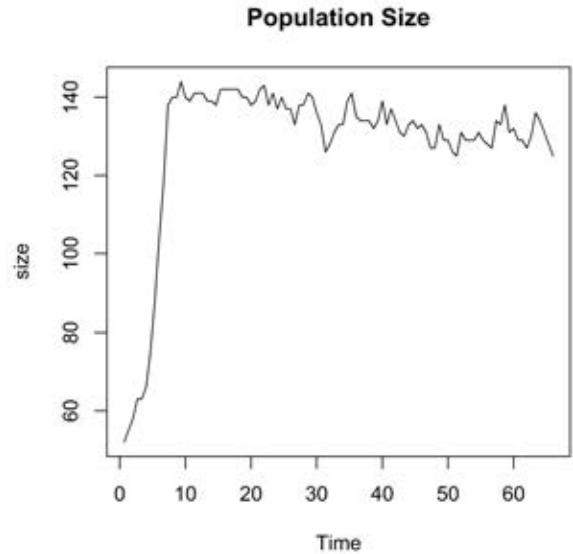
Figure 4: Illustration of the evolution over time of the population density.

tion keeps changing spatially, shifting local attractors, thus occupying the space differently throughout the run. We can see clusters and new paths forming in different spots from image *a* and image *b*.

This result is what we would expect from an ecosystem's dynamics where these local attractors are emerging structures from the self-organization of the individuals in their natural behavior and struggle.

This graph also shows an interesting exploratory behavior, where a considerable number of agents move away from the aggregation areas (the shortest path between the interest points) diverging in the landscape. This pattern results probably from the learning algorithm which allows a dynamic prioritization of goals. Again, this pattern is interesting from the human simulation point of view. Agents not only move between their goals, tendentiously following the less expensive emerging pathways, as they also explore their landscape and aggregate in dynamic clusters that form and reshape. These are patterns of behavior that developers of simulations of human behavior might take advantage in their animations.

On the continuation of this spatial analysis, we found it relevant to also look at the actions performed. We have registered what was each of the agents doing at the exact moment of the snapshot. *Figs. 6* depicts an arbitrary moment of time. We can see that agents were mostly moving from point A to point B. However, others were also engaged in interactions. These interactions differed in nature and occurred spatially disperse in the environment.

We can also observe variation in the internal states when

Spatial distribution

Figure 6: Graph showing a snapshot of the actions that agents are performing at one arbitrary moment of time.
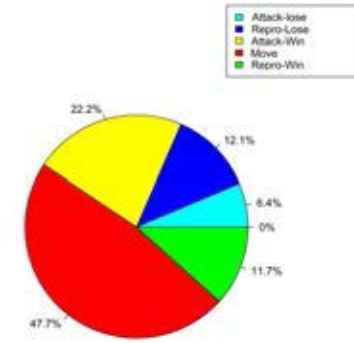
Figure 7: Actions being performed in the world. Summary of the run.

Figure 5: The graph shows a juxtaposition of frames taken at intervals of one minute each, revealing a time-lapses of the spatial flow of the population over time. Top: First ten minutes of the run; Bottom: Second interval of ten minutes. The red circles indicate attractor points where agents go when they have nothing else to do. The arrows in the bottom image point to areas where we noticed a significant change of occupation.

interactions were of an identical type. *Fig. 7-Top* overlaps the interactions throughout the first ten minutes of the run. We found remarkable the level of exploratory behavior shown. However, it seems that individuals find beneficial the strategy of maintaining themselves in areas of high density where interactions tend to occur.

Fig. 7 presents a summary of the activities throughout the run. As expected, agents have spent most of their time (48%) walking. The rest of the time, they were involved in interactions. 29% of their time they spent it in trading activ-

ities (Attack-Lose, Eat), and 24% mating (Mating-Lose, Reproduce). 34% of the time agents were successfully co-operating and only 19% they were recorded interacting with a non-cooperative attitude. We can justify the relatively high value of time spent involved in interactions with the limited number of goals that we have specified initially.

We present the parameters used in the animations in *Fig. 8*. We found large standard deviations in both the duration of the interactions and the maximum number of neighbors. These results confirm the expected high level of variation amongst the individuals, which was our initial goal. The variation of the personal space is much smaller comparatively. We can justify this difference with the fact that we have restricted the boundaries of this parameter to maintain the animation within plausible parameters, with characters interacting in acceptable proximity. Even though, there is still a vast discrepancy in the population as we can see in the visualization of the personal space available in *Fig. 8.2*. The maximum speed was similarly tuned (restricted) for the

43
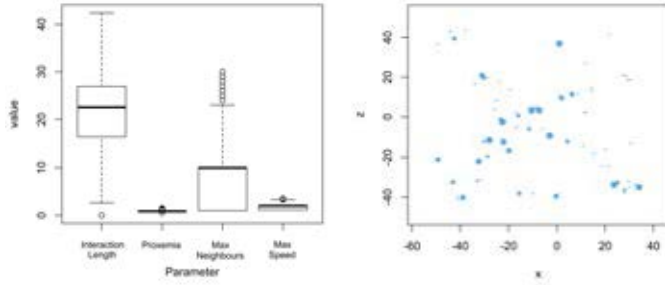
Figure 8: Interaction parameters. Left: Animation parameters (Duration of the interactions, Personal space, Maximum number of neighbors and Maximum speed); Right: Snapshot of the individual personal spaces at an arbitrary moment of time.

animation sake, to avoid characters moving too fast. All these different levels of individual variation are important for group and multitude simulation since they provide multiple degrees of adjustment of individual expression. Following, we proceed with this discussion and will also make the final remarks to conclude the paper.

Conclusions

We have looked at techniques from the framework of ALife to study their impact on the simulation of populations of virtual humans. For this purpose, we have generated a population of autonomous agents whose bio-inspired behavior was implemented drawing upon a set of techniques originated from the framework of Alife. Agents are individually defined with a DNA-blueprint. They simulate a life cycle including death and reproduction, and their metabolism motivates them to search in permanence for new partners to trade useful energy or procreate. Agent's behavior is defined using a Markov chain, with dynamic probabilities updated using intrinsic reinforcement learning. Learning is consequent from appraisals of the interactions of the agent that are both functional and emotive. We have defined a three-layered psychological model, integrating: *i)* short-term emotions, which result from goal achievement; and a temperamental factor, combining *ii)* a long-term mood, which is the accumulated memory of these emotions; and a *iii)* biological imprint, which is a genetically determined component of the personality of the agent. Mehrabian's PAD is used to represent these personality traits.

The resulting population is composed of self-organizing social individuals that are autonomous and able to adapt and prioritize their goals and behaviors, expressing rich and varied behaviors that are relatively consistent and coherent with their past actions. They are capable of spontaneous interactions where personality and emotions play a relevant role.

These interactions are biologically motivated, and its quality is heavily dependent on these psychological traits as they impact their viability, outcome, and duration. Results of analyzing a run of this population seem to indicate that indeed, these techniques play important roles in improving a set of features that contribute raising the levels of realism of the simulations.

We have identified a set of factors gaining direct benefits from the implementation of this framework: *i)* in the first place, spontaneity and heterogeneity are critical aspects when it comes to simulate human multitudes - we found that self-organization, motivated by the agent's basal metabolic and sexual instincts, results in populations showing a high level of heterogeneous and spontaneous behaviors. What this entails for human simulation is the possibility of creation of generative communities that have intrinsic motivations to act and interact autonomously; *ii)* another aspect noteworthy of this study is the potential diversity entailed by this framework - human crowds are not homogeneous in shape and form, and the level of entropy in our simulation, caused by the inclusion of DNA-based reproduction indicates a raised level of differentiation amongst the individuals in the population. As mentioned earlier, this carries the advantage of simplifying the definition of interaction relationships based on class and hierarchies; *iii)* the third aspect of observable benefits are the fluctuations of population density caused by birth and deaths leading to communities whose size and dimension varies over time; *iv)* finally, as a result from their differentiated psychology, we were also able to observe differences in the quality of socialization amongst the individuals, with nuanced variation in their spatial relationship with their neighbors and the rest of the crowd.

From the results above, it seems clear that ALife's framework provides rich tools to be used in the context of human crowd simulators. This framework allows the construction of generative populations of intrinsically motivated agents that: *i)* self-organize and interact autonomously, *ii)* showing spontaneous and heterogeneous behaviors, with *iii)* a high level of diversity and individuality amongst its individuals.

To conclude, we can say that the progressive presence of the bio-inspired techniques originated in ALife in the field of the human group, and crowd simulation is enriching these simulations in multiple and varied aspects that raise in some levels the plausibility of the animations produced. We cannot ignore this growing interest, and this fact has justified the pertinence of a study on the impact of this framework in that type of work. Our contribution fits this need, and in this paper, we have identified and studied some of these structural techniques helping to understand their impact and benefits.

Acknowledgements

References

Antunes, R. (2013). On Computational Ecosystems in Media Arts. *PhD Thesis, University of London*.

Antunes, R. F. and Leymarie, F. F. (2013). Real-Time Behavioral Animation of Humanoid Non-Player Characters with a Computational Ecosystem. In et al., R. A., editor, *13th Conference on Inteligent Virtual Agents (IVA 2013), LNAI 8108*, pages 382–395. Springer.

Cassenti, D. (2009). Performance Moderated Functions Server s (PMFserv) Military Utility: A Model and Discussion. *Army Research Laboratory*.

Dorin, A. (2009). Habitat: Engineering in a Simulated Audible Ecosystem. In Giacobini, M., Brabazon, A., Cagnoni, S., Caro, G., Ekárt, A., Esparcia-Alcázar, A., Farooq, M., Fink, A., and Machado, P., editors, *Applications of Evolutionary Computing*, volume 5484 of *Lecture Notes in Computer Science*, pages 488–497. Springer Berlin Heidelberg.

Electronic Arts (2015). *The Sims, http://gb.thesims3.com/*.

Helbing, D. (1992). A Fluid Dynamic Model for the Movement of Pedestrians. *Complex Systems*, pages 391–415.

Holland, J. (1996). *Hidden Order: How Adaptation Builds Complexity*. Helix Books.

Hughes, R. (2003). The Flow of Human Crowds. *Annual Review of Fluid Mechanics*, pages 169–182.

Maïm, J., Haegler, S., Yersin, B., Müller, P., Thalmann, D., and Gool, L. J. V. (2007). Populating Ancient Pompeii with Crowds of Virtual Romans. In *Proceedings of the 8th International conference on Virtual Reality, Archaeology and Intelligent Cultural Heritage (VAST'07)*, pages 109–116.

Maslow, A. H. (1943). A theory of human motivation. *Psychological Review*, pages 370–396.

McCormack, J. (2001). Eden: An Evolutionary Sonic Ecosystem. In Sosik, J. and Kelemen, P., editors, *Lecture Notes in Artificial Intelligence, Vol 2159, Advances in Artificial Life*, pages 133–142. Springer-Verlag.

Mehrabian, A. (1996). Analysis of the Big-Five Personality Factors in Terms of the PAD Temperament Level. *Australian Journal of Psychology*, 48(2):86–92.

Mulder, C. P. H., Bazeley-White, E., Dimitrakopoulos, P. G., Hector, A., Scherer-Lorenzen, M., and Schmid, B. (2004). Species Evenness and Productivity in Experimental Plant Communities. *Oikos*, pages 50—63.

Navarro, L., Flacher, F., and Meyer, C. (2015). SE-Star: A Large-Scale Human Behavior Simulation for Planning, Decision-Making and Training. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)*, pages 1939–1940.

Park, S. I., Quek, F., and Cao, Y. (2012). Modeling social groups in crowds using common ground theory. In *Proceedings of the Winter Simulation Conference WSC '12*.

Rockstar Games (2015). *Grand Theft Auto 5, http://www.rockstargames.com/grandtheftauto/*.

Sevin, E. d. and Thalmann, D. (2004). The complexity of testing a motivational model of action selection for virtual humans. In *Proceedings of Computer Graphics International (CGI)*, pages 540–544.

Shannon, C. (1948). A Mathematical Theory of Communication. *The Bell System Technical Journal*, pages 379–423.

Shao, W. and Terzopoulus, D. (2006). Populating Reconstructed Archaeological Sites with Autonomous Virtual Humans. In Gratch, J., Young, M., Aylett, R., Ballin, D., and Patrick, O., editors, *Intelligent Virtual Agents*, pages 420–424. Springer.

Silverman, B. G., Badler, N. I., Pelechano, N., and OBrien, K. (2005). Crowd simulation incorporating agent psychological models, roles and communication. In *First International Workshop on Crowd Simulation (V-Crowds 05)*, pages 21–30.

Taylor, T. and Hallam, J. (1997). Studying Evolution with Self-Replicating Computer Programs. In Husbands, P. and Harvey, I., editors, *Fourth European Conference on Artificial Life*, pages 201–204.

Trescak, T., Bogdanovych, A., and Simoff, S. (2014). City of Uruk 3000 B.C. : Using Genetic Algorithms, Dynamic Planning and Crowd Simulation to Re-enact Everyday Life of Ancient Sumerians. In *Proceedings of the Simulating the Past to Understand Human History Conference (SPUH 2014)*.

UbiSoft (2014). *Assassins Creed 4, http://assassinscreed.ubi.com/en-us/home/index.aspx*.

Weizi, L. and Allbeck, J. M. (2011). In *Motion in Games*, pages 132–143. Springer Berlin Heidelberg.

Yaeger, L. (1994). Computational Genetics, Physiology, Metabolism, Neural Systems, Learning, Vision, and Behavior on Polyworld: Life in a New Context. In Langton, C., editor, *Artificial Life III, SFI Studies In The Sciences Of Complexity, Vol. XVII*, pages 263–298. Addison-Wesley.

Towards Emergent Design: Analysis, Fitness and Heterogeneity of Agent Based Models Using Geometry of Behavioral Spaces Framework.

Martin Cenek[1] and Spencer Dahl[2]

[1]Department Computer Science and Engineering, University of Alaska Anchorage, Anchorage AK, USA
[2]Columbia College, Columbia University, New York NY, USA
mcenek@uaa.alaska.edu

Abstract

Detection and analysis of collective behavior in natural and artificial systems is a difficult task which is commonly delegated to a human observer. We present a statistical framework to automatically detect emergent, collective behavior of agents in agent based simulations which exhibit swarming and flocking behavior. Our tunable, transitional-, rotational-, and scale- invariant framework – geometry of behavioral spaces – identifies common behaviors among agents and translates these behaviors into a system's behavioral primitives, along with the agent transitions from one behavioral primitive to another. Finally, we use complex network analysis to detect collectives of agents that gravitate into a common cluster of behavioral primitives as the system's emergent behavior condenses or decays. We apply complex network theory to the analysis of collective behavior dynamics in the simulations of flocking and swarming to validate our analysis. Our framework does not use the knowledge of the parameter space that drive the models, and only relies on the temporal agent trajectories of exhibited behavior. The utility of detecting emergence from exhibited behavior makes this technique suitable as a fitness function for stochastic search algorithms, analyzing evolutionary dynamics of systems with collective behaviors, detecting structures in artificial chemistry experiments, or analyzing physical system such as bacterial formations.

Introduction

Agent Based Models (ABMs) are widely used to study and model collective phenomena in natural and artificial systems. However, there is no universal methodology for how to construct or analyze these systems. The solutions that are found by a group of cooperating agents that only use local agent-to-agent interactions to problems defined on a system-wide level such as foraging for food, designing grassroots movements in societies, or escaping a predator, can frequently only be understood by a human observer. Designing ABMs with cooperative behavior to solve problems is difficult since it required bridging the scope between the design of individual interaction rules and the system-wide problem definition. We present the results of applying previously developed statistically based computational framework to detect the onset and dissipation of collective behavior in a system using only displayed system dynamics – without any knowledge of the parameter space that controls the system or the laws that drive the system's components. Our work is focused on analyzing agent based models, although framework can be used in both the analysis of emergent system behavior and system engineering.

In nature, flocking and swarming are examples of the system-wide behaviors used by organisms to interact with their environment as a collective rather than as individuals. Many natural and artificial systems with collective dynamics share common characteristics of how they are constructed: they are composed of simple agents that interact with each other using simple interaction rules, yet the sophisticated collective dynamics of the ensemble results in the system's evolutionary advantage, survival, or a task completion. We use the flocking and swarming ABM models behavior to test our framework's ability to (1) generalize agent's "noisy" behaviors into common behavior groups and (2) differentiate between the model behaviors that only differ by the velocity and shape of the agent ensemble.

Stochastic search algorithms are commonly used to automatically design the interaction rules of the ABM's agents. These search tools rely on a fitness function to measure the quality of system behavior to solve a problem. Building a fitness function that includes the level of cooperation among agents that leads to desired goal requires significant human expertise and (if successful) results in a custom-tailored, single purpose function that is hard to generalize for ABM design for different environment. Simple fitness functions, on the other hand, often only measure if the simulation reached or made progress towards a goal – regardless of *how* the goal was reached. It is not guaranteed that the resulting system solves a problem using emergent system behavior. Even if the collective behavior is present in the system dynamics it is usually by coincidence and not by design, which is a significant source of error when modelling real-world systems. We show the utility of our framework in measuring the amount and direction of collective behavior in a system, which can be a useful component of a fitness function design to reward the ABMs with cooperative behavior.

Our analytical framework measures both the quality and

quantity of cooperation among agents. Our analytical tool functions independently of the scale and movement of agent behaviors, is applicable to multiple models and is applicable to different optimization techniques. The bee-hive optimization, particle optimization, swarm robotics are examples of additional techniques developed to solve problems using collective, hierarchical system organization. Although these tools share common principles, there is a lack of tools to analyze system behavior for different optimization tools on varying granularity of analysis. Since our analysis only uses the agent behaviors without any knowledge of the model's parameters, the framework is independent of the optimization techniques. This allows for a computational comparison of not only different model executions, but also comparison of results found by different optimization techniques.

The presented results illustrate both the ability to generalize stochastic agent behaviors in the system dynamics into common patterns of behavior and differentiate between two very close collective behaviors of flocking and swarming. Plotting the counts of agents in each prototypical behavior at each time-step of the model's execution allows us to predict the phase transition in the system dynamics from collective to dis-associative and vice-versa. Inspecting the temporal dimension of the behavioral spaces allows us to explicitly measure the level of self-organization, emergence and the directionality of the system dynamics: towards the condensation or decay.

The fitness function design for the stochastic search algorithms to evolve agent systems that solve problems using emergence and analysis of ABM dynamics to analyze the collective behaviors in an ABM model are only two applications of the the geometry of behavioral spaces framework. The framework can also be used to explore the multi-parameter spaces that drive various models, fine tune models to increase performance, or identify a set of high diversity solutions by identifying *how* they solved a problem. With alternations, the framework can be used as a general pattern finding engine regardless of application.

Background and Motivation

The hierarchical system decomposition (HSD) (20), pattern oriented modeling (POM) (4) and morphogenetic engineering (MGE) (3) are common methods used to build systems of agents that solve problems. The HSD is a reductionist approach where the system is built by addition of the constituent components and fails to construct systems with non-linear dynamics. The POM requires a spatially explicit landscape for its agents to move across and interact with. The spatial nature of these models allows for the use of spatial statistics to detect local, emergent behavior, but fails to identify the emergent behavior on the global scale. The MGE is the most sophisticated and relies on "expressing" the system's construction to measure its quality. For example, the rule-based swarms and graph-based grammars describe the

system construction and self-organization in computational development swarms (10; 18). The solution's quality is measured as system's ability to perform its task or the difference between the expressed versus desired patterns (14). The POM and MGE made significant progress towards developing quantifiable ABM models. The general tools to analyze the dynamics of these models lag behind.

Analyzing the non-linear system dynamics has been at the forefront of science for years with a few efforts to analyze the dynamics of the individual and agent based models. Miller used a system of non-linear equations to describe the ABM's dynamics to measure the effect of changing the model's parameter configurations to the resulting system dynamics (11). A summary of statistical and mathematical tools to describe the collective dynamics of swarm and multi-agent systems was presented by Lerman et al. (7; 8). The pragmatic efforts of Calvez and Stonedahl resulted in extension and implementation of automatic stochastic search tools to explore the parameter search-spaces of ABM configurations (1; 15). Controlling the vehicle swarms using physics-based expert systems was proposed by Spears et al. (13) while Miner et al. proposed Markov processes to characterize multi-agent behaviors (12).

The individual and system wide dynamics were analyzed using the information theoretic tools by: Van et al. who studied the expected agent behaviors in the KuglerTurvey's ant colony model (6; 17), Lizier et al. used Shannon based entropy to analyze the micro- and macro-level agent dynamics (9), and the behavior of the swarm robots was analyzed by Wang et al. and Lizier et al. also using the information theoretic tools (9; 19).

The geometry of behavioral spaces framework complements previous research by providing a problem domain independent analysis with the following features: the ability to detect emergent processes from the agent behaviors, producing heuristics to tune the system behavior, the ability to filter agent behaviors of varying frequency and visualize the network of behavioral primitives for both high level and detail inspection of the model's constituent behaviors. One feature of our framework that is not addressed by the previous work is the ability to provide a real-time system monitoring to measure the velocity and direction of the system's condensation or decay towards organized behavior.

Methodology

Geometry of behavioral spaces framework is a multi-step process that analyzes recorded agent trajectories for common patterns of behavior. First, each agent in the model records the direction of its movement then the statistics and similarities of agent behaviors at each time step are computed. The second scan of the recorded behaviors is used to construct a behavioral transition state-space and the distribution of how many agents at each time steps were in which behavioral primitive.

a. recorded agent behavior

agent's trajectory / past and future histories δ⁻/δ⁺

b. encoding past and future vectors δ

Location	$v(\delta^-_{xt})$	$C(v(\delta^-_{xt}))$	$v(\delta^+_{xt})$	$C(v(\delta^+_{xt}))$
1	2,1,1,2,2	2,3,0,0	3,3,3,4,4	0,0,3,2
2	1,1,2,2,3	2,2,1,0	3,3,4,4,4	0,0,2,3
3	1,2,2,3,3	1,2,2,0	3,4,4,4,4	0,0,1,4
4	2,2,3,3,4	0,2,2,1	4,4,4,4,1	1,0,0,4
5	2,3,3,4,4	0,1,2,2	4,4,4,1,1	2,0,0,3
6	3,3,3,4,4	0,0,3,2	4,4,1,1,2	2,1,0,2

c. co-occurrence matrix **C**

$C(v(\delta^+_{xt}))$ across columns; $C(v(\delta^-_{xt}))$ down rows.

C	0,0,0,5	0,0,3,2	0,0,2,3	0,0,1,4	...	1,0,0,4	2,1,0,3	2,0,0,3	5,0,0,0
0,0,0,5									
0,2,2,1						1			
0,1,2,2								1	
0,0,3,2							1		
0,0,2,3									
⋮									
0,0,1,4									
1,2,2,0				1					
2,3,0,0		1							
2,2,1,0			1						
5,0,0,0									

d. row clustering to behavioral primitives

$\varepsilon_{(0,2,2,1)}$, $\varepsilon_{(0,1,2,2)}$, $\varepsilon_{(0,0,3,2)}$

e. ε lookup for δ⁻

$v(\delta^-_{xt})$	$\varepsilon_{C(v(\delta^-_{xt}))}$
2,1,1,2,2	$\varepsilon_{(0,0,3,2)}$
1,1,2,2,3	$\varepsilon_{(0,2,2,1)}$
1,2,2,3,3	$\varepsilon_{(0,2,2,1)}$
2,2,3,3,4	$\varepsilon_{(0,2,2,1)}$
2,3,3,4,4	$\varepsilon_{(0,1,2,2)}$
3,3,3,4,4	$\varepsilon_{(0,0,3,2)}$
3,3,4,4,4	$\varepsilon_{(0,0,3,2)}$
3,4,4,4,4	$\varepsilon_{(0,1,2,2)}$
...	...

f. state-space transition matrix **T**

$\varepsilon_{C(v(\delta^-_{xt}))}$ across columns and down rows.

T	$\varepsilon_{(0,2,2,1)}$	$\varepsilon_{(0,0,3,2)}$	...	$\varepsilon_{(0,1,2,2)}$
$\varepsilon_{(0,2,2,1)}$	2			1
$\varepsilon_{(0,0,3,2)}$	1	1		1
...				
$\varepsilon_{(0,1,2,2)}$		1		

Figure 1: Geometry of behavioral spaces methodology: (a) recording of agent trajectories in four cardinal directions, (right) extracting past and future behaviors δ^- and δ^+ at the first six time-steps, (b) compression of extracted histories into a vector, (c) recoding the co-occurrences of encoded past and future histories into the matrix C, (d) clustering similar rows into behavioral primitives, (e) the behavioral primitive look-up for two consecutive past histories during the second scan of the agent's trajectory vector and (f) recording the behavioral transitions in two subsequent steps creating the state-space transition matrix T

The following sections will describe the details of each analytical step: simulation space discretization and recoding of agent trajectories (Figure 1 a), at each time step agent's past and future windows of behavior are first compressed and then logged into the co-occurrence matrix (Figure 1 b, c), then the groups of common past behaviors that have the same or similar enough future behaviors are created and define the behavioral primitives of stable behaviors (Figure 1 d), and finally, the behavioral space is constructed to reveal the system-wide transitions among behavioral primitives along with the likelihood of agents to transition from one behavioral primitive to another stable pattern of behavior (Figure 1 e,f). The final analysis of the behavioral space is used to differentiate if the system dynamics are condensing into the flocking versus swarming behavior or if the system behavior is decaying into a random agent behavior. The complex network analysis is showed in the results section (Figure 3 and 6).

Data Collection and Space Discretization

In order to report agent trajectories, every agent records its initial position and compares that position to its current position. When the distance travelled exceeds a threshold value, the agent resets its initial position to its current position, reports the general direction it traveled, in the case a value between 1 and 4, and repeats this process. In this manner high-frequency (small-scale, repetitive) behaviors are not recorded, as they fall below the threshold for distance travelled from the previous report, whereas large scale movements are more frequently reported because these agents move more quickly and therefore report more often. Using this approach, we can ensure that every agent experiencing the same behavior reports in the same manner, independent of the agent's spatial disposition within the model.

At the end of the simulation, each agents trajectory is a vector of directions that the agent moved with values ranging from one to four, the time-stamp and agent's identifier. At any time-step t, the sequence of δ^- previous moves is called

agent's past history and the δ^+ subsequent moves is agent's future history.

Co-occurrence matrix M

Each agent's past and future histories at each time step t are compressed into a vector of length four, where each vector position (attribute) has the number of times the history contained the move in a given direction (Equation 1). The number of cardinal directions to record agent's movement can vary to yield higher resolution analysis or to reflect the different space tessellations. In this paper, we use four cardinal direction.

$$\text{v}(\delta^-) = \langle \Sigma(1 \in \delta^-), \Sigma(2 \in \delta^-), \Sigma(3 \in \delta^-), \Sigma(4 \in \delta^-) \rangle$$
$$v(\delta^+) = \langle \Sigma(1 \in \delta^+), \Sigma(2 \in \delta^+), \Sigma(3 \in \delta^+), \Sigma(4 \in \delta^+) \rangle$$
$$(1)$$

The co-occurences of encoded past and future behaviors $v(\delta^-)$ and $v(\delta^+)$ are logged into $C[v(\delta^-_{x_t}), v(\delta^+_{x_t})]$ for all time steps k (Equation 2).

$$\text{C}[\text{v}(\delta^-_{x_t}), v(\delta^+_{x_t})] = \Sigma_{\forall k}(v(\delta^-_{x_{t+k}}) : v(\delta^+_{x_{t+k}}))(2)$$

Behavioral Primitives

Each row of the co-occurrence matrix C is a likelihood distribution of a given past behavior resulting in any of the observed future behaviors. The next steps, we group all such rows that are similar to each other, creating clusters of past behaviors that resulted in sufficiently similar future behaviors to be considered a behavioral cluster - a behavioral primitive ϵ (Equation 3). We used the χ^2 test of statistical independence between any two rows that were not previously assigned to a behavioral primitive to measure the similarity between two row distributions of past behaviors. Other tests can be used to calculate this similarity.

$$\epsilon_i = if \left(\chi^2 \big(C[i,:], C[j,:] \big) < \alpha \right) then \ \epsilon_i \cup C[j,:] \ (3)$$

Behavioral Space T

The behavioral matrix $T_{\epsilon \times \epsilon}$ describes the high-level agent behaviors and the transition of agent's behavior from one stable behavior to another. To construct the matrix T, we scanned the agent trajectories for a second time. For each agent, we look up which behavioral primitive agent's past behavior belongs to at two consecutive time steps t and $t+1$ $p = v(\delta^-_{x_t})$ and $r = v(\delta^-_{x_{t+1}})$ respectively. We record the agent transitions between two behavioral primitives into a matrix $T[p, r]$ - a state-space transitional matrix of exhibited behaviors in the simulation (Equation 4).

$$\text{T}[p,r] = \Sigma_{\forall k}\left(\epsilon_p : \big(v(\delta^-_{x_t}) \in \epsilon_p\big), \epsilon_r : \big(v(\delta^-_{x_{t+1}}) \in \epsilon_r\big)\right)(4)$$
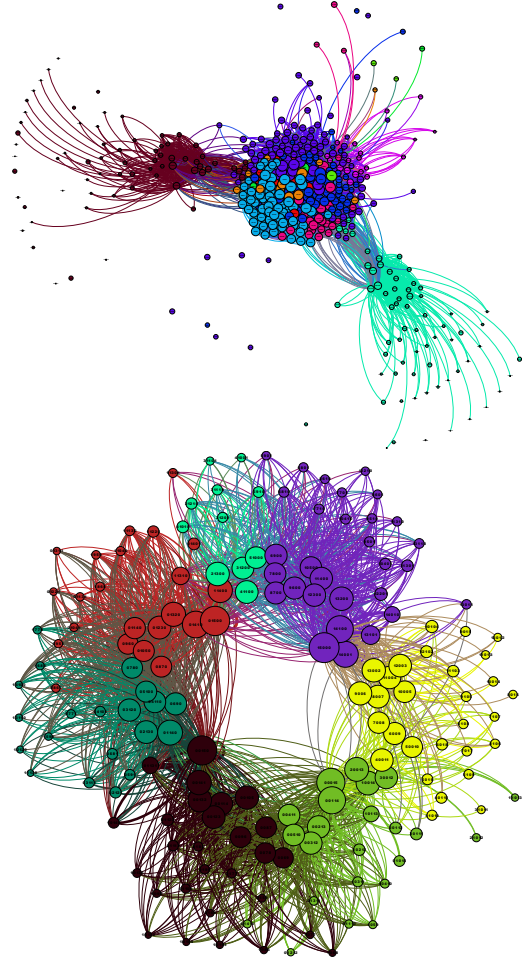
Figure 2: A complex network visualization of the final behavioral matrix T for swarming (top) and flocking (bottom) models with the system behavior varied in 400 time-steps from cooperative to random. The communities of highly connected nodes were further clustered into communities and colored with the same color.

Geometry of Behavioral Spaces

The final step is a complex network analysis of the matrix T, with the behavioral primitives being the network's nodes and the behavioral transitions are the network's edges. We further clustered the tightly coupled behavioral primitives into the communities of closely related behaviors, filtered transition edges with low edge weight, and used a network layout to visualize the behavioral space (5). For additional details of the behavioral spaces analysis, please see (Cenek and Dahl, in press).

To construct a dynamic view of the agent behaviors during the simulation, tracking the progress towards the condensation or dissipation of collective behavior, we counted
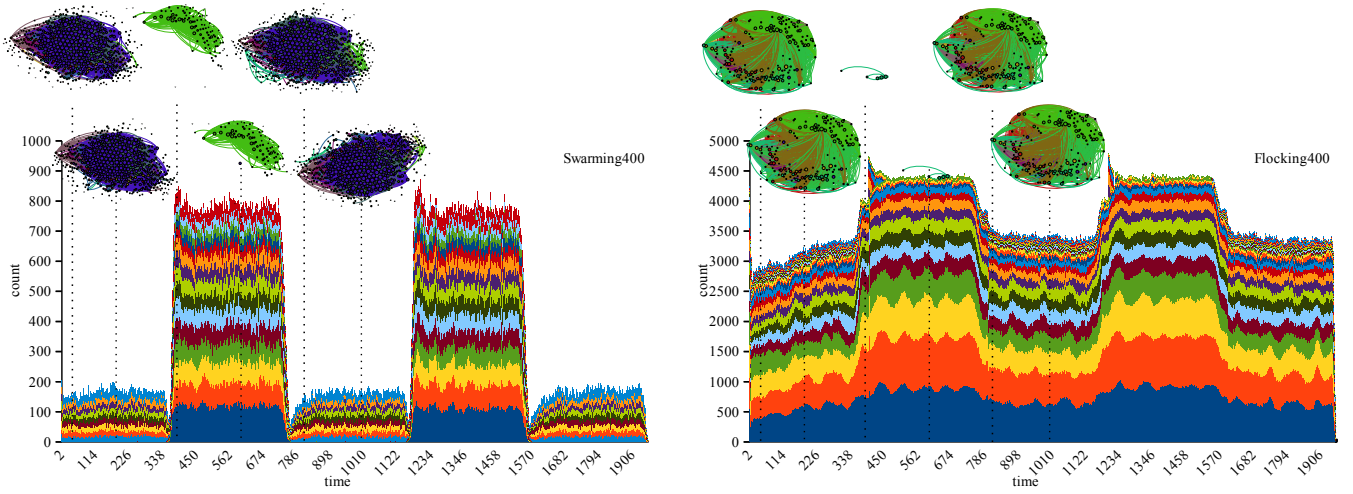
Figure 3: Agent counts in each behavioral primitive at every time-step of the swarming (left) and flocking (right) model executions with the oscillation period between condensed cooperative and decayed random behaviors every 400 time-steps. At each time-step a bar chart shows the top 25 behavioral primitives with highest counts of agents that were selected (globally) at the end of the simulation. Each color of in the bar represents one active behavioral primitive and its size is proportional to the number of agents with that behavior at that time-step. The top of each plot shows a miniature snapshot of the active transition edges of the behavioral matrix T as a complex network 200 time-step intervals.

how many agents, at any given time step, are at which behavioral primitive. Figure 4 shows the histogram of the top 25 most frequent behavioral primitives and the same aspect is showed in Figure 2 where the network nodes representing each behavioral primitive have their size proportional to the count of agents in that behavior (the node's weighted degree).

Results

We applied the geometry of behavioral spaces framework to the flocking and swarming models with global co-operative behavior among agents (16). To illustrate the framework's ability to detect the regime changes between the cooperative and random system behaviors, we varied the model parameters that control agent's ability to coordinate with other agents (alignment, vision, radius etc.) every 200 or 400 time-steps to force the cooperative system behavior or its dissipation into a random, dis-associative behavior.

The models ran for between 1500 and 1900 time-steps which allowed for 7 and 2 regime changes at 200 and 400 time-step periods respectively. The system dynamics were analyzed using the past and future history vectors of 15 components long, four cardinal direction of reporting agent movement, and the threshold $\alpha = 0.05$ for χ^2 measure to group the rows of co-occurrence matrix C into the behavioral primitives.

The complex network view of the behavioral space only shows the behavioral primitives of the giant component with the $degree > 0$ and the minimum edge weight of 10. Figure 2 shows the global behavioral landscape of for two models.

The network's node and edge size is proportional to their weighted degree and weight.

The node clusters with the same color identify the highly connected behavioral primitives (5). These communities of behavioral primitives show common transitions among agent behaviors. For example, the behavioral landscape of the swarming model (Figure 2 top) has the behavioral primitives of condensed cooperative behavior organized as the center cluster of nodes. The peripheral behavioral clusters represent the agent behaviors after system decayed into random behavior. The communities of behavioral primitives in the flocking model (bottom) are interconnected, since the model's condensed emergent system behavior has several small groups of agents that move in one of eight generalized directions. Each direction is seen as one community of nodes with the same color. After the decay of global cooperative behavior, the agents trajectories slowly diverge and increasing number of behavioral primitives are activated. This change of behavior can be seen in the time-slice plots of the active behavioral spaces as the miniature sub-networks (top) in Figures 3 (please note the coloring and layout for these miniature graphs is different than in Figure 2. A window of 20 time-steps was used to show the active edges of the behavioral space at each time-slice (Figure 3 top).

Analysis and Discussion

All system measures presented in our result plots are quantitative measures of system dynamics and can be used to analyze a system's emergent behavior, to be included as a parameter in the fitness function of the stochastic search algo-
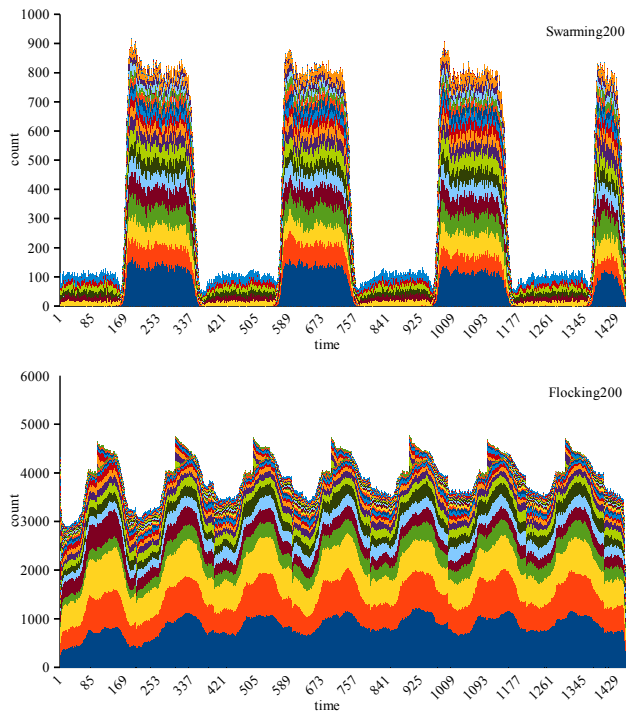
Figure 4: Counts of how many agent behaviors were recorded in each behavioral primitive during the models execution. The x-axis shows all possible behavioral primitives sorted in the alphabetical order from $< 0,0,0,15 > \rightarrow < 15,0,0,0 >$.

rithms, or to identify the high diversity solutions by differentiating the indifferent solutions. Additionally, this simplified and descretized data set is entirely contained in a behavioral matrix, allowing the model to be compared with other models. Machine learning can also be used in order to name behaviors or translate this raw data into human form, although this is also simple to do manually.

The behavioral primitive histogram (Figure 3, 4) and the global behavioral primitives count plots (Figure 5) are the fingerprints of the system's dynamics. The histograms of active behavioral primitives shows regime shift as the change of distributions (and counts) of agent behaviors. The blue color on the bottom of the plot is the most frequent behavioral primitive and represents the "random" behavior. As the system behavior decays, the number of agents with random behavior increases, so do the counts of active behavioral primitives.

The qualitative analysis of the system's behavior is done by construction of the behavioral landscape networks showed in Figure 2. Note that although the framework's parameters are tunable, we ran the analysis "straight out of the box" without fine tuning any of the analytic framework's parameters to identify different behavioral features from the system's dynamics

We executed the model with different random seed, but the analytical framework's results were stable with little variation in the reported dynamics. We do not report the results of the statistical validation and variance of the analytical framework. The stability of the analysis to generalize the system dynamics can be seen as highly correlated agent counts in each behavioral primitive in the series Swarming200 vs. Swarming400 and Flocking200 vs. Flocking400 showed in Figure 5.

Note that the "random" behavior is different in each of the models. In the flocking model, the disordered regime constitutes the agents slowly drifting apart at random which means they dissociate slowly from their ordered flocking and never reach a fully random state. In the swarming model, the random behavior is agents moving in completely random directions almost instantaneously after the model parameters were changed. The regime shifts are showed in the Figure 3, as the sudden increase in the counts of behavioral primitives between the steps 400 and 450. This trend parallels the increase in the system behavior diversity. The difference between how the two systems decay to random behavior is the difference between the sudden increase of in the behavioral primitives counts in the swarming model (left) versus the gradual decay (the gradual increase) in the flocking behavior (right). Behavioral primitives histogram in Figures 3 and 4 show the same dynamics features as the time series distributions in Figure 6. They both can be used to detect when system dynamics are condensing towards stable, cooperative behavior and dissipating into random behavior.

The ability to generalize the stochastic agent behaviors into stable behavioral primitives can be seen in all histogram figures (Figures 3, 4) and the system dynamics measured on the state-space transition networks (Figure 6). The oscillation of model's behavior between the cooperative and random regimes resulted in the histogram's oscillations, but more importantly, the framework repeatedly generalized the agent behaviors to the same behavioral primitives. This is seen as the same color pattern and structure in the histogram's periods. The framework also characterized the agent behaviors to the same behavioral primitives in the independent model executions with the regime changes every 200 or 400. The histogram pattern in the Figure 4 is close to identical to the histogram in Figure 3 if the time axis was compressed.

Figure 6 shows the same observation about the framework's stability to generalize agents' random behaviors. Regardless of the oscillation period between the cooperative and random regimes in different model executions, or the repeated condensation and decay of behaviors within the same model execution, the shape difference between a pair of model's executions is minuscule. The difference in offset between the 200 and 400 plot-lines is because the latter model execution lasted 500 time-steps longer and resulted in higher agent counts in each behavioral primitive.
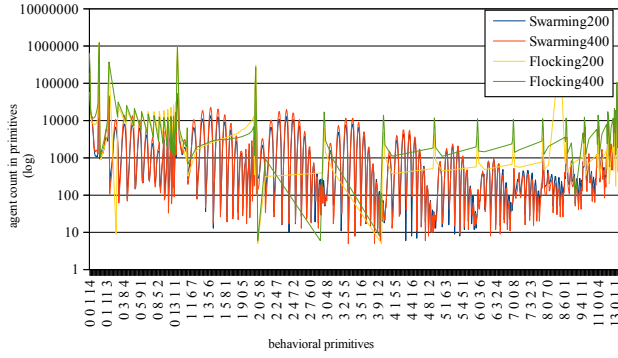
Figure 5: Counts of how many agent behaviors were recorded in each behavioral primitive during the model's execution. The x-axis shows all possible behavioral primitives sorted in the alphabetical order from $< 0, 0, 0, 15 > \rightarrow < 15, 0, 0, 0 >$.

All analytical measures applied to the resulting geometry of behavioral spaces on the system dynamics of flocking and swarming models clearly show the system's condensation towards cooperative behavior and subsequent dissipated towards random behavior. In this context we used the term "geometry" to refer to the structural features calculated using the complex network analysis. Figure 6 shows the overview of the network's measures that can be used to identify system's dynamics towards a regime shift. A sudden increase or decrease in the measure's slope indicate the direction and the velocity of system's impending behavior regime shift towards condensation or dissipation. A stable system behavior (cooperative or random) will have the measured slope near zero.

Stochastic search algorithms are popular tools that do not need to know *how* to find a solution, only *what* needs to be solved and if the candidate solution made a progress towards a desired goal. As explained earlier, designing a fitness function that drives the automatic searches should be inclusive of different aspects of evolved system dynamics. In this case, reward the solutions that (1) use cooperative behavior among agents to solve a problem and (2) are different than the rest of the solutions. If a stochastic search function found multiple methods of solving a given problem, the final state-space transition matrices and the complex measures of the geometry of the behavioral spaces are one way of differentiating **how** different candidate solutions solved the problem. A correlation measure between two system measures in Figure 6 will reveal how different the system dynamics are from each other. This allows for rewarding the solutions with different system dynamics than the rest of the solutions that also solved the problem, creating a metric for originality in computer models and a mechanism for creative problem solving.

In the future we plan to test our methodology on a broader
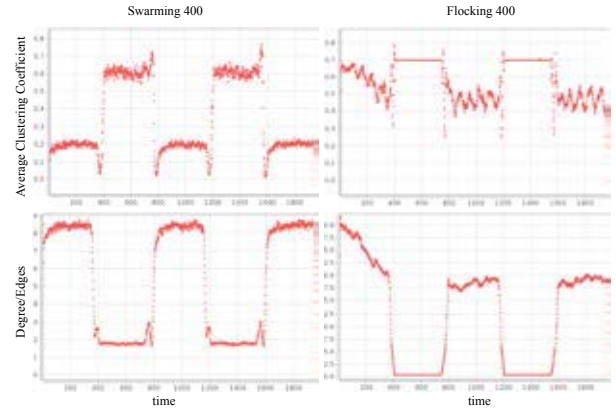
Figure 6: The complex network analysis of active behavioral spaces at each time-step of the model's execution. Examples of the active sub-networks are showed as the miniature graphs in Figure 3. The figure shows the average clustering coefficient (top row) and degree/edge distributions (bottom row) over time for the swarming (left column) and flocking (right column) models with varying condensation and dissipation of system behavior every 400 time-steps.

range of models with emergent system behavior and focus on evolving agent rules using a fitness function that incorporates the measure of cooperation among system agents to achieve the simulation goal. Outside of the ABM application, we plan on using the methodology to analyze the recorded trajectories of bacteria from a high-speed, high-power video feed to automatically detect the formation of bacterial colonies. Using the framework's strength as a general pattern finder, we hope to apply our methodology to detect the emerging signatures in cyber-security intrusion attacks.

Acknowledgements

References

Calvez, B. and Hutzler, G. (2006). Automatic tuning of agent-based models using genetic algorithms. In *MABS 2005: Proceedings of the 6th International Workshop on Multi-Agent-Based Simulation*, pages 41–57. Springer.

Cenek, M. and Dahl, S. Geometry of behavioral spaces: A computational approach to analysis and understanding of agent based models and agent behaviors. *Chaos: An Interdisciplinary Journal of Nonlinear Science (Under review. Pre-print available on request.)*.

Doursat, R., Sayama, H., and Michel, O. (2013). *Morpho-*

genetic Engineering: Toward Programmable Complex Systems. Springer Publishing Company, Incorporated.

Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S. K., Huse, G., Huth, A., Jepsen, J. U., Jrgensen, C., Mooij, W. M., Mller, B., Peer, G., Piou, C., Railsback, S. F., Robbins, A. M., Robbins, M. M., Rossmanith, E., Rger, N., Strand, E., Souissi, S., Stillman, R. A., Vab, R., Visser, U., and DeAngelis, D. L. (2006). A standard protocol for describing individual-based and agent-based models. *Ecological Modelling*, 198(12):115 – 126.

Hu, Y. (2011). Algorithms for visualizing large networks. *Combinatorial Scientific Computing*, 5(3):180–186.

Kugler, P. N. and Turvey, M. T. (1987). *Information, natural law, and the self-assembly of rhythmic movement*. Lawrence Erlbaum Associates, Inc.

Lerman, K. and Galstyan, A. (2001). A general methodology for mathematical analysis of multi-agent systems. *ISI-TR-529, USC Information Sciences Institute, Marina del Rey, CA*.

Lerman, K., Martinoli, A., and Galstyan, A. (2005). A review of probabilistic macroscopic models for swarm robotic systems. In *Swarm robotics*, pages 143–152. Springer.

Lizier, J. T. (2013). *The Local Information Dynamics of Distributed Computation in Complex Systems*. Springer Theses. Springer, Berlin / Heidelberg.

Lobo, D., Fernández, J. D., and Vico, F. J. (2012). Behavior-finding: morphogenetic designs shaped by function. In *Morphogenetic Engineering*, pages 441–472. Springer.

Miller, J. H. (1998). Active nonlinear tests (ants) of complex simulation models. *Management Science*, 44(6):820–830.

Miner, D. and Desjardins, M. (2009). Predicting and controlling system-level parameters of multi-agent systems.

Spears, W., Spears, D., Hamann, J., and Heil, R. (2004). Distributed, physics-based control of swarms of vehicles. *Autonomous Robots*, 17(2-3):137–162.

Spicher, A., Michel, O., and Giavitto, J.-L. (2012). Interaction-based modeling of morphogenesis in mgs. In *Morphogenetic Engineering*, pages 409–440. Springer.

Stonedahl, F. J. (2011). *Genetic Algorithms for the Exploration of Parameter Spaces in Agent-based Models*. PhD thesis, Northwestern University, Evanston, IL, USA. AAI3489404.

Tisue, S. and Wilensky, U. (2004). Netlogo: A simple environment for modeling complexity. In *in International Conference on Complex Systems*, pages 16–21.

Van Dyke Parunak, H. and Brueckner, S. (2001). Entropy and self-organization in multi-agent systems. In *Proceedings of the fifth international conference on Autonomous agents*, pages 124–130. ACM.

Von Mammen, S., Phillips, D., Davison, T., Jamniczky, H., Hallgrímsson, B., and Jacob, C. (2012). Swarm-based computational development. In *Morphogenetic Engineering*, pages 473–499. Springer.

Wang, X. R., Miller, J. M., Lizier, J. T., Prokopenko, M., and Rossi, L. F. (2012). Quantifying and tracing information cascades in swarms. *PLoS ONE*, 7(7):e40084.

Zeigler, B. P., Praehofer, H., and Kim, T. G. (2000). *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*. Academic press.

The BINC Manifesto:

Technology driven societal change, science policy & stakeholder engagement

Steen Rasmussen[1][2]

[1]Center for Fundamental Living Technology (FLinT), University of Southern Denmark
[2]Santa Fe Institute, New Mexico, USA

steen@sdu.dk

Abstract

Throughout history, whenever new technologies have emerged that change our means of production and ability to communicate they have tended to transform society. Spearheaded by digitization, followed by emerging living and intelligent technologies, our world is currently being transformed into something we have difficulty imagining. The transition is likely similar in scale to what we experienced moving from an agriculturally based society to the industrial society, although it will occur at a much faster pace. I present key qualities of our emerging societal transition, discuss underpinning scientific issues, and propose a way the scientific community could engage. Finally, I document how part the European Commission, the Danish Parliament and Press as well as interested stakeholders engage (or not) in the process of creating a "brave new world".

The postindustrial world

Our political institutions, the rule of law, human rights, the banking system, our education system – and even capitalism as we know it – are mainly a products of the industrial age. Using narratives from the industrial age we have learned to navigate the industrial economy as individuals, and as societies we can exert some control to define its shape and limits. But what comes next, in a postindustrial world?[1] Even in the past decade, digital products and services, the internet and mobile technology have changed our lives. This is mainly the result of accumulated advances over the past 50 years and there is much more to come. For example, recent studies (Frey and Osborne, 2013)[2] indicate that digitization is likely to replace about half of known job functions within 20 years.

Thanks to automation, only a small percentage of the population will be needed to produce and distribute what everybody needs. For example, today less than 3% of the Danish population is engaged in agriculture and fishery[3], down from almost everybody some 150 years ago – and these 3% can feed several countries the size of Denmark. As technology becomes more life-like (Rasmussen et al., 2011)[4] more components can be recycled, in the same way that materials are within biological systems. With the development of personal fabricators (Girshenfeld, 2003[5] and Packard et al., 2010[5]) – super-advanced 3D-printers – it's likely citizens will be able to design, share, manufacture and recycle pretty much everything they need locally.

These new technologies are likely to lead to big changes in society, and these could be as drastic as the differences between the Stone Age and the Bronze Age, or between the agricultural society and the scientific age of industry. Inevitably, such a shift leads to changes in economic and political systems, national sovereignty, balances of power, the environment, the human condition, even religion. But this time the changes will not take place over hundreds of years, but within a generation or so. These changes hold promises for amazing possibilities as well as grand challenges.

The BINC Manifesto

Because of these ongoing changes, part of the scientific community is in the process of assembling a so-called BINC Manifesto[7] named after the key converging technologies that shape the ongoing changes: the bio-, info-, nano- and cogno (BINC) technologies. The BINC Manifesto calls scientists and interested stakeholders to action to identify and document observables, trends, mechanisms and key issues concerning the emerging mainly technology driven societal transition. (1) The primary mission is to find out how things are (the facts). (2) Secondary - and separately from (1) - we as citizens and scientists may propose possible scenarios for how to develop our new postindustrial societies.

The Manifesto is organized around five cross-cutting issues, each formulated as a list of scientific conjectures that can be falsified or verified:

A) How is the digital economy different from the industrial?
A1) Digital products and services represent an increasing part of the value creation.
A2) Only the first digital unit requires capital, land and labor, the following copies are practically free of cost. This means profit without production and less need for employees. Further, digital products have no transportation costs, they are global from the moment they are released, so the best products win and take all: there is no market for the second best. However, the threshold is also lower to enter (a fair) market.
A3) The marginal costs of material production approach zero (0) due to automation.
A4) With Personal Fabrication pretty much anything can be manufactured locally (open source software and hardware).
A5) An increasing part of the economy is based on derivative trading (speculation).
A6) Established economic theories are inadequate to address the current reality.

B) Citizens in cyberspace and citizens as biological creatures
B1) Information and communication technology (ICT) design and infrastructure implementation cements power structures (central or decentralized).

B2) Currently, ICT is mostly implemented with a resulting greater concentration of power (government, communication, banking, platforms for social media).

B3) Big business, governments and international intelligence use the digital infrastructure to access private data from the citizens, which means loss of freedom and power for the citizens.

B4) Massive control of the information flows and the associated perception enables modeling of individuals' decision processes and value chains, which in part determines what it means to be human.

B5) Synthetic biology (SB) increasingly makes it possible to alter our genetic makeup. These technologies in a very direct manner have the potential to impact what it means to be human.

B6) ICT and SB generate a significantly more complex world.

C) In the developed economies the middle class and democracy is eroding

C1) There is greater return on investment in speculation than in production, you become more wealthy from being rich than from working.

C2) Information, humans and money can travel freely across national borders. On the individual level, everybody increasingly participates in one global job market.

C3) Businesses move to places where they don't need to pay taxes. Nation states compete among each other to provide the lowest taxes.

C4) The middle class is increasingly challenged to provide the tax revenue needed to secure good nation state governance.

C5) Economic elites are taking over the political power and democracy is deteriorating. Economic and hence political power gets concentrated on still fewer hands.

C6) Fair markets are manipulated by search engine algorithms when they have monopoly.

C7) Elections can be manipulated by search engine algorithms.

D) The global interconnectedness also means global interdependency and that no nation state can take care of their citizens alone.

D1) We have entered the Anthropocene. There is only one environment, e.g. local consumption generates global warming and human impact is causing a mass extinction of species.

D2) The converging bio-, info-, nano- and cogno- (BINC) technologies, which are developed everywhere, will transform the world faster than ever before into something we have difficulty imagining as the pace of these new inventions increases exponentially.

D3) The global population continues to grow and is predicted to reach 9 billion around 2050, accompanied by a wide range of migration issues and cultural clashes. Also the emergence of, and migration to, mega-cities have created new local-global communities.

D4) It becomes increasingly challenging to align radically different economic, cultural and governance structures, i.e. traditional Arabic, industrial Russian and digitized Scandinavian as individuals from previously distant cultures are now mixing.

D5) There are no global institutions in place that can handle this transformation, nor do we have the necessary legal frameworks and theories.

E) The need for new narratives.

E1) The political spectrum of left and right used to be about capital interests. Left and Right emerged from the industrial society and the struggle for power between workers and capital. Today, increasingly the largest corporations don't own production capital in the traditional sense (e.g. Google, Facebook, AirB&B, Uber, Amazone) and our pensions depends on the performance of the stock market. Today, in the West, from left to right there is consensus about the open society, liberal democracy, market economy, and some measure of public welfare (disagreement on level of taxation and social benefits), but overall, systemic agreement about the model. As a result, voters are uniting along different lines e.g.: globalization; defense; cyberspace privacy; sustainability; new public management; balance of cities and countryside; financial sector regulation.

E2) Postmodern deconstruction, globalization and the above mentioned erosion previous narratives are undermining our grand narratives about reality, which used to keep societies together, i.e. religion, nation and class - and to some extend also science. As they lose their explanatory power, some are re-discovered in totalitarian form.

E3) The only grand narrative that has survived is "the free market", which provides consumer goods efficiently but is incapable of solving any of the problems stated above. In fact, it fuels them.

Science, policy and stakeholder engagement

Experiences from national and international science and technology advisor activities regarding these issues are documented through interviews (film), text and policy initiatives. Further, experiences are discussed regarding stakeholder and citizen engagement. Finally, as an example, it is documented and discussed how and why part of the Danish Parliament and the Press regrettably have detached themselves from part of reality and now live in a post-factual subculture[7] with respect to the impending societal impact of technology.

References

1 https://theconversation.com/tomorrows-technology-will-lead-to-sweeping-changes-in-society-it-must-for-all-our-sakes-36023.

2 http://www.oxfordmartin.ox.ac.uk/downloads/academic/The_Future_of_Employment.pdf

3 https://www.cia.gov/library/publications/the-world-factbook/fields/2048.html

4 http://dl.acm.org/citation.cfm?id=2001579

5 https://edge.org/conversation/personal-fabrication

6 http://cordis.europa.eu/fp7/ict/fet-proactive/docs/ie-sept10-14-packard_en.pdf

7 http://raeson.dk/2015/vincent-hendricks-vi-kan-ende-i-det-postfaktuelle-demokrati-hvor-sandhed-er-ligegyldig-saa-laenge-du-kan-saette-soesaette-robuste-narrativer/

8 https://en.wikipedia.org/wiki/Manifesto & http://flint.sdu.dk/index.php?page=technology-impact

9 http://www.postindustrialworld.org

The Carried Network Demarc

David H. Ackley

University of New Mexico, Albuquerque, NM 87131

ackley@cs.unm.edu

Abstract

Software-based artficial life will increase the robustness, and enable vastly increased size, of computing systems. To enhance human potential and protect individual liberty in future society-scale systems, the boundary between 'private' and 'public' digital spaces—known in telephone networks as a *demarcation point* or "demarc"—should be set so that a significant amount of physical computing machinery can be counted as fundamentally *personal*, for assigning rights and responsibilities. To that end, this note offers a principle called the *carried network demarc*: **The machines that you routinely carry under your own power, and their contents and interactions, should be considered *part of your body* as a matter of law and social norm**. Such machines today may be as prosaic as a watch, pacemaker, or cellphone, but in the future you may regularly carry machines inhabited by multitudes of beneficial alife creatures—akin to the bacterial microbiomes that surround and perfuse our biological bodies—that would likewise be considered you and yours in both their physical and computational aspects. The author solicits input from others with expertise bearing on this topic.

Physical and computational convergence

In the ubiquitous modest-sized computers of today, the 'random access memory' organization makes the physical location of the hardware components largely irrelevant to machine operation. But in any sufficiently large computational system—for example as envisioned using *indefinitely scalable* computer architectures (Ackley, 2013)—actual physical distances and computational or communications distances are inherently coupled by the speed of light. In such fundamentally *spatial computers* (Beal et al., 2012, e.g.), physically close components are inevitably faster and cheaper to access than remote ones, and they are more likely to share fate under the large and small vagaries of reality.

Existing location-free concepts of computation like "cyberspace" and the "cloud" not only fail to capture but actively obscure the physicality of computation, with the often-overlooked consequence that the "computation" and the "user" are imagined to exist, somehow, in utterly unrelated spaces. We argue that view is not only manifestly false but also insidiously dangerous—and the carried network demarc proposal, in part, attempts to reframe it.

The idea that a human "self" is physically identical with its natural "meat" body is certainly obvious, but to apply that notion uncritically in future converged physical/computational environments would put the individual human at a crippling disadvantage. Such a view, by default, would expect the human to attend to tasks that artificial entities will routinely delegate to other artificial entities—not just high-level information-processing jobs like sorting email and other interruptions, but also far more fundamental and autonomic tasks like maintaining location awareness and performing continuous threat and opportunity assessment within one's physical/computational surroundings.

We should expect such low-level processing to be protected by limits stronger than just property law. The state or other actors should not be allowed to impede it without the most extraordinary cause, because such an intervention should be viewed as less like a civil forfeiture or a contract negotiation tactic and more like unwanted brain surgery. As our *world* becomes a converged physical/computational world, our *bodies* must be allowed to do the same.

The carried network demarc

Of course, as always in discussions of the rights of individuals in societies, the problem of rights limits, overlaps, and conflicts must be addressed. Especially in this case, where we are proposing a high level of individual protection, there must be limits—and importantly, "natural" or obvious limits—to the extension of that protection. The *carried network demarc* proposed in this paper's abstract is an attempt to make room for, but set natural limits on, our machines to be considered part of our bodies. We read it informally as "you are what you carry" (or `#OurMachinesOurBodies`) and argue it represents a plausible "sweet spot" along a spectrum of viewpoints.

For example, a narrower approach could draw the "body" boundary at your skin, or some close approximation to it. Such a view would allow an implanted pacemaker to be "you," but not a cellphone. An even more restrictive view would hold that no *manufactured* object can be "you" regardless of purpose or location, not even a pacemaker or

bone screw. At the other end, a more expansive alternative would rope in all your property, from your car to your vacation homes to that squash racquet you've forgotten you own.

We argue that "you are what you carry" is a better compromise than those alternatives. Although in the future there may well be myriads of devices literally under our skin, monitoring or maintaining our health, it would seem at least inelegant to require we implant or otherwise ingest our sensorimotor interfaces to the computational world, just to earn them equivalent protection. On the other hand, allowing someone to claim arbitrary property as "self", even when they do not interact with it and are unaware of its status, strains the key notion of *utility for ongoing processing* that is intended to underlie the notion of the extended body.

One final alternative for this brief note: Why not use an *actual* network demarc as the body's demarc in computational space? In modern telephony, a *Network Interface Device* ('NID') forms the demarcation point between private and public utility portions of the network. With one pair of wires running into the house and another pair running up the telephone pole, the NID is a clean and well-understood solution to dividing network rights and responsibilities. Unfortunately, the NID is a clean solution only if all transferred data actually moves through the device—but in the converged physical/computational world, data moves not just by wired and wireless networks, but also video cameras and all manner of environmental sensors public and private. There simply is no clean chokepoint through which all data transfers will flow. The carried network demarc recognizes that *some* basic expectation of a boundary is required nonetheless.

Related work

Questions of self and technology cut across human endeavors; here we touch briefly on technology itself, philosophy, and law. Mann (1997) pioneered advances in wearable computing and augmented reality (Azuma, 1997, is an early survey); the carried network demarc stands to regularize and strengthen protections for such wearable machinery.

In the other direction, the "Internet of Things" (Al-Fuqaha et al., 2015) exemplifies the accelerating technological convergence of our physical and computational environments—as does the growth of automated surveillance (Lyon, 1994). Under the carried network demarc, the individual is free to deploy a "computational skin" made of *living technology* (Bedau et al., 2013)—to interact with, but also to insulate the individual from, potentially massive environmental computing powers. And, crucially, manufacturers of such living technology cannot be faulted for striving ceaselessly to make such machines loyal only to their individual.

From a more philosophical perspective, Froese (2014) offers a recent exploration focused, like the current proposal, on technology placed in or near the physical body—and conjectures, as do I, that living technology stands to offer a positive benefit-risk balance.

And finally, legal aspects will be paramount. To this non-lawyer computer scientist, following Lessig (2009), the United States Constitution looks like legacy software for a distributed operating system—itself forked from a much older codebase dating to the massively refactored Justinian Code (Blume, 2009), released in A.D. 534. And as usual in complex software, there's often more than one way to implement things. In a recent controversy over cellphone encryption, for example, several authors (Hart and Vance, 2016, e.g.) offer attacks and defenses framed by Fourth Amendment prohibitions against unreasonable search and seizure. It will take a shift in thinking, but the carried network demarc will surround your future cellphone with a Fifth Amendment defense *against self-incrimination*.

Call to action

As technological society advances, exactly where to draw the line between self and non-self is never precise. But to enhance human potential and protect individual liberty, it must be possible to include a significant amount of manufactured computing and communication machinery under protections as strong as those accorded to our bodies and our minds. Though cellphones have served here as an example, today they are far too brittle and untrustworthy for life inside the carried network demarc. We can do fundamentally better.

The purpose of this paper is to seek complementary expertise and to open discussions on how to ensure the future technological world makes adequate room for us as individuals, citizens, and humans. The goal is to guide the coming physical/computational convergence into the powerful and empowering mechanism for human liberty, development, and knowledge that it can—but is far from certain to—become.

Acknowledgments

References

Ackley, D. H. (2013). Bespoke physics for living technology. *Artificial Life*, 19(3_4):347–364.

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., and Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys Tutorials*, 17(4):2347–2376.

Azuma, R. T. (1997). A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385.

Beal, J., Dulman, S., Usbeck, K., Viroli, M., and Correll, N. (2012). Organizing the aggregate: Languages for spatial computing. *CoRR*, abs/1202.5509.

Bedau, M. A., McCaskill, J. S., Packard, N. H., Parke, E. C., and Rasmussen, S. R. (2013). Introduction to recent developments in living technology. *Artificial Life*, 19(3_4):291–298.

Blume, F. H. (2009). The Annotated Justinian Code, 2nd edition. At `https://www.uwyo.edu/lawlib/blume-justinian/ajc-edition-2/`.

Froese, T. (2014). Bio-machine hybrid technology: A theoretical assessment and some suggestions for improved future design. *Philosophy and Technology*, 27(4):539–560.

Hart, G. and Vance, C. (2016). Privacy, encryption, and the Fourth Amendment. *The Huffington Post*. `http://www.huffingtonpost.com/gary-hart/apple-iphone-encryption-privacy_b_9299170.html`.

Lessig, L. (2009). *Code 2.0*. CreateSpace, Paramount, CA, 2nd edition.

Lyon, D. (1994). *The electronic eye: The rise of surveillance society*. U of Minnesota Press.

Mann, S. (1997). Wearable computing: a first step toward personal imaging. *Computer*, 30(2):25–32.

Origins of Life and Protocells

Digital Replicators Emerge from a Self-Organizing Prebiotic World

B. Greenbaum[1] and A. N. Pargellis[2]

[1]Tisch Cancer Institute, Icahn School of Medicine at Mount Sinai, New York, NY 10029
[2]Bonsall, CA 92003
apargellis@yahoo.com

Abstract

Some artificial chemistries model the synthesis, evolution and complexity of digital life consisting of sequences of computer operations (opcodes) that are driven by point mutations to compete for memory and CPU time. One of us previously built Amoeba, a computer world inspired by Tierra and designed to study the emergence of self-replicating sequences of opcodes from a prebiotic world initially populated by randomly selected opcodes. Eventually an "ancestral opcode sequence" would emerge. The current version of Amoeba uses a computationally universal opcode basis set and the same addressing as Tierra. It was previously thought such changes would preclude the emergence of self-replicators. Instead, these modifications radically affect the emergence of self-replicators from the primordial soup; Amoeba exhibits a self-organization phase, after which self-replicators emerge. First, the opcode basis set becomes biased. Second, short opcode "building blocks" are propagated throughout memory space. When sufficiently dense, these prebiotic sequences combine to form self-replicators. Self-organization is quantified by measuring the time evolution of n-opcode sequence frequencies, the size distribution of sequences, and the mutual information of opcode pairs.

Introduction

Artificial computer worlds have been used to study such diverse topics as: artificial chemistry architectures (Suzuki, 2011; McMullin, 2012), the synthesis of life (Ray, 1992), emergence of life (Pargellis, 1996a), modeling life (Adami, 1995), biological complexity (Adami, 2000).

There has been considerable debate as to how self-replicators can emerge from a primordial "soup" of initially random computer operations ("opcodes"). One hypothesis is that replication may require two or more cooperating entities (Eigen, 1971; Tangen, 2010).

Amoeba is an artificial chemistry specifically designed to study the process of self-organization in a prebiotic world that eventually leads to the emergence of self-replicators. Amoeba's memory space is initially loaded with opcodes randomly selected from a set of 25 possible opcodes. The Amoeba programs compete for memory space and CPU time and evolve through point mutations (Pargellis, 2003).

The original version ("Amoeba-I") used a limited set of 16 possible opcodes and a memory topology similar to that in Avida (Adami, 1998) where virtual CPUs operated on short sequences of opcodes situated on a 2D "interaction grid" (Pargellis, 1996b). Complements of the opcodes themselves were the addresses so it was impossible to move to arbitrary positions in memory. Amoeba-I could not simulate an infinite Turing tape as there were no stacks assigned to the CPUs (Adami, 1998).

Amoeba-II added two stacks for each CPU and expanded the opcode set (Pargellis, 2001). While these opcodes formed a computationally universal set, the addressing used opcode::address pairs, making it difficult to navigate throughout memory.

Amoeba-III used a new topology for opcode memory space; the 2D interaction grid was replaced by 500 parallel "mini Turing tapes" or bands, each consisting of thousands of opcodes (Pargellis, 2003). Amoeba-III still used opcode::address pairing.

Although some biasing of the original opcode basis set was observed (Amoeba-II and –III), there was no additional self-organization of opcode sequences into "building blocks." Instead, a randomly generated sequence of opcodes, capable of self-replication, would "spontaneously emerge".

We report that a modified version of Amoeba ("Amoeba-IV"), with addressing that freely accesses memory, not only exhibits emergence, but does so using a far richer pathway.

Description of the Amoeba-IV System

The current version of the Amoeba system ("Amoeba-IV") uses the same toroidal 2D memory space used in Amoeba-III (Pargellis, 2003). The opcode basis set is similar but there are some significant differences that have radically changed the self-ordering of Amoeba's memory space, the emergence of ancestral replicators, and the diversity of those replicators.

There are several major improvements. NOPs are used for addressing as was done with the Tierra/Avida systems. The self-exam process for calculating a cell's size is more complex. Eight opcodes use the Avida methodology of a default operation modified by means of a following NOP.

The most significant change is the use of NOP-addressing. It was originally thought that using NOPs would require a prohibitively long time for emergence from a soup of random opcodes since a replicator would require at least five NOPs in addition to a minimum of seven opcodes required in Amoeba-III for primitive protobiotics. However, we find that Amoeba-IV exhibits emergence by self-organizing its opcodes through several phases. First, the opcode set coalesces into a reduced set. Second, primordial "building-blocks," consisting of short opcode-sequences, are propagated throughout the memory

space. Finally, replicators emerge if the building-block density is sufficiently high.

This propagation of primordial building-blocks leading to emergence is the most significant observation of our current Amoeba research.

Opcode Basis Set

The set of opcodes used in Amoeba-IV is listed in Table 1.

No.	Opcode Abbrev.	Description
1	NOP1	Address label. Complement is "2".
2	NOP2	Address label. Complement is "1".
3	MALL	Allocate a virtual CPU for child.
4	COPY	Copy one opcode from parent to child.
5	DIVD	Initiate child CPU.
6	IFAG	If AX > BX, skip next opcode.
7	IFAL	If AX ≤ BX, skip next opcode.
8	JMPB	Jump IP back to NOP complement.
9	JMPF	Jump IP forward to NOP complement.
10	CALL	CALL address (similar to JMPF).
11	RETN	Return to opcode after CALL.
12	ADRB	Search back for address, put in EX.
13	ADRF	Search forward for address, put in EX.
14	SWEF	Switch EX value with FX value.
15	TOGS	Toggle active stack (Stk-A and Stk-B).
16	PSHA	Push AX (BX) value onto active stack.
17	POPA	Pop active stack value to AX (BX).
18	ADDA	AX = AX + BX (BX = AX + BX)
19	SUBB	BX = BX – CX + 1 (AX = AX – CX)
20	BEQA	BX = AX (AX = BX)
21	AEQZ	AX (BX) = 0
22	INCA	Increment AX (BX) register by one.
23	DECA	Decrement AX (BX) register by one.
24	INCD	Increment DX register by one.
25	DECD	Decrement DX register by one.

Table 1: Description of opcodes used in Amoeba.

Differences in the Opcode Definitions. Ideally, an opcode basis set is computationally universal, enabling a system to develop algorithms of arbitrary complexity. But, the choice of basis set is not the only criterion for a Turing machine. Another requirement is the ability to move to any location in memory. Maley showed that the Tierra system could simulate a Turing machine, although somewhat inefficiently (Maley, 1994). Amoeba-IV addresses this issue by means of the Tierra-like NOP-addressing.

The JMPB, JMPF and CALL opcodes move the pointer to the address complement of the following NOP(s). In the absence of such a NOP, the IP will jump to the address stored in the EX address register if an opcode such as ADRB or ADRF had previously loaded the EX register.

Several opcodes (PSHA, POPA, ADDA, SUBB, BEQA, AEQZ, INCA, DECA) invoke the Avida methodology. A default operation is modified if the following opcode is a NOP. The alternate action is shown in parentheses in Table 1.

Memory Space and Virtual CPUs

Amoeba-IV uses a 2D memory space with periodic boundary conditions, organized into 500 "Tierra-like" bands, each with 2399 opcodes, for a total of ~1.2×10^6 opcodes. The basis set has 25 unique opcodes. A cell's sequence is confined to a particular band. However, cells can allocate memory for their children in adjacent bands and at other start positions within those bands. We chose 2399 opcodes per band because 2399 is a prime number. This prevents cells from being able to generate an integral number of children along a band, creating a "barrier" to altering cell sizes in future generations.

There are 2000 virtual CPUs, allocated in two ways. First, at the start of a new generation, 5% of the CPUs are assigned to sequences of randomly generated opcodes. Second, a cell (opcode sequence) allocates the next CPU in the queue (using the MALL opcode) prior to copying opcodes to its child. A new generation starts when all CPUs have been allocated.

Each CPU has four numerical registers, two address registers, two stacks, and an instruction pointer (IP) that operates on opcodes. Additional parameters include the cell's size and its IP location (band and position within the band). CPUs are accessed sequentially. Each CPU is given a slice of CPU time that is proportional to its cell's size. Slices range from a minimum of 6 to a maximum of 100 operations.

Figure 1 is a snapshot of part of Amoeba's memory space (color-coded opcodes) along with a schematic of a virtual CPU. An opcode sequence in one of the bands is expanded.
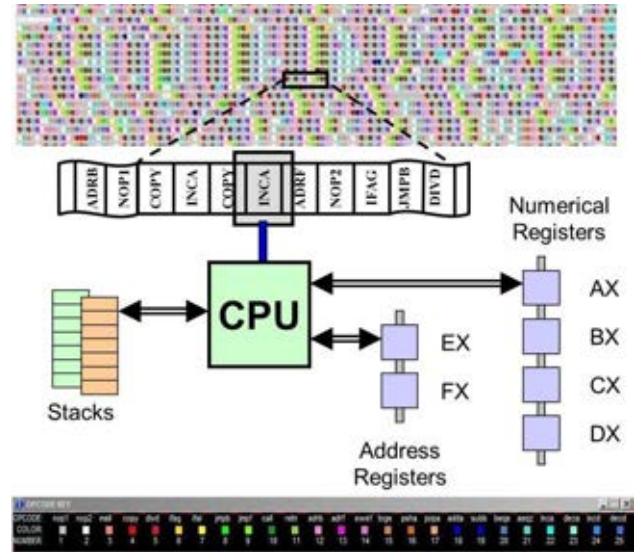
Figure 1: A small portion of the Amoeba memory space is shown at the top. Opcodes are color-coded; the color key is at the bottom.

In Figure 1, IPs move from left to right along a band unless jumped to a NOP-address along the band by means of a JMPB, JMPF, CALL, or RETN opcode.

Evolution and Mutations

Replicators evolve in three ways: opcode mutations; randomly generated sequences; and cells overwriting each other's code. Opcodes are mutated at two different times. Each time a cell copies an opcode to its child (COPY opcode) the probability that opcode is substituted by another is 0.005. Each time a cell initiates its child (DIVD opcode), the child's opcode sequence can undergo one of three types of mutations, for a total rate of 0.10. Empirically chosen mutation rates are *insertion* (0.02), *deletion* (0.02), and *substitution* (0.06). Divide mutation rates above 0.20 randomize opcodes faster than building-blocks critical for replications are propagated while rates below 0.05 overly bias the opcode basis set; NOPs dominate and some opcodes critical for replication become too infrequent.

At the start of each new generation, 100 sequences, each consisting of one, two, or three randomly selected opcodes, are randomly distributed throughout memory space. The probability of inserting a random sequence is peaked at the middle of the 500 bands. The middle bands tend to be "melted", but the edge bands have no random sequences.

There is no write-protection in Amoeba. The MALL opcode only defines the start position for a child's opcodes and assigns it a virtual CPU.

Anatomy of a Typical Self-replicator

Figure 2 shows the anatomy of a typical self-replicator.
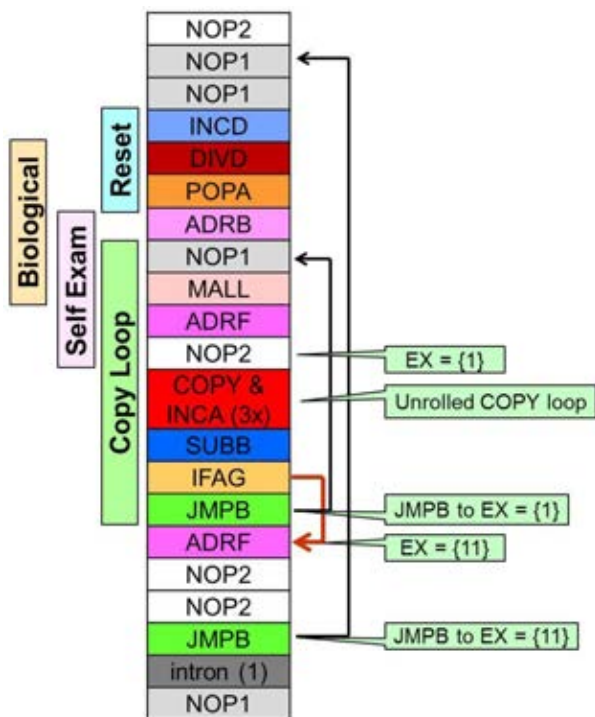
Figure 2: Anatomy of a replicator. Run: 06-08-6140. Gen: 3.499M.

The opcode color-coding is as in Figure 1. This replicator has four "replicator genes" (shown on the left-hand side): *Self-Exam* (ADRB, NOP1, ADRF, NOP2, SUBB), *Copy Loop* (NOP1, COPY, INCA, IFAG, JMPB), *Biological* where a virtual CPU is allocated and then initiated (MALL, DIVD), and *Reset* registers (INCD, POPA, JMPB). Note that the genes overlap and/or can be split into multiple pieces.

The main feature of the Amoeba-IV system is the self-organization of the initially random distribution of opcodes in memory.

Self-organization Leads to Emergence

The development and evolution of an Amoeba system from the initial random distribution of opcodes to the emergence of "ancestral self-replicators" occurs during three main stages: prebiotic, protobiotic, biotic (Pargellis, 1996a). The prebiotic phase coalesces the original set (alphabet) of 25 opcodes into a reduced, biased set of about 15 to 20 opcodes. This reduced set continues to self-organize, generating short sequences of n-opcodes ("n-ops") that propagate critical building blocks required for replication. An inefficient proto-replicator emerges, usually within a million generations, when the building-block density is sufficiently high. Mutations drive the proto-replicators to evolve into robust replicators that eliminate unneeded opcodes and unroll the copy-loop (multiple {COPY, INCA} sequences per loop).

Figure 3 shows the distribution of (binned) emergence times for a set of 25 runs out of a total of 49 runs.
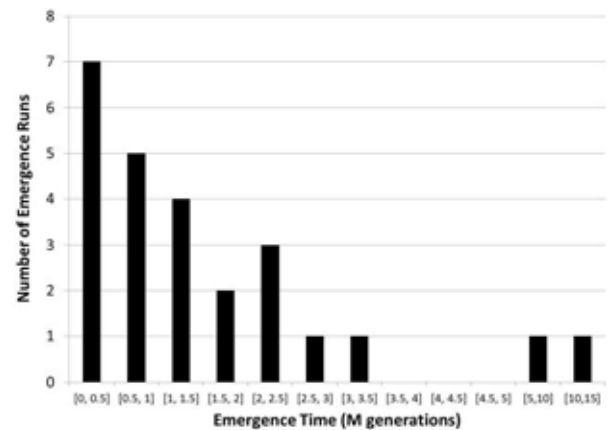
Figure 3: Distribution of self-replicator emergence times.

The opcode basis set always biases into a consolidated set of about 20 opcodes within the first 50,000 generations. Nearly half of the emergences occur within the first million generations. One cause for the steady drop in emergence probability with time is the continued biasing of the opcode basis set. Significantly, after ten million generations more than 25% of the opcodes are either NOP1 or NOP2. This reduces the frequencies of other opcodes critical for replication, including conditional opcodes (IFAL and IFAG) and branching opcodes (JMPB and JMPF).

Self-organization

The propagation of n-ops throughout memory can be visualized in the (partial) screen-shots shown in Figure 4. (A) Initial random opcode distribution, (B) prebiotic self-organization, (C) post-emergence dominated by self-replicators.
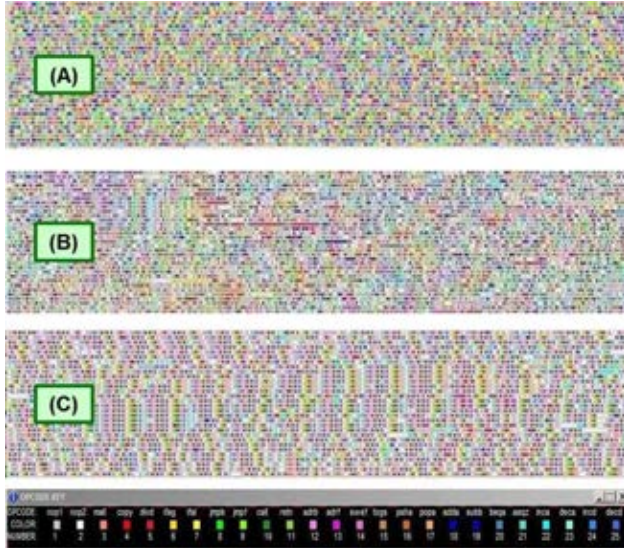
Figure 4: Self-organizing memory space in Amoeba. (A) Initial random opcodes; (B) Prebiotic propagation of short opcode sequences; (C) Post-emergence generation of self-replicators. Color-coding shown in key at bottom. Run: 07-08-2015_54654.

The initially random opcode distribution of Figure 4A becomes partially ordered during the self-organization phase of Figure 4B. Some opcodes are replicated sequentially, shown by the horizontal lines of a single color. A series of snippets are visible in the left-hand third of Figure 4B. Figure 4C shows that after emergence, the memory map consists of thousands of replicators across bands and within bands.

We can quantify the memory-map organization of Figure 4 by calculating the two-position opcode correlations within a band. Opcodes are uncorrelated at the beginning of a new run. Once self-replicators emerge, opcodes are correlated over multiples of the replicator lengths. We have calculated the correlations within bands over windows 100 opcodes wide. We do not sow those results here because the correlations do not add significantly to what we already learn from visual inspection of the memory space.

Size Distribution of Opcode Sequences. The self-organization and increasing frequencies for selected n-ops leads to a population of children with steadily growing sizes.

Figure 5 shows the growth in the sizes of children. Initially, the sizes are mostly one to three opcodes as this is the size range for the (100) randomly generated sequences placed throughout the Amoeba world at the beginning of each new generation. Longer opcode sequences become more prevalent, eventually leading to the emergence of a proto-replicator. It

should be noted that some Amoeba runs have generated large proto-replicators with sizes of several hundred opcodes.
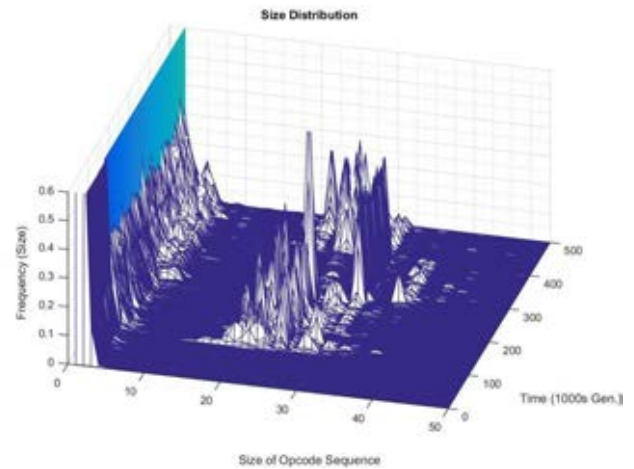
Figure 5: Evolution of sizes (number of opcodes) for children. Protobiotics emerge at about 70,000 generations and steadily evolve into a distribution of robust replicators. Run: 07-12-2015_31648.

Frequency Growth of Critical n-opcode sequences. A fundamental difference between Amoeba-IV and earlier versions is that the "ancestral" protobiotic does not suddenly emerge through some fortuitous combination of opcodes. The inclusion of NOPs for addressing in the Amoeba-IV state machine makes the probability of such an ancestor's spontaneous emergence highly improbable. To date, the smallest self-replicators contain about 15 opcodes (14 if the replicator "guesses" its size, e.g. does not have the self-exam gene). This results in $25^{14} \cong 10^{19.6}$ combinations of which a tiny (not easy to estimate) fraction are self-replicators.

Surprisingly, rather than losing self-replicator emergence, the Amoeba memory space self-organizes over a period of several hundred-thousand generations by propagating opcode sequences of ever growing length and complexity. Eventually, in about half the runs, a proto-replicator ancestor emerges that quickly evolves into a population of robust replicators.

Case Example: Self-organization to Emergence

In this section, we analyze one example for the emergence from an initial primordial "soup" of random opcodes. We first present the anatomy of the emerged replicator, followed by data on the self-organization that leads to the propagation of building blocks for the "replicator genes."

The most complicated replicator gene is the copy-loop because it includes machinery for copying opcodes from the parent to its child, a branch opcode to repeat the loop, and a conditional check that breaks out of the loop. This means a typical copy-loop consists of some version of {NOP, COPY, INCA, IFAG, JMPB}. As Figure 2 above shows, this can be complicated in cases where parts of other genes, such as the Self-Exam gene, are embedded in the Copy-Loop gene.

In this section, we choose a somewhat unusual case using a CALL-RETN copy-loop rather than the more common loop

ending with a JMPB opcode. We chose the CALL-RETN case because there are no extraneous opcodes (parts of other replicator genes) embedded in the copy-loop, despite the fact that the CALL-RETN loop has six opcodes instead of five. (Actually, the example shown here unrolled the {COPY, INCA} combination very quickly. We will show the building-block development that led to this.)

Anatomy for a "CALL-RETN" Replicator. The anatomy of a "CALL-RETN" replicator is shown in Figure 6. There are 14 unique opcodes in this replicator that are useful for replication. Irrelevant opcodes (introns) have been neglected for brevity. The CALL-RETN replication method is rare because it requires two opcodes (CALL and RETN in that order) for closing the copy-loop. Most replicators just use one opcode (JMPB) to close their copy and reset loops.
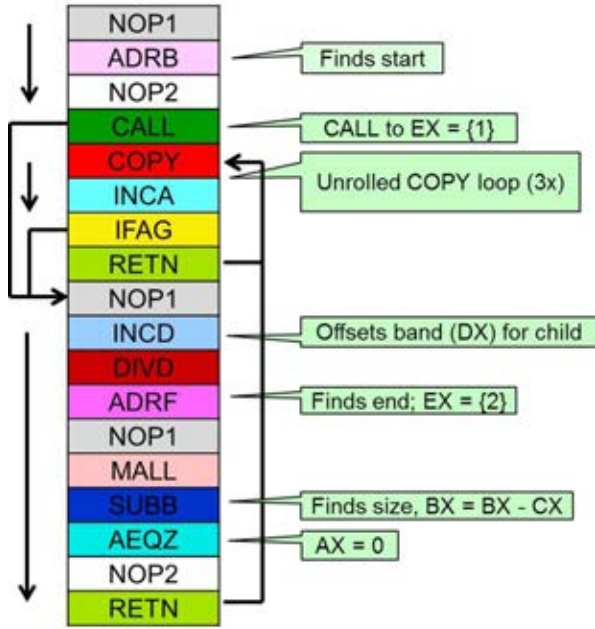
Figure 6: Anatomy of a replicator that uses the CALL/RETN combination for both the (extended) COPY loop and IP return. Run: 11-10-2015_25271. Gen: 321,982.

The CALL opcode is handled similarly to the JMPF opcode; CALL first looks for one or more NOPs immediately following. In the absence of subsequent NOPs (as in this example), CALL will check its EX-register for any loaded address and then jump to that address. In this case, the ADRB-NOP2 combination has loaded the EX-register with the complementary address, "1" (NOP1). The RETN is used both for the COPY loop and resetting the IP after the DIVD. (Amoeba ignores the DIVD operation unless the parent had first allocated a CPU for its child by means of the MALL opcode). The ADRB is replaced with ADRF in only 2000 generations. This is frequently seen in Amoeba as a parent automatically puts the beginning of its child sequence in the child's CX register. The only advantage of retaining the

ADRB opcode is to capture "rogue IPs", enabling the host to commandeer the virtual CPUs associated with other cells in the same band.

Growing Single-Opcode Frequencies. Initially, all 25 possible opcodes are equally distributed and the frequency (fraction, $f(m_j)$, of all 1.2 million opcodes) is the same; $f(O_j) = 0.040$ for all O_j. However, the frequencies for some opcodes useful to propagating sequences of opcodes preferentially grow at the expense of other opcodes.

Figure 7 shows the increase in frequency over time (left-hand scale) for selected single opcodes (1-ops). Emergence occurred at about 292,000 generations (labeled, vertical line).

On the right-hand scale in Figure 7, we plot the "monomeric opcode entropy", $S(O)$, as a quick check into the status of self-organization of single opcodes and given by,

$$S(O) = -\sum_{j=1}^{D} \frac{m_j}{M} \ln\left(\frac{m_j}{M}\right). \qquad (Eq.\ 1)$$

where $D = 25$ is the size of the alphabet (number of unique opcodes in the basis set), $M \cong 1.2 \times 10^6$ is the size of the memory space (total number of opcodes), m_j is the number of occurrences (counts) for the jth opcode, given by the symbol, O_j, and ln is the natural log. An estimate of the effective size of the opcode basis set is $E(O) = exp(S)$. For the initial, equally distributed opcodes, $S(O) = 3.218$ and $E(O) = 25$. By 500,000 generations, the entropy drops to $S(O) = 2.935$, indicating the useful part of the basis set shrinks to about 19 opcodes. Most runs evolve a basis set of 15 to 16 opcodes.
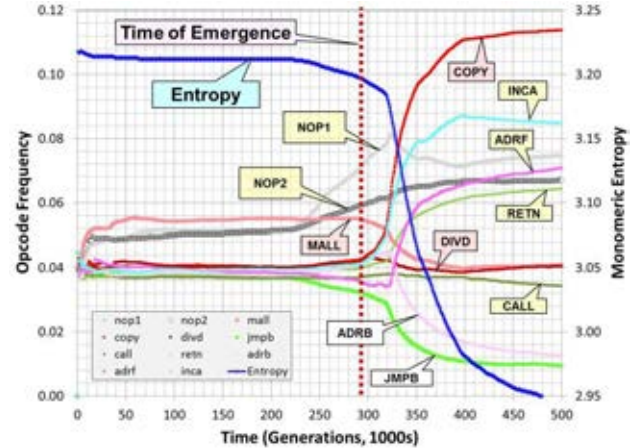
Figure 7: Growth of 1-ops and commensurate decrease in entropy for the CALL-RETN replicator of Figure 6. Run: 11-10-2015_25271.

Initially, 1-ops conducive to replication grow in frequency: NOP1, NOP2, MALL. The NOP1 and NOP2 frequencies begin to increase about 70,000 generations before emergence. The NOPs are arguments for the CALL and ADRB opcodes.

The ADRB opcode is not required in order for a cell to calculate its size and load that value into its BX-register. Amoeba automatically loads into a child's beginning location

in the band into the CX-register when a parent first allocates memory (using the MALL opcode) for its child. However, ADRB is useful for resetting the CX registers of rogue IPs, essentially hijacking other virtual CPUs for the host. Once the replicator population becomes dominant, the ADRB is no longer useful. In this case example, the ADRB opcode mutated into ADRF which also loaded the EX-register with the NOP1 address (used by CALL). Consequently, $f(ADRF)$ doubles and $f(ADRB)$ drops.

After emergence, COPY and INCA become more prevalent when the copy-loop is "unrolled" and the combination {COPY, INCA} is repeated.

Growing Multi-Opcode Frequencies. The co-occurrence of two opcodes, and therefore the degree of opcode self-organization, can be quantified using the mutual information (*MI*) measure (Manning, 2000),

$$ MI = \sum_{\{O_i,O_j\}} p(O_i,O_j)\ln\left[\frac{p(O_i,O_j)}{p(O_i)p(O_j)}\right]. \quad (Eq.\ 2) $$

where the sum is over all $25^2 = 625$ possible opcode pairs for our basis set of 25 opcodes. The argument in the sum is the pointwise mutual information, $MI(O_i,O_j)$, which is the log of the odds-ratio of opcode pairs and is zero in the absence of any correlation.

We plot *MI* versus time with the dashed blue line in Figure 8 (right-hand scale). The MI shows clearly that self-organization has occurred. Critically, what we previously noted as the time of emergence coincides with the maximum increase in the mutual information. Scientifically, this is a key finding. We demonstrate the degree of ordering for the joint probability versus the uncorrelated single probabilities for a select set of opcode pairs (2-ops) by plotting the time evolution of the "odds-ratio", the argument of the natural logarithm in *(Eq. 2)*, in Figure 8. We used the CMU toolkit for counting n-ops (Clarkson, 1997).
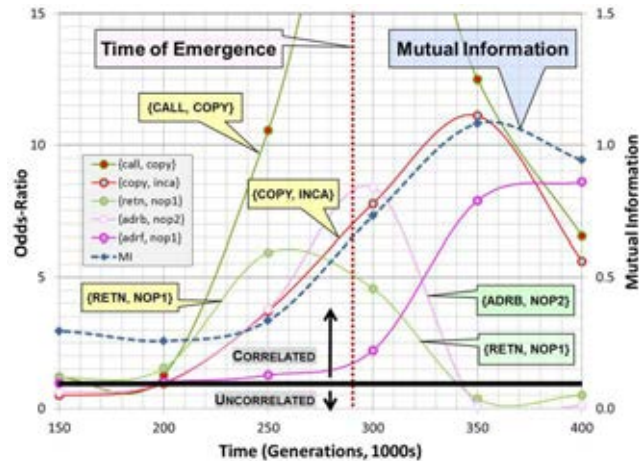
Figure 8: MI (RH) and Odds-ratios (LH) for 2-ops that lead to the COPY-RETN building block for the replicator of Figure 6.

A key observation is that 2-ops critical to development of the CALL-RETN loop, {CALL, COPY, INCA, RETN}, grow in abundance at least 50,000 generations *before* emergence during the self-organization period. The ADRB opcode gets replaced by the ADRF opcode once it is no longer useful after emergence and the {RETN, NOP1} drops after emergence because introns are inserted between the RETN and NOP1.

Development of the Copy-Loop Building Block. The copy-loop building block is noteworthy since the primordial COPY-RETN loop, {CALL, COPY, INCA, RETN}, is clearly a prebiotic building-block; it is impossible for an IP to break out of this loop!

This sequence copies opcodes throughout memory, but without a conditional check (IFAG or IFAL) the IP can never break the loop and initiate a child. Nevertheless, this propagator is a useful building block, capable of propagating itself and other opcodes. An insertion mutation could have modified this building block prior to emergence by inserting an IFAG conditional check before the RETN opcode.

Figure 9 is a schematic showing how the copy-loop is built from the CALL-RETN building-block.
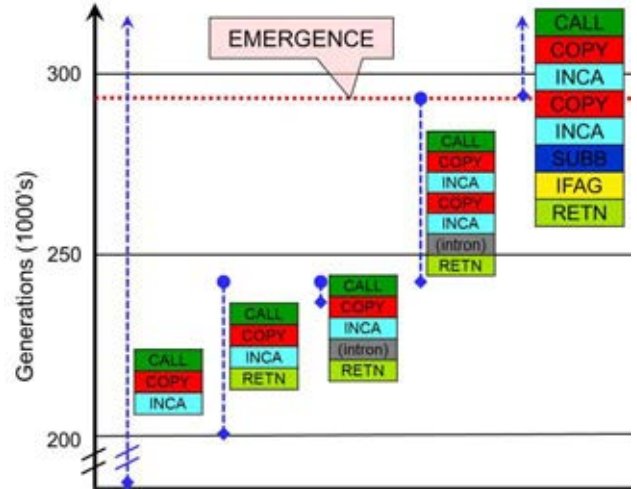
Figure 9: Timeline for the copy-loop building block.

The CALL opcode calls the NOP1 address (see Figure 6). Usually, the CALL opcode is followed by a NOP and the call is to that NOP's complement. In the absence of a subsequent NOP, the CALL looks into its address register, EX, to see if it has an address. In this run, the previous ADRB opcode loaded the EX-register with NOP1, the complement to NOP2. Note: the {ADRB, NOP2} combination was many opcodes prior to the CALL-RETN loop in prebiotic propagator sequences. There was also a subsequent NOP1 in the primordial form of the CALL-RETN loop. An intron preceded the NOP1 during pre-emergence.

The vertical dashed blue lines show the time range over which each of the sequences exist. The solid blue diamonds are when we first saw the sequence in either a log-file or a memory snapshot. Times of extinction are shown by solid blue circles. Once a sequence occurs, it is propagated for

65

many generations until replaced by a more viable alternative. For example, the {CALL, COPY, INCA, RETN} sequence persists until the copy-loop is unrolled (generation 239,000).

Emergence of Ancestral Replicators

A consequence of the openness of the Amoeba-IV system is the diversity of ancestral replicators that emerge from the self-organizing, initially random "soup" of opcodes.

It is a challenge to identify the emergence of an "ancestor" in Amoeba-IV. Ancestors are logged in a file when a parent has faithfully copied itself to children for at least four generations. However, the early "protobiotic" replicators do not copy themselves with any fidelity. It typically takes several thousand generations before a "faithful" replicator is generated and logged. During this time, millions of cells have been initiated and some small subset of that number will eventually lead to an ancestor that faithfully copies its opcodes to its children. It is possible that an ancestor emerges out of a hypercycle of interacting components (Eigen, 1971) but this type of interaction is nontrivial to track. One can examine the "world snapshots" that are periodically saved, but each snapshot is a list of the entire Amoeba memory; the analysis of any interactions within that map is difficult.

The next examples show the rich diversity of replicators that is one of the novel outcomes of the modified Amoeba-IV artificial chemistry.

"Size-guesser" Protobiotic. An example of a class of robust proto-replicator that has emerged in several runs is shown in Figure 10. This is an inefficient replicator; it does not include the "self-exam" gene so it "guesses it size" and cannot generate a complete copy of the parent until the 3rd child.
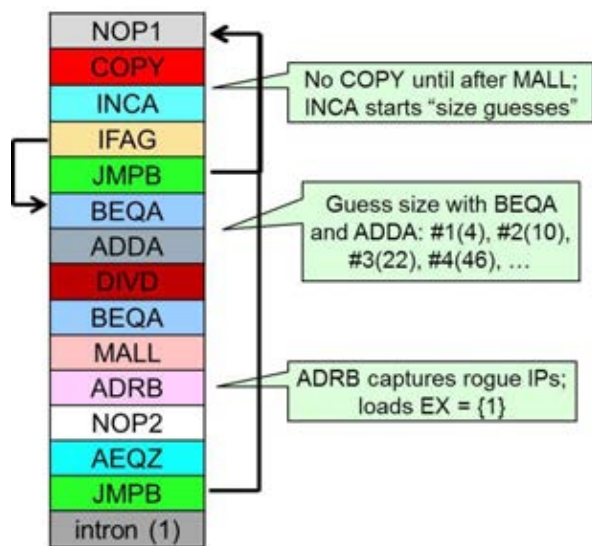
Figure 10: Anatomy of a replicator that guesses its size; the self-exam gene is missing. Run: 06-17-2015_54809. Gen: 1.042M.

About 200,000 generations later, this proto-replicator evolved into a robust replicator that used ADRF and SUBB opcodes to properly calculate its size.

"Conditional Ladder" Replicator. The anatomy of a "conditional ladder" replicator is shown in Figure 11.
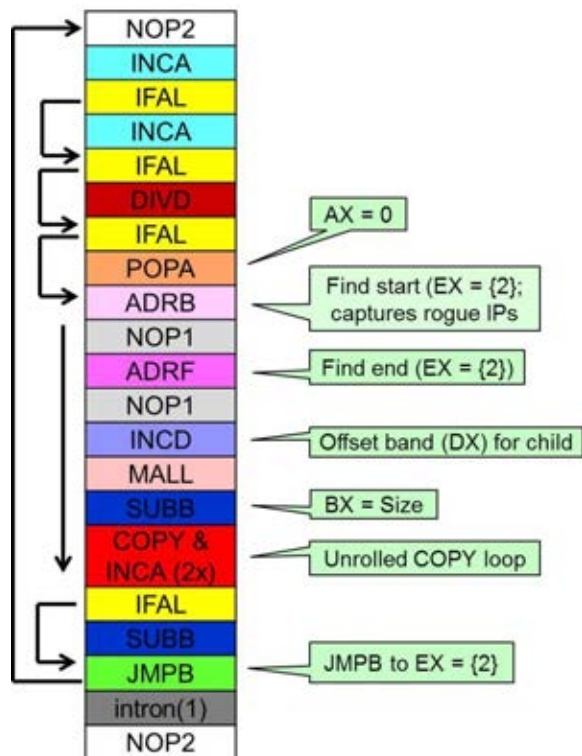
Figure 11: Anatomy of a replicator that uses a "conditional ladder" of IFAL opcodes to avoid resetting registers (DIVD, POPA, SUBB). Run: 06-20-2015_31615. Gen: 4.315M.

This replicator's COPY loop is inefficient because it includes the entire cell's opcode sequence. An "IFAL-ladder" is used to avoid prematurely resetting the cell's registers while copying its code to a child. When done copying, the replicator's 4th IFAL check fails and the BX register is set by means of the 2nd SUBB opcode. The IFAL ladder checks then "fail" and the replicator initiates its child (DIVD) and resets its AX register (POPA).

Discussion and Future Research

There are several other interesting observations in addition to the success in generating self-replicators due to opcode self-organization from a primordial soup consisting of Tierra-like opcodes and addresses.

Many proto-replicators lack some of the replicator genes shown above in Figure 2. An example of a proto-replicator without the self-exam gene was shown in Figure 10. Other replicators do not retain their IP. This was observed in earlier versions of Amoeba where cells are members of a colony. The

colony members lose their IP after generating a single child. Their IPs then execute opcodes "belonging to" other cells further down their respective Tierra-band.

We observe runs where replicators cheat and get a larger CPU time-slice. This issue arises because the time-slice is proportional to a cell's size. Earlier versions of Amoeba automatically incremented the AX-register as part of the COPY opcode; the size of a cell was the number of opcodes copied. Amoeba-IV now requires the INCA opcode to increment the AX-register. This raises an interesting question: how to determine the size of a child when the DIVD opcode initiates it? We cannot increment the size every time a COPY is used since prebiotics can copy the same opcode many times (they lack the INCA opcode). Currently, we use the AX-register value to determine the size. The "cheaters" copy their replicator sequence, typically about 30 opcodes. Then, prior to the DIVD opcode, they use multiple ADDA opcodes to increase the AX-value to many times the cell's size. We are still investigating how to prevent this parasitic behavior without decreasing replicator genome diversity.

The Amoeba systems have always been dominated by variants of a single species after emergence. We have never observed two radically different species co-existing. Viruses occur, but are quickly eliminated when the host mutate. For example, the retention of the ADRB opcode, even though not required for a self-exam, effectively "highjacks" viral IPs and their associated CPUs. We believe a variation in the externally imposed fitness landscape may enable alternative species, as well as parasites and hosts, to co-exist for extended times. Amoeba-IV partially addresses this by varying the rate at which random sequences are distributed throughout the bands. However, replicators are still able to quickly scatter their children throughout all bands. There are several options to consider: slowing down the rate at which children are spread throughout the bands, imposing write-protection when a cell allocates memory for a child (similar to Tierra), or modifying the time-slice parameters for different regions in memory so that replicator sequences of different sizes would find some parts of memory inhospitable.

Only half (25 out of 49) of the Amoeba runs exhibit emergence (see Figure 3). We observe that the probability of a replicator emerging after about one million generations is low. This implies that there are some self-organization processes that can lower the probability of emergence. One observation is that the NOP1 and NOP2 opcodes steadily become more prevalent in runs without emergence. This reduces the likelihood of creating building-block propagators because other opcodes critical to replication occur less frequently.

The earlier Amoeba versions I and II demonstrated that the probability of a randomly generated opcode sequence being a replicator increased with size (number of opcodes). It has been argued that computation-universal chemistries such as Avida or Tierra would exhibit a probability that *decreased* with increasing size (Adami, 1998). Preliminary observations with Amoeba-IV indicate this may not be the case – ancestral replicators typically include 200 to 300 opcodes – and this aspect of our findings warrants further exploration. While ancestors in Amoeba-IV are generated from building-blocks that are propagated during the self-organization phase, these "genes" are generated multiple times in huge ungainly

"protobiotics" with sizes ranging up to the maximum currently allowed (450 opcodes per cell).

Another topic of interest is the issue of "propagators," "protobiotics," and "replicators." Amoeba-IV has shown it is very difficult to identify precisely when an ancestral replicator "emerges." It may be that groups of propagators form hypercycles that eventually lead to robust self-replicators (Eigen, 1971; Eigen, 1981; Eriksson, 2006). We are currently investigating this by logging sequences that have been generated at least 50 times, regardless of whether or not the sequences are the same as their parental sequence.

Acknowledgements

References

Adami, C. (1995). On Modeling Life. *Artificial Life*, 1:429-438.

Adami, C. (1998). *Introduction to Artificial Life*. Springer-Verlag, New York, NY.

Adami, C., Ofria, C., and Collier, T. C. (2000). Evolution of biological complexity. *Proc. Natl. Acad. Sci.*, 97(9):4463-4468.

Clarkson, P. R. and Rosenfeld, R. (1997). Statistical language modeling using the CMU-Cambridge Toolkit. In Kokkinakis, G. et al., editors, *Fifth European Conf. on Speech Comm. and Tech.*, pages 2707-2710. ISCA archive, W. Hess (ed.), Bonn, Germany.

Eigen, M. (1971). Selforganization of Matter and the Evolution of Biological Macromolecules. *Die Naturwissenschaften*, 58:465-532.

Eigen, M. et al. (1981). The Origin of Genetic Information. *Scientific American*, 244(4):88-118.

Eriksson, A. et al. (2006). Quasi-Species and Aggregate Dynamics. In Rocha, L. M., et al., editors, *Artificial Life X*, pages 145-151. MIT Press, Cambridge, MA.

Maley, C. C. (1994). The Computational Completeness of Ray's Tierran Assembly Language. In Langton, C. G., editor, *Artificial Life III*, pages 503–514. Addison-Wesley.

Manning, C.D. and Schutze, H. (2000). *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, MA.

McMullin, B. (2012). Architecture for Self-reproduction: Abstractions, Realisations and a Research Program. In Adami, C., et al., editors, *Artificial Life 13*, pages 83-90. MIT Press, Cambridge, MA.

Pargellis, A. (1996a). The spontaneous generation of digital "Life". *Physica D*, 91:86-96.

Pargellis, A. (1996b). The evolution of self-replicating computer organisms. *Physica D*, 98:111-127.

Pargellis, A. N. (2001). Digital Life Behavior in the Amoeba World. *Artificial Life*, 7(1):63-75.

Pargellis, A. (2003). Self-organizing Genetic Codes and the Emergence of Digital Life. *Complexity*, 8(4):69-78.

Ray, T. (1992). An approach to the synthesis of life. In Langton, C. G. et al., editors, *Artificial Life II*, pages 371-408. Addison-Wesley.

Ray, T. (1994). An evolutionary approach to synthetic biology. *Artificial Life*, 1:179-209.

Suzuki, H. (2011). Ch. 3. Artificial Chemistry and Molecular Network. In Sawai, H., editor, *Biological Functions for Information and Communication Technologies*, Studies in Computational Intelligence, v. 320, pages 87-161. Springer-Verlag, Berlin.

Tangen, U. (2010). The Emergence of Replication in a Digital Evolution System using a Secondary Structure Approach. In Fellermann, H., et al., editors, *Artificial Life XII*, pages 168-175. MIT Press, Cambridge, MA.

Exploring Constraint: Simulating Self-Organization and Autogenesis in the Autogenic Automaton

Stefan Leijnen[1], Tom Heskes[2] and Terrence W. Deacon[3]

[1]Amsterdam University of Applied Sciences, Amsterdam, Netherlands.
[2]Radboud University Nijmegen, Nijmegen, Netherlands.
[3]University of California, Berkeley, CA, USA.
s.leijnen@hva.nl

Abstract

Many origin of life theories argue that molecular self-organization explains the spontaneous emergence of structural and dynamical constraints. However, the preservation of these constraints over time is not well-explained because of the self-undermining and self-limiting nature of these same processes. A process called autogenesis has been proposed in which a synergetic coupling between self-organized processes preserves the constraints thereby accumulated. This paper presents a computer simulation of this process (the Autogenic Automaton) and compares its behavior to the same self-organizing processes when uncoupled. We demonstrate that this coupling produces a second-order constraint that can both resist dissipation and become replicated in new substrates over time.

Introduction

Life's ability to resist degradation and persist in hostile environments is both ubiquitous and astonishing. Generation of structure, preservation by repair, and trait persistence through reproduction are perpetually organized in a continuous struggle against the destabilizing mechanisms of the second law of thermodynamics. Despite their often pivotal role in explaining the emergence of life, self-organizing processes are limited in their capacity to maintain structure (Prigogine and Stengers, 1984; King, 1982). Autogenesis is a recently proposed theory that suggests that, beyond mere self-organization, a synergetic coupling between self-organizing processes is a minimal requirement for life (Deacon, 2012). Through this higher-order linkage, the processes that generate structure may persistently recreate a capacity for self-creation, leading to robustness and a potential capacity for long-term sustenance and natural selection. An instance of such an autogenic process, a proto-life model called *autogen*, shows how two self-organizing processes – reciprocal catalysis and self-assembly – maintain each other's boundary conditions and thereby mutually increase their probability of persistence over time (Deacon, 2006).

Currently, the autogen model is a theoretical proposition that remains to be validated experimentally. This paper describes a series of simulation experiments that investigate the self-organizing and self-preserving properties of the autogen model. A simplified particle system simulation called the Autogenic Automaton models the synergetic linkage of self-organizing processes that leads to the emergence of autogens.

Second-Order Self-Organization

The nonlinear amplification that is typical for self-organization tends to push the thermodynamic conditions for further propagation toward the unfavorable (Haken, 2006). This may occur up to a point where the system is no longer far-from-equilibrium and the local thermodynamic entropy increase comes to a halt. For example, in a reciprocally catalytic set reaction rates may increase exponentially as more and more catalysts are produced, up until the point when not enough reactants are available for further propagation and the self-organizing process ends (Plasson et al., 2011). Alternatively, self-organization may break down due to unfavorable changes in external conditions. Considering the universal presence of self-organization in living systems, how can it be possible that organization persists long enough for complex organisms to come about?

When the product of an autocatalytic reaction enables a second autocatalytic reaction, which produces a reactant that enables the first (or a third, etc., as long as the causal chain is eventually closed), a so-called *hypercycle* emerges (Eigen and Schuster, 1979). Hypercycles represent one possible way in which self-organizing processes, autocatalytic cycles in this case, may be linked together in a dynamical process hierarchy. However, with respect to preventing dissipation this kind of second-order self-organization does not provide a sufficient solution: every autocatalytic cycle that the hypercycle consists of represents a potential weakest link, which may cause the fragile hypercycle to break down entirely when the reactants or energy necessary for the cycle are no longer available. Conversely, autogenesis suggests another type of second-order self-organization where two or more self-organizing processes not only promote each other, but also act as a supportive environment if one of them breaks down in such a way that their self-

undermining tendencies are reciprocally counteracted.

Autogenesis

The formation of crystals through self-assembly is a self-organizing process as the probability of particle detachment decreases with the number of adjacent particles that keep it in place. This asymmetric process causes particles to cluster together, creating a spatial difference in particle locations. This difference, maintained by the probabilities of attachment and detachment, may be viewed as a constraint on the spatial distribution of particles. More generally, this reduction of variety of macroscopic states can be understood as a constraint producing process.

In a reciprocally catalytic system, each reaction initially leads to an increased probability for another reaction to take place as more and more catalysts are created. Exponential growth ensues until the reactants are depleted. Reciprocal catalysis leads to exponential increase of reactions that is limited solely by the number of available reactants.
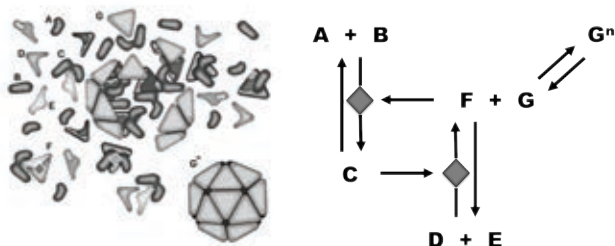
Figure 1: Illustration of autogen formation (left) and the reactions involved (right). Autogens constitute a dynamical linkage between self-assembly and reciprocal catalysis. In this model system, self-assembly is a self-organizing process where G particles attach to one another, forming G^n crystals of size n. These crystals may break up due to detachment. G particles are generated by a reciprocally catalytic set of six different particle types (A to F). In turn, crystals may contain C and F particles, isolating these catalysts from potential reactants and thereby ensuring a potential for G-particle production over time.

A boundary or container would prevent the exhaustion of reactants by shielding them from the environment, thereby preserving a chemical potential for further dissipation (Maturana and Varela, 1980; Rosen, 1991). Such a container may itself be formed by a self-organizing process, e.g. crystal growth through self-assembly (Fellermann et al., 2007). The autogen model (fig. 1) goes further in suggesting that the form and function of a self-assembled container is dynamically linked to the catalytic process as it prevents the reactants from being depleted; similarly, it explains how the catalytic process dynamically shapes the form and function of the crystals as it affects the process of self-assembly.

Following the type of second-order self-organization described above as autogenesis, constraint preservation is enabled by a juxtaposition of constraint producing processes, such that they actively support each other's persistence. Whereas self-promoting self-organizing processes such as hypercycles tend toward self-undermining and ultimately a breakdown of the causal cycle, this reciprocally counteracting juxtaposition actively prevents self-undermining from taking place. As the autogen is able to do work on its own conditions for sustenance, it grows independent from the conditions of its environment and becomes more dependent on its internalized constraint. When the autogen is damaged, it likely begins to repair itself; under some conditions the probability of growth and sustainment may become higher than that of breakdown.

The relative stability of these structural synergies allows for a simple type of natural selection to occur, as some will be better suited to prevailing conditions than others and therefore have a better chance of sustaining themselves. This eventually leads to a higher-level reduction of variety, as unsuccessful variety-reducing synergies are removed.

The Autogenic Automaton

The processes that generate and preserve constraint and which are necessary for autogen formation are simulated in a program called the Autogenic Automaton. This simulation does not provide a physically accurate model, but is merely aimed at demonstrating the viability of the proposed constraint hierarchy (McMullin and Varela, 1997; Varela et al., 1974). A two-dimensional tile grid is used as a discrete model of a closed reaction-diffusion system (fig. 2). Six particle types are used for modeling reciprocal catalysis while a seventh type models the formation of crystals through self-assembly, following figure 1. Particle movement and the reaction rules that govern particle attachment, detachment, and the creation and removal of particles are all computed locally per tile. As such, the system resembles a cellular automaton (Bilotta and Pantano, 2005) where multiple particles may occupy one tile. Particle movement and diffusion caused by particle-to-particle collisions is approximated by random movement to a horizontally or vertically neighboring tile with a probability of 0.1 at every time step for each particle. With respect to movement, crystals are treated as single particles together with any particles they encapsulate.

The simulation is initialized by distributing predefined quantities of particle types randomly on the grid. Next, particle positions and reactions are computed per tile. Modeling only localized reactions ensures that only small subsets of the total number of particles interact at each time step, reducing the computational complexity of the simulation. Furthermore, only the aspects of self-organization that are necessary for demonstrating the viability of autogenesis are modeled. Physical properties such as kinetic energy, heat dissipation, and crystal geometry are not simulated. Due
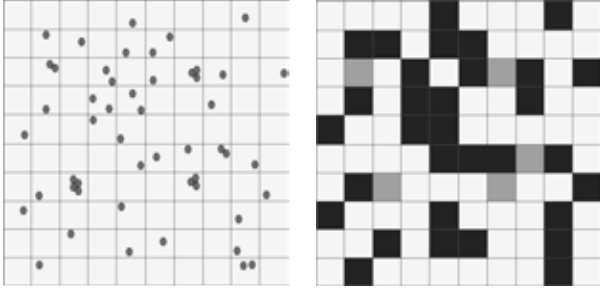
Figure 2: A closed continuous particle system (left) is modeled approximately in the Autogenic Automaton as a non-toroidal discrete grid of 10 x 10 tiles (right).

to this absence of heat and friction, the entropy potential necessary for far-from-equilibrium systems to become self-organizing is defined with respect to chemical equilibrium rather than thermodynamic equilibrium. The simulation is initialized with chemical non-equilibrium conditions, so before any crystallization has occurred or catalytic reactions have taken place.

Quantifying Constraint

Through the course of a simulation run, the system moves through various macroscopic states caused by processes that generate, preserve and select constraint. Constraint can be expressed by means of the information entropy over the spatial probability distribution of particles or reactions in the tile grid (Kauffman, 1993; Harder and Polani, 2013; Polani, 2008). Notable characteristics of the particle system surface by observing the (in-)homogeneity of locations that correspond to these characteristics. So, the quantification of statistical entropy described below yields an indirect observation of the underlying processes.

Given a set of probabilities p_i for $i = 1, \ldots, n_{\text{tiles}}$ with $0 \leq p_i \leq 1$ and $\sum_i p_i = 1$, the information entropy is defined as (Shannon, 1948; Cover and Thomas, 1991):

$$H = -\sum_{i=1}^{n_{\text{tiles}}} p_i \log_2 p_i .$$

When we consider an event (i.e. particle occurrence or reaction) X, we substitute

$$p_i = \frac{|X_i|}{|X|} ,$$

with $|X_i|$ the number of events at tile i and $|X| = \sum_i |X_i|$ the total number of events, to obtain

$$H(X) = -\sum_{i=1}^{n_{\text{tiles}}} \frac{|X_i|}{|X|} \log_2 \frac{|X_i|}{|X|} .$$

For ease of interpretation we often consider the so-called normalized information entropy (Jost, 2006)

$$\hat{H}(X) = -\frac{1}{\log_2 n_{\text{tiles}}} \sum_{i=1}^{n_{\text{tiles}}} \frac{|X_i|}{|X|} \log_2 \frac{|X_i|}{|X|} , \qquad (1)$$

which normalizes the standard information entropy by its maximum value such that always $0 \leq \hat{H}(X) \leq 1$. For a completely homogeneous distribution of events over tiles we get $\hat{H}(X) = 1$, whereas $\hat{H}(X) = 0$ when all events X are concentrated at a single tile.

Where it is necessary to measure the spatial difference between two event types, X and Y, the Kullback-Leibler divergence of their distributions over the grid is used (Kullback and Leibler, 1951):

$$D_{KL}(P||Q) = \sum_{i=1}^{n_{\text{tiles}}} p_i \log_2 \frac{p_i}{q_i} .$$

Substituting p_i and q_i with $\frac{|X_i|}{|X|}$ and $\frac{|Y_i|}{|Y|}$, respectively, would yield infinite divergence for a distribution with a tile i such that $|X_i| > 0$ and $|Y_i| = 0$. To resolve this problem, a smoothing function is used (Bigi, 2003), where

$$q_i = \begin{cases} \alpha \frac{|Y_i|}{|Y|} & \text{for} \quad |Y_i| > 0 \\ \epsilon & \text{for} \quad |Y_i| = 0 \end{cases} , \qquad (2)$$

with $\epsilon = 10^{-5}$ and normalization coefficient α chosen such that the probabilities sum to 1. p_i is substituted similarly with $\frac{|X_i|}{|X|}$. For ease of exposition, we will omit this smoothing in subsequent formulas.

Constraint Generation

The Autogenic Automaton is used here to simulate the generation of constraint in the formation of crystals through self-assembly, and in the locally nonlinear reactions taking place in a reciprocally catalytic set. In later sections, these two processes will be combined to simulate autogenesis.

Self-Assembly

Self-assembly is modeled in a simplified way, by defining attachment and detachment reactions between G particles and crystals G^n:

$$G^n + G \rightleftarrows G^{n+1} , \qquad (3)$$
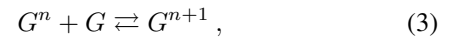
with $n \geq 1$ and $G^1 \equiv G$. At every time step, when a G particle is located within the same tile as either another G particle or crystal, the probability of attachment is given by reaction parameter γ^+:

$$P_g^+ = \gamma^+ \in [0, 1] .$$

Once formed, G particles have a probability P_g^- of detaching from the crystal again. Larger crystals are more tightly connected and less likely to break apart than smaller crystals due to a larger number of kinks holding the individual particles together (Burton and Cabrera, 1949). An increased size yields a lower probability of detachment and therefore increases the probability for further growth. This introduces asymmetry in the crystal growth process, reflected in our model system by a detachment probability function that is negatively exponential to the crystal size n with reaction parameter γ^-:

$$P_g^- = (1 + \exp[\gamma^-])^{-n} \, ,$$

with $\gamma^- \in \mathbb{R}$, $n \geq 2$. Following equation (1), event X_i is defined with respect to self-assembly as the observation of a G particle at tile i, where G^n crystals are counted as n observations. Therefore, the generation of constraint during this process is examined using

$$\hat{H}(G,t) = -\frac{1}{\log_2 n_{\text{tiles}}} \sum_{i=1}^{n_{\text{tiles}}} \frac{|G_i(t)|}{|G(t)|} \, \log_2 \, \frac{|G_i(t)|}{|G(t)|} \, ,$$

for the normalized information entropy of G at time t.
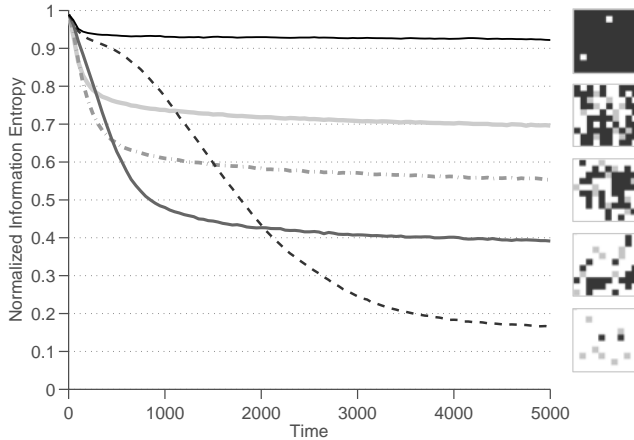
Figure 3: Decrease of normalized information entropy $\hat{H}(G,t)$ during self-assembly of 1000 G particles with $\gamma^+ = 1$, for several detachment probabilities: ——— $\gamma^- = -5$, ••••••• $\gamma^- = -4$, ——— $\gamma^- = -3$, •••••• $\gamma^- = -2$, ——— $\gamma^- = -1$. The images on the right depict how $\hat{H}(G,t)$ correlates with the distribution of G particles over the grid, ranging from an almost homogeneous distribution (top) to a few G^n crystals (bottom). Results are averaged over 100 trial runs.

Figure 3 shows the development of $\hat{H}(G,t)$ over time, for different values of γ^-. With $\gamma^- = -5$ the probability of detachment P_g^- is relatively high, leading many crystals to fall apart and the constraint on particle G locations being low.

With $\gamma^- \in [-1, -2]$, P_g^- is relatively low: once formed, crystals only break apart sporadically, leaving hardly any G particles available for attachment and potential for further growth. At $t = 5000$, the G particle locations are maximally constrained for $\gamma^- = -4$.

Reciprocal catalysis

Particle types A to F are used to model self-organization through reciprocal catalysis. Particles of type A and type B may react to form a C particle when both are located in the same tile; similarly for particles D and E forming F:

$$A + B \overset{F}{\rightleftharpoons} C \, , \tag{4}$$

$$D + E \overset{C}{\rightleftharpoons} F \, . \tag{5}$$

Particles F and C are catalysts for the left-to-right reactions of (eq. 4) and (eq. 5), respectively. In order to accommodate for the system dynamics required in later sections, reaction probability P_r^+ decreases exponentially with $1/n$ where n is the number of catalysts present at the same tile i, i.e. $n = |F_i|$ for (4) and $n = |C_i|$ for (5):

$$P_r^+ = (1 + \exp[\varrho^+])^{-(1+n)^{-2}} \, ,$$

with $\varrho^+ \in \mathbb{R}$. The right-to-left reactions (i.e. C splitting into A and B and F into D and E) occurs with probability P_r^- for each C and F particle at every time step:

$$P_r^- = \varrho^- \in [0, 1] \, .$$

Similar to self-assembly, reciprocal catalysis is a locally nonlinear process: one catalytic reaction increases the likelihood of another catalytic reaction occurring. However, the observable artifacts of reciprocal catalysis (i.e. the produced catalysts) may cease to exist once this amplification process no longer takes place, or they may diffuse to different locations. To quantify the generated constraint, we therefore use the probability distribution of reaction locations as observed events, rather than the locations of catalysts themselves. The constraint generated by reciprocal catalysis is quantified by a decrease in normalized information entropy. Following (eq. 1) event X_i is defined as the observation of catalytic reaction R at tile i:

$$\hat{H}(R,t) = -\frac{1}{\log_2 n_{\text{tiles}}} \sum_{i=1}^{n_{\text{tiles}}} \frac{|R_i(t)|}{|R(t)|} \, \log_2 \, \frac{|R_i(t)|}{|R(t)|} \, .$$

In order to investigate the effect of parameters ϱ^+ and ϱ^- on the normalized information entropy, $\hat{H}(R,t)$ is averaged over time:

$$\frac{1}{t_{\max}} \sum_{t=1}^{t_{\max}} \hat{H}(R,t) \, .$$
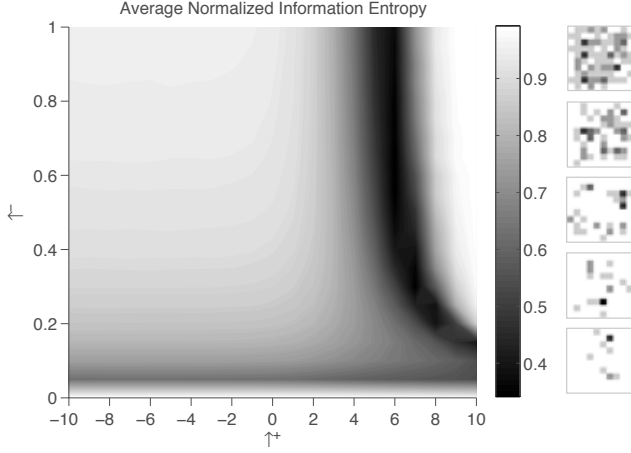
Figure 4: Normalized information entropy during reciprocal catalysis after initialization with 1000 particles distributed equally among types A, B, D, and E. For given ϱ^+ and ϱ^-, $\hat{H}(R,t)$ is averaged over 5000 time steps and 10 trial runs. The right-side images depict the distribution of catalytic reactions over the grid.

Results for $t_{\max} = 5000$ are shown in figure 4. It is found that distribution R is maximally constrained for $\varrho^+ \approx 6$ and $\varrho^- > 0.5$ (i.e. when catalysts break up regularly).

Constraint Preservation

Under particular extrinsic conditions (i.e. γ^+, γ^-, ϱ^+ and ϱ^-) self-assembly and reciprocal catalysis generate constraint spontaneously. If these conditions are subject to unfavorable changes, the constraints produced will also be eliminated spontaneously. Here, we consider how a higher-order linkage between these processes may preserve constraint by preventing spontaneous dissipation in unfavorable conditions.

The autogenic process consists of a mutually constraining coupling. G particles generated by reciprocal catalysis are created in close proximity to one another due to the locality of catalytic amplifications, thereby increasing the likelihood of crystal growth; at the same time G^n crystals preserve a potential for reciprocal catalysis, by encapsulating catalysts and thereby preventing exhaustive catalysis.

Modeling this synergetic linkage, the left-to-right reaction of (eq. 5) is changed as follows:

$$D + E \xrightarrow{C} F + G\,.$$
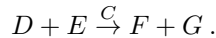
In order to balance out the production of G particles and keep the system (approximately) closed, G particles are removed from the simulation with probability $P_g^- = (1 + \exp[\gamma^-])^{-1}$ for every G at each time step. Furthermore, reaction (eq. 3) is modified such that crystal growth leads to the encapsulation of any C or F particles located in the

same tile, while crystal breakup results in a release of C and F particles:

$$G^n(kC, mF) + G + pC + qF \rightarrow G^{n+1}((k+p)C, (m+q)F),\tag{6}$$

$$G^n(kC, mF) \rightarrow G^{n-1} + G + kC + mF\,,$$

with $k, m, p, q \geq 0$ and $n \geq 2$, and where a crystal of size n containing k C particles and m F particles is denoted as $G^n(kC, mF)$.

Parameterization

The results in figure 3 show that a decrease in $\hat{H}(G,t)$ may occur when γ^+ is fixed at 1. Similarly, figure 4 shows that $\varrho^- > 0.5$ allows for relatively low values of $\hat{H}(R,t)$. Here, we investigate the ranges of γ^- (horizontal axis) and ϱ^+ (vertical axis) that allow for self-assembly and reciprocal catalysis to occur simultaneously.
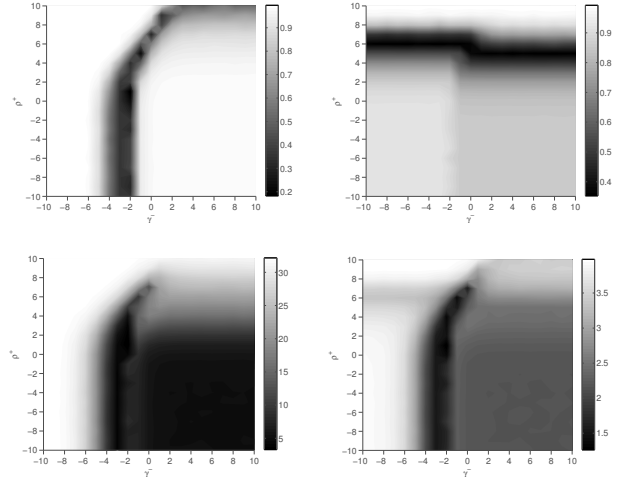
Figure 5: After initializing the simulation with 1000 particles uniformly distributed among types A, B, D and E, we run it for 5000 time steps with $\gamma^+ = 1$, $\varrho^- = 0.5$ and $\gamma^-, \varrho^+ \in [-10, 10]$. The four figures above show the average normalized information entropy of G particle locations (top left), the average normalized information entropy of catalytic reaction locations (top right), the average symmetrized Kullback-Leibler divergence with smoothing, where $SD_{KL}(\gamma^-, \varrho^+) = \max(SD_{KL})$ if no occurrences are found (bottom left), and the sum of these three figures, where SD_{KL} has been normalized using scaling coefficient β (bottom right).

The redundancy between the distributions of G particle locations and catalytic reactions is considered to be an indication of the amount of interaction, i.e. it measures whether crystals tend to be located in proximity to catalytic reactions and vice versa. This redundancy is quantified using the

Kullback-Leibler divergence with smoothing (eq. 2), which is symmetrized to obtain a commutative measure:

$$SD_{KL}(G, R, t) = D_{KL}(G(t)\|R(t)) + D_{KL}(R(t)\|G(t))$$

$$= \sum_{i=1}^{n_{\text{tiles}}} \left[\frac{|G_i(t)|}{|G(t)|} - \frac{|R_i(t)|}{|R(t)|} \right] \log_2 \left[\frac{|G_i(t)|}{|G(t)|} \bigg/ \frac{|R_i(t)|}{|R(t)|} \right],$$

with smoothing (eq. 2) applied if necessary.

Autogenesis requires that self-assembly and reciprocal catalysis both take place in each other's proximity. The desired values for parameters γ^- and ϱ^+ are therefore estimated by minimizing

$$\hat{H}(G, t) + \hat{H}(R, t) + \beta \, SD_{KL}(G, R, t) \,,$$

where coefficient β scales $SD_{KL}(G, R, t)$ to $[0, 2]$

$$\beta = 2 \left(\max_{\gamma^-, \, \rho^+ \in [-10, 10]} SD_{KL}(G, R, t) \right)^{-1}$$

such that the normalized information entropies and the divergence between the distributions contribute equally to the sum.

Comparison
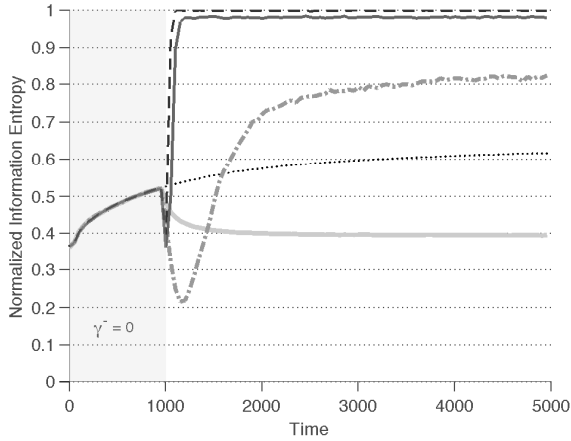
Figure 6: Normalized information entropy $\hat{H}(G, t)$ during autogenesis, with $\gamma^+ = 1$, $\varrho^+ = 6$, $\varrho^- = 0.5$, and $\gamma^- = 0$ for $t \in [0, 1000]$, and 1000 particles evenly distributed among types A, B, D and E initially. For $t \in [1001, 5000]$, $\cdots\cdots$ $\gamma^- = -4$, —— $\gamma^- = -3$, $\cdots\cdots$ $\gamma^- = -2$, —— $\gamma^- = -1$, —— $\gamma^- = 0$. Results are averaged over 1000 trials.

Figure 6 shows the normalized information entropy when, after an initial phase with conditions that enable self-assembly, the value of γ^- is changed. For $\gamma^- \in [-5, -4]$ after $t = 1000$, the high probability of detachment is not conducive for the prolonged persistence of crystals, and they
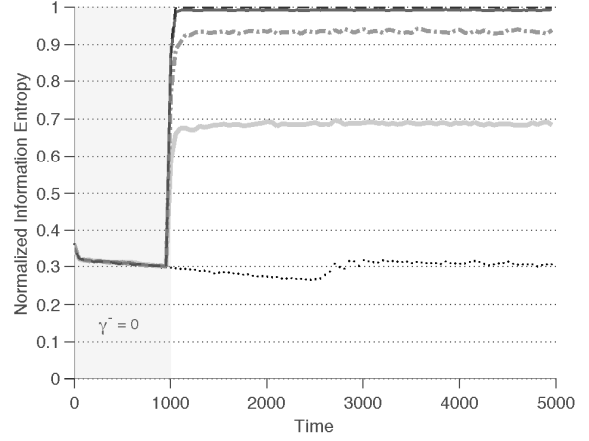
Figure 7: Normalized information entropy $\hat{H}(G, t)$ during self-assembly (without reciprocal catalysis). Here, instead of being generated by a catalytic reaction, G particles are added to random grid locations at the same rate as in the previous simulation (see fig. 6). Again, $\gamma^+ = 1$, $\varrho^+ = 6$, $\varrho^- = 0.5$, and $\gamma^- = 0$ for the first 1000 time steps, while thereafter $\cdots\cdots$ $\gamma^- = -4$, —— $\gamma^- = -3$, $\cdots\cdots$ $\gamma^- = -2$, —— $\gamma^- = -1$, —— $\gamma^- = 0$. Results are averaged over 1000 trials.

fall apart. For γ^- at 0, $\hat{H}(G, t)$ continues to develop unperturbed. Changing γ^- to -1 results in a lower normalized information entropy, as more G particles detach and subsequently attach to larger crystals (cf. fig. 3). With γ^- changed to -2, $\hat{H}(G, t)$ initially drops and eventually finds a new equilibrium at a value higher than with $\gamma^- \in [-1, 0]$.

Figure 7 shows a similar experiment where the particles necessary for reciprocal catalysis (A to F) are omitted from the experiment. Here, G particles are added at random grid locations at the same rate as they were generated by reciprocal catalysis in the previous experiment. Spatial proximity is therefore no longer biased by reciprocal catalysis and the absence of catalysts excludes the possibility of encapsulation.

Comparing both figures, we find that for $\gamma^- = -2$ constraint is preserved when a synergetic linkage between self-assembly and reciprocal catalysis is present while it largely falls apart in the case of mere self-assembly. Also for $\gamma^- = -1$, the value of $\hat{H}(G, t)$ remains lower with this linkage than without it. These particular changes to γ^- show that an autogen may resist dissipation despite unfavorable extrinsic conditions; the intrinsic dynamical constraint between self-assembly and reciprocal catalysis allow it to persist.

Constraint Selection

The preservation of dynamical constraint is a higher-order process: not only is the number of macroscopic states re-

duced as the spatial distribution of events becomes more constrained, but the distribution over specific constraint *types* (e.g. the specific form of crystals) is itself also reduced. To experimentally quantify this second-order reduction we initialize each newly formed crystal with a property c, a random integer between 1 and 5 that affects equation (6) by limiting the total number of catalysts that a crystal may contain. This property c reflects how the shape of a crystal may affect its containment capacity (fig. 8) without explicitly modeling the physical geometry of crystals.

The experiments of figures 6 and 7 are repeated while crystals are assigned random values for c. The average size of crystals is shown in figure 9.

Figure 8: Due to the way G particles attach to one another, crystals with different geometries may come about. This illustration shows several crystals G_c^n of equal size (i.e. $n = 5$) but with different capacities (c) for containing catalysts due their particular shape.
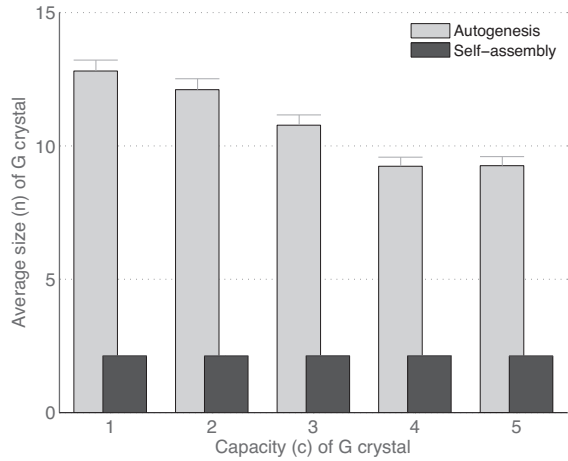
Figure 9: Fixing γ^- at -0.5 and $\gamma^+ = 1, \varrho^+ = 6$, and $\varrho^- = 0.5$, the grid is again initialized with 1000 particles of types A, B, D and E. For 1000 trial runs over 5000 time steps, the mean crystal size and standard error of the mean are reported for the five different containment capacities.

The difference in average crystal size between autogenesis and self-assembly can be inferred from the results of the previous experiments. However, the competition between crystals with different c leads to a second notable difference: the size of an autogen appears to be correlated with its containment capacity. In our simulation the geometry of a crystal does not affect its size directly, as the probabilities of crystal formation and detachment are independent of c.

Rather, the value of c affects the size of crystals indirectly, as the numbers of catalysts in a tile affects the production of new G particles, and thereby the crystal's capacity for reconstitution if those catalysts are released upon detachment. This work cycle creates a difference in size between crystal geometries, a difference which is maintained despite the absence of a direct causal link between crystal geometry and the underlying self-organizing processes.

This higher-order constraint is quantified using the normalized information entropy over the distribution of the containment capacities of crystals. With p_c the probability that a crystal has a containment capacity c, $|G_c^n|$ the number of crystals with capacity c and $|G^n| = \sum_c |G_c^n|$ the total number of crystals,

$$p_c = \frac{|G_c^n|}{|G^n|},$$

$$\hat{H}(G_c^n, t) = -\frac{1}{\log_2 5} \sum_{c=1}^{5} \frac{|G_c^n(t)|}{|G^n(t)|} \log_2 \frac{|G_c^n(t)|}{|G^n(t)|}.$$

Figure 10 shows the average $\hat{H}(G_c^n, t)$ for the previous experiments. Selection of crystal geometries accounts for the difference in normalized information entropy over containment capacities during autogenesis.
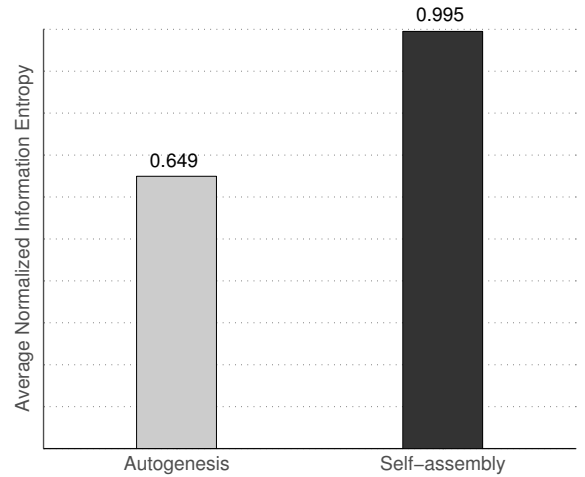
Figure 10: Higher-order constraint: the average normalized information entropy over the distribution of crystal capacities $\hat{H}(G_c^n, t)$ is substantially lower for autogenesis (self-assembly + reciprocal catalysis) than for self-assembly alone. Results averaged over 1000 trial runs.

Conclusions

The statistical distributions used to quantify self-organization and autogenesis in this paper (G particle

locations, R reaction locations, and G_c^n containment capacities) are all expressed in terms of information entropy. This type of quantification does not distinguish between the physico-chemical constraints produced by self-organization and the substrate independent, formal constraint made possible by autogenesis. Taking the physical processes that underlies the maintenance of far-from-equilibrium states into account requires further research (Beer, 2004). New tools capable of expressing this dynamical difference need to be developed (Deacon and Koutroufinis, 2014).

The experimental explorations described in this paper do not quantify all aspects of autogenesis, nor do they provide a complete overview of autogenic properties and phenomena. Rather, they serve to demonstrate the preservation capacity of synergetically coupled processes, and the higher-order reduction of macrostates constituted by formal type selection that may emerge from competition between autogens. These two autogenic capacities may help explain the possible emergence of proto-life. Understanding life's origins does not necessarily imply understanding life as we find it around us today (Cleland, 2013), but the emergent dynamics that created life may also take part in shaping mind and society (Thompson, 2010) as the accumulation of generated constraints that is allowed by preservation through a higher-order linkage is what ultimately makes selection possible.

References

Beer, R. D. (2004). Autopoiesis and cognition in the Game of Life. *Artificial Life*, 10(3):309–326.

Bigi, B. (2003). Using Kullback-Leibler distance for text categorization. In *Proceedings of the 25th European Conference on IR Research*, ECIR'03, pages 305–319, Berlin, Heidelberg. Springer-Verlag.

Bilotta, E. and Pantano, P. (2005). Emergent patterning phenomena in 2D cellular automata. *Artificial life*, 11(3):339–362.

Burton, W. K. and Cabrera, N. (1949). Crystal growth and surface structure. Part I. *Discuss. Faraday Soc.*, 5:33–39.

Cleland, C. E. (2013). Conceptual challenges for contemporary theories of the origin(s) of life. *Current Organic Chemistry*, 17:1704–1709.

Cover, T. M. and Thomas, J. A. (1991). *Elements of Information Theory*. Wiley.

Deacon, T. W. (2006). Reciprocal linkage between self-organizing processes is sufficient for self-reproduction and evolvability. *Biological Theory*, 1(2):136–149.

Deacon, T. W. (2012). *Incomplete Nature: How Mind Emerged from Matter*. W.W. Norton and Company, New York, NY.

Deacon, T. W. and Koutroufinis, S. A. (2014). Complexity and dynamical depth. *Information*, 5(3):404–423.

Eigen, M. and Schuster, P. (1979). *The Hypercycle - A Principle of Natural Self-Organization*. Springer, Heidelberg.

Fellermann, H., Rasmussen, S., Ziock, H.-J., and Solé, R. V. (2007). Life cycle of a minimal protocell—a dissipative particle dynamics study. *Artificial Life*, 13(4):319–345.

Haken (2006). *Information and self-organization: a macroscopic approach to complex systems*. Springer.

Harder, M. and Polani, D. (2013). Self-organizing particle systems. *Advances in Complex Systems*, 16.

Jost, L. (2006). Entropy and diversity. *Oikos*, 113:363–375.

Kauffman, S. A. (1993). *Origins of Order: self-organization and selection in evolution*. Oxford University Press, New York, NY.

King, G. A. (1982). Recycling, reproduction, and life's origins. *Biosystems*, 15:89–97.

Kullback, S. and Leibler, R. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86.

Maturana, H. R. and Varela, F. J. (1980). *Autopoiesis and cognition*. Reidel, Boston, MA.

McMullin, B. and Varela, F. J. (1997). Rediscovering computational autopoiesis. In Husbands, P. and Harvey, I., editors, *Fourth European Conference on Artificial Life*, pages 38–48. MIT Press, Cambridge, MA.

Plasson, R., Brandenburg, A., Jullien, L., and Bersini, H. (2011). Autocatalysis: At the root of self-replication. *Artificial Life*, 17(3):219–236.

Polani, D. (2008). Foundations and formalizations of self-organization. *Advances in Applied Self-Organizing Systems*, pages 19–37.

Prigogine, I. and Stengers, I. (1984). *Order Out of Chaos*. Bantam Books, New York, NY.

Rosen, R. (1991). *Life itself: A Comprehensive Inquiry into the Nature, Origin, and Fabrication of Life*. Columbia University Press, New York.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423.

Thompson, E. (2010). *Mind in Life*. Harvard University Press, Cambridge, MA.

Varela, F. J., Maturana, H. R., and Uribe, R. (1974). Autopoiesis: The organization of living systems, its characterization and a model. *BioSystems*, 5:187–196.

Protocells: what we have learned about minimal life and evolvability

Steen Rasmussen[1,2], Adi Constantinescu[1] and Carsten Svaneborg[1]

[1]Center for Fundamental Living Technology (FLinT), University of Southern Denmark
[2]Santa Fe Institute, New Mexico, USA
steen@sdu.dk

Abstract

In the paper we review lessons learned about two major evolutionary transition from a bottom up construction of protocells. We use a particular systemic protocell design process as a starting point for exploring two fundamental questions: (1) how may minimal living systems emerge from nonliving materials? - and (2) how may minimal living systems support open-ended evolutionary richness?

Non-life-to-life transition

Novel functionalities in physicochemical systems can be generated naturally in three ways: by the assembly of structures (equilibrium processes), by self-organization (non-equilibrium processes) and by a combination of the two, through the evolution of structures (Rasmussen et al., 2001). Our approach to create minimal living systems, which we define as protocells (Szostak et al., 2001, Rasmussen et al., 2004, Sole et al., 2007, Kurihara et al., 2015), utilizes both self-assembling and self-organizing processes. We investigate how a controlled environment together with coupled self-assembly and externally driven self-organization may play together to generate minimal, self-replicating, physicochemical systems. Thus our systemic protocell approach requires a simple metabolism controlled by information both kept together by a container.

We have successfully implemented a particular protocell around a ruthenium tris(bipyridine) Ru(bpy)$_3$ complex that uses light to catalyze redox reactions on precursors of both the amphiphiles and the information in bulk. The informational system serves as part of an electron relay that modulates the metabolic reaction rate, which in turn depends on the redox potential (the nucleobase composition) of the information molecule (DeClue et al., 2009).

In particular, we have established that the amphiphile production can be controlled by chemical information. The reduction potential of a nucleobase, 8-oxo-guanine [oxoG], can be exploited by the photocatalyst to produce amphiphiles, but not that of guanine (the next most easily oxidized nucleobase) or by extension, those of A, C, U and T. Furthermore, fatty acid vesicles will influence the production rates as a detailed investigation of the information-photo-catalyst configuration showed, especially when both oxoG and the Ru(bpy)$_3$ are independently attached through hydrophobic anchors into the container (Maurer et al., 2011). Further we have established a photochemical fragmentation scheme to ligate DNA oligomers. First, the deprotection of an oligomer is performed using a Ru(bpy)$_3$ photosensitizer. This oligomer can only then, and in the presence of a template, be ligated with another oligomer (Cape et al., 2012).

We have demonstrated several advantages of our systemic approach integrating the three mutually supporting components: (i) self-assembly of a decanoic acid container; (ii) anchoring to the container a metabolic ruthenium complex as well as (iii) a conjugated nucleic acid information complex; (iv) container feeding and growth; (v) metabolically driven container replication; (vi) metabolically driven nucleotide oligomer ligation (part of replication); (vii) one pot metabolic production of both amphiphilic molecules and ligated oligomers, new information molecules. These are all key milestones toward the construction of a minimal living system. However, one key milestone is not yet reached before full protocell integration can occur: To implement an effective DNA self-replication process based on template directed ligation of two smaller oligomers.

Missing link: Template directed ligation

We can derive the dependence of the overall replication rate constant on hybridization energies, temperature and strand length, by employing a model for the minimal ligation-based replication process of a single-stranded template in which the ligation of oligomers is involved in the formation of the complementary replica. Within the template directed replicator system, two complementary oligomers hybridize to a single stranded template. An irreversible ligation reaction (i.e., formation of covalent bonds in a condensation reaction) transforms the oligomers into the complementary copy of the template. The newly formed double strand can dehybridize, thus allowing for iteration of the process. Throughout the replication mechanism, we neglect both the production of waste as well as the hydrolysis of ligation. The resulting overall reaction rate is derived, Constantinescu et al., 2016, and summarized below in Fig 1, where two cases are discussed, both assuming parabolic template growth. When product inhibition is rate limiting both longer strands and higher temperatures increase the replication rate. When hybridization is the rate limiting factor, an optimal set of temperature and strand lengths exist.

In the presented protocellular system evolution may be defined in the following way: Compositional information, which is defined as the content and location of oxoG in the information strand, determines the metabolic reaction rates through an electron relay. These processes require a variety of environmental conditions including sacrificial proton and electron donors (DeClue et al., 2009, Maurer et al., 2011, Cape et al., 2012).

Thus a modification of the compositional information generally results in modified metabolic reaction rates. Thus, for the simple protocellular model in a fixed environment, we expect Darwinian evolution to be a metabolic reaction rate *optimization* process where presumably the overall replication rate (the phenotype) is enhanced. At the molecular - or "genotypic" - level, this means a change (through selection
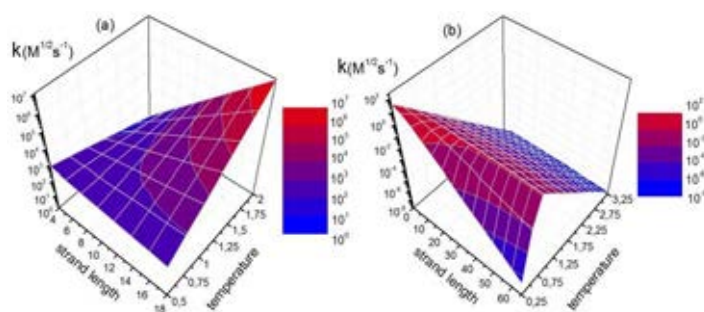
Figure 1. Effective overall replication rate constant k as a function of strand length and temperature. (a) and (b) correspond to a template direct replication mechanism which suffer from product inhibition within a slow (i.e., rate limiting), respectively fast (i.e., not rate limiting) ligation reaction (After Constantinescu et al., 2016). We note that the replication rate depicted in (a) has been obtained by Fellermann and Rasmussen, 2011, employing thermodynamic arguments as well as a polymer model for oligonucleotides that allows simulation of their diffusion and hybridization behavior.

and amplification) of the compositional information of the nucleotide strands, as they are being inherited.

	D_1	D_2	D_3	D_4	D_5	D_6	
excluded volume	X	X	X	X	X	X	billiard dynamics
monomer bonds		X	X	X	X	X	polymer dynamics
fields phobic/philic			X	X	X	X	micelles/vesicles
chemical reactions				X	X	X	metabolic container division
base pairing					X	X	combined info. repl. cont. div.
more fields/reactions						?	more evo. innov. ?

Figure 2. Connection between the details included in the simulations and the ability for the simulations to generate targeted observables. Left side of table summarizes the included physical model (each row). Right side of table indicates the higher order observable phenomena/functionalities generated by the simulation. Top of table depicts the qualitative information details needed in a molecular model representation (the data structure) of the simulation (columns). Simulations with more detailed, and thus more complex, molecular components are able to generate increasingly more complex dynamics and functionalities. As an example, data structure D_3 has included enough molecular interaction details to allow the simulation to generate molecular self-assembly and e.g. micellar and vesicle formation. The last row is left open as we conjecture: to obtain a higher evolutionary potential we need to add more components and/or resources to the system.

Expanding evolvability

We know from experimental and theoretical investigations that if constituent components and environment are too simple, only trivial emergent structures will be generated. As

(appropriate) diversity/complexity of the constituent components is increased, emergence of hierarchies or multilevel structure may occur. Thus, it seems natural to assume this conjecture could also be extended after self-replication and simple Darwinian evolution has been achieved for a protocell. A discussion of the involved constituent components of a protocellular simulation is found in Fig 2.

In practice, more variation could include more and different resource oligomers (short nucleotide libraries), changes in the fatty acid composition, adding different photosynthesizes molecules. Impact or performance could then be measured at the resulting metabolic rate, container division properties and life-cycle (generation) time. However, given our experimental experiences, this would be a challenging and time-consuming enterprise as each new component in the mix in principle could cause undesired (destructive) side effects.

Bedau et al., 1998, propose a statistical characterization of evolutionary processes that aims to quantify the innovative potential of an evolutionary process by measuring the rate that innovative changes are produced during the evolutionary process. In this classification scheme our protocellular systems falls into Class 2. Class 1 is a neural evolutionary process and includes diffusion processes. Class 3 is defined as evolutionary processes with an apparent open-ended ability to innovate and includes examples from biological evolution and technological evolution.

Conclusions

(a) The distinction between non-living and living matter is best characterized as a grey-zone where minimal living systems have the properties discussed above. (b) If we require life to exhibit open-ended evolution, the presented protocell (or for that matter, any published protocellular model we are aware of) does not qualify as a minimal living physicchemi-cal process. However, if Class 2 evolution suffices, several of the published protocellular models, if successfully integrated and experimentally implemented, would qualify as minimal life-forms. (c) To enhance the evolutionary potential of a protocellular system (any system) more richness has to be added to the system. How this system expansion could occur depends on the details of the system.

References

Bedau, Snyder, Packard, Artif. Life VI, eds. Adami, Belew, Kitano, Taylor, MIT Press, p 228

Cape, Edson, Spencer, DeClue, Ziock, Maurer, Rasmussen, Monnard, Boncella (2012) *Bioconjugate Chem.* **23** (10) 2014

Constantinescu, Svaneborg, Rasmussen (2016) *in preparation*

DeClue, Monnard, Bailey, Ziock, Boncella, Rasmussen (2009) *J. Am. Chem. Soc.* **131** 931

Fellermann, Rasmussen (2011) *Entropy* **13**(10) 1882

Kurihara, Okura, Matsuo, Toyota, Suzuki, Sugawara (2015) *Nature Comm.* DOI: 10.1038/ncomms9352

Maurer, DeClue, Albertsen, Dörr, Kuiper, Ziock, Rasmussen, Boncella, Monnard (2011) *ChemPhysChem* **12** 828

Rasmussen, Nilsson, Baas, Mayer, Nilsson, Olesen (2001) *Artf. Life* **7** 329

Rasmussen, Chen, Deamer, Krakauer, Packard, Stadler, Bedau (2004) *Science* **303** (5660) 963

Sole, Bedau, Rasmussen (2007) *Phil. Trans. B* (Spec. Issue) **362** (1486)

Szostak, Bartel, Luisi, *Nature* **409**, 387 (2001)

Microarray of programmable electrochemically active elements

John S. McCaskill[1], Thomas Maeke[1], Lukas Straczek[2], Jürgen Oehm[2], Dominic Funke[3], Pierre Mayr[3], Abhishek Sharma[1], Asbjørn Müller[4], Uwe Tangen[1,4], Norman Packard[4,6], Steen Rasmussen[4,5]

[1]Microsystems Chemistry and Biomolecular Information Processing (BioMIP), Ruhr University Bochum, Germany
[2]Analog Integrated Circuits (AIS), Ruhr University Bochum, Germany
[3]Integrated Systems (IS), Ruhr University Bochum, Germany
[4]Center for Fundamental Living Technology, University of Southern Denmark, Denmark
[5]Santa Fe Institute, New Mexico, USA
[6]European Center for Living Technology, Venice, Italy
john.mccaskill@rub.de

Abstract

This paper describes possible applications of a two dimensional array of programmable electrochemically active elements to Alife. The array has been developed as part of the MICRE-Agents project, and after several design phases, is now a mature enough device for general use beyond the project. Here we describe the general properties of the device based on the first two design phases, some of its capabilities, including portable experimentation, and discuss its potential application to ALife and in education.

Device Origin and Design

The device we describe here is a part of a larger project, MICREAgents (McCaskill *et. al.*, 2012). The original vision of the project included two major technological components: *Lablets* are small (~100 x 100 x 50µm), autonomous electronic elements, comprising a form of smart, programmable, electro-chemically active 'dust', and unlike conventional smart dust communicating via pairwise interactions rather than wireless radiation. Lablets are poured into a solution, and can interact with the surrounding solution, with each other, and with smart surfaces. A *dock* is such a static two dimensional array of 256 x 256 microelectrodes (see Fig. 1) beneath a fluid film and connected to a host computer, from which each of the sites may be independently controlled. One goal of MICREAgents was to develop this technology to enable a new form of evolution through the interaction of chemistry with these new hybrid informational-electrochemical elements.

In this article we concentrate on the dock, and aim to give an overview of some of its properties and capabilities for ALife. We defer technical details to upcoming publications.

Novel dock electronics were developed at AIS to allow individually programmable electrochemical coating of sensors and actuators. Chemical solutions may be applied on the dock, either in the form of droplets covering a subset of dock sites, or a fluid layer covering the entire lattice.

Experimental Examples

Electrochemiluminescence (ECL). Luminescence may be stimulated electronically (Liu, *et. al.* 2015), and this has found widespread application in both display technology and diag-

nostic tests. ECL provides a convenient optical report of electrical activity via electrochemical reactions. We implemented a version of ECL based on Ruthenium Tris(2,2')bipyridyl/Tri-propylamine (Zu, Bard, 2000), illustrated below in Fig. 2, to demonstrate the spatially resolved activation of reactions possible with the dock.

Galvanic deposition. Also known as electroplating, this is one of the chemically simplest forms of electrochemistry. Application of a potential across two electrodes in a salt solution can cause deposition of metal on one of the electrodes. It has been shown that appropriate conditions (concentrations and temporal variations in voltage application) may generate complex structures on sub-micron scales (Sharma, *et .al.*, 2008). Complex surfaces can be useful on the nano- and micro-scales, because they can be used to enhance the surface area of catalysts and also to form supercapacitors.
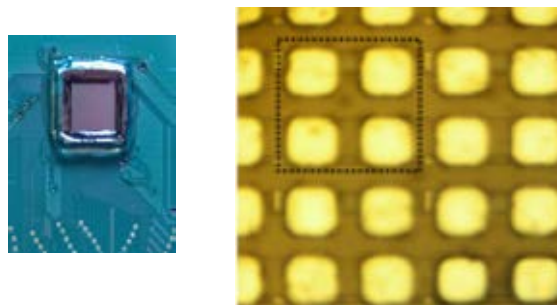
Figure 1. Left: a view of the dock close to actual size (4.6 mm square). Right: A closeup of the dock. A unit cell of the 128 x 128 array contains four 12 µm square electrodes (outlined by the black dotted lines), with a differential sensors and a split actuator. Between the electrodes, the structure of the the electronic control circuitry is visible, on a layer beneath the electrodes.

More complex electrochemical reactions involving DNA and signal amplification can also be initiated, and the project has investigated differential control of chemical reactions using different voltage signals (Freage et.al., 2015).

Figure 2. An implementation of spatial ECL ($Ru(bpy)_3^{2+}$ based) on the (CMOS1) dock, with each lighted site stimulated by the application of a voltage to a microelectrode (global counter electrode).

Combinatorial exploration with the dock

In general, the dock provides a high throughput platform to explore an extremely large space of possible chemical reactions, and chemical reaction sequences. This is true for a wide variety of base chemistries, including DNA (Freage et.al. 2015), peptide, carbohydrate, and RNA chemistries (Chen, La, Zhou, 2014). Efficient use in combinatorial exploration requires reproducible cleaning of microelectrodes. Gold electrodes on the dock can be cleaned either chemically (e.g. (Fischer et. al., 2009) or physically, e.g. with CO_2 snow (Kern, 1990), but some coatings are difficult to clean without resource to mechanical polishing.

Portable experimental setup

A portable experimental setup is constructed using a 3D-printed scaffolding with a USB microscope mounted on top of the dock's computer controller, see Fig 3.

Figure 3. Portable experimental setup for the dock. The microscope camera is mounted atop of an adjustable aluminum tube, mounted onto a 3D printed scaffolding (black - with lights inside), fitted onto a 3D printed dock holder (black), sitting atop of the USB dock-controller box (red). The raspberry pi host computer (running linux) is shown on the right. Cables are removed for clarity.

Because of the nature of the dock's electronic fabrication, it is relatively easy to make many copies, so the dock could also be used in an educational context, with each student able to have their own copy of a dock.

Application to artificial life

The dock should be useful for novel origin of life experiments, to discover chemistry that enables the transition from nonliving to living matter. A version of the Miller-Urey experiment could be implemented, with the dock's spatial separation and control giving far more experimental range. Redox potentials provide a specific source of energy, and specifically coated electrodes provide a programmable distribution of mineral or organic catalysts that can allow controlled investigation of complex spatially resolved chemical evolution.

The dock is also able to interact in programmable ways with microparticles, including the lablets discussed previously. Such interplay between autonomous programmable mobile electrochemical elements and smart docking surfaces may allow the construction of artificially self-reproducing systems with both electronic and chemical facets (Tangen et al., 2015). Further, McCaskill has proposed electronic genomes that can direct chemistry and are heritable. Wills and McCaskill have conceived the electrochemical equivalent of a genetic code: this time coupling copyable electronic with hard to copy chemical systems, rather than DNA genes and proteins.

The high throughput capabilities of the dock could be further enhanced with fluidic overlays. The simplest possible overlay would simply enable laminar flow of fluid across the dock from one side to another. Standard PDMS microfluidics could create 128 independent channels across the dock.

References

Chen, C. D., La, M., & Zhou, B. B. (2014). Strategies for designing of electrochemical microRNA genesensors based on the difference in the structure of RNA and DNA. *Int. J. Electrochem. Sci*, **9**, 7228-7238.

Fischer, L.M., Tenje, M., Heiskanen, A.R., Masuda, N., Castillo, J., Bentien, A., Émneus, J., Jakobsen, M.H. and Boisen, A., 2009. Gold cleaning methods for electrochemical detection applications. *Microelectronic Engineering*, **86**(4), 1282-1285.

Freage, L., Trifonov, A., Tel-Vered, R. , Golub, E., Wang, F., McCaskill, J.S. and Willner I. (2015). Addressing, amplifying and switching DNAzyme functions by electrochemically-triggered release of metal ions. *Chem. Sci.* **6**, 3544–3549.

Kern, W. (1990). The evolution of silicon wafer cleaning technology. *Journal of the Electrochemical Society*, **137**(6), 1887-1892.

Liu, Z., Qi, W., & Xu, G. (2015). Recent advances in electrochemiluminescence. *Chemical Society Reviews*, **44**(10), 3117-3142.

McCaskill, J.S., von Kiedrowski, G., Ohm, J., Mayr, P., Cronin, L., Willner, I., Herrmann, A., Rasmussen, S., Stepanek, F., Packard, N.H. and Wills, P.R., (2012). Microscale Chemically Reactive Electronic Agents. *International Journal of Unconventional Computing* **8**, 289-299. & http://www.micreagents.eu//index.html

Sharma, J., Tai, Y., & Imae, T. (2008). Synthesis of confeito-like gold nanostructures by a solution phase galvanic reaction. *The Journal of Physical Chemistry C,* **112**(44), 17033-17037.

Tangen, U.; Fellermann, H.; Rasmussen, S., Simulating self-replicating, chemically immersed, microchip swarms, *Proc. ECAL 2015*, p. 438

Wagler, P. F.; Tangen, U.; Maeke, T.; McCaskill, J. S. (2012) Field programmable chemistry: Integrated chemical and electronic processing of informational molecules towards electronic chemical cells. *Biosystems*, **109** (1), 2-17.

Zu, Y., & Bard, A. J. (2000). Electrogenerated chemiluminescence. 66. The role of direct coreactant oxidation in the ruthenium tris (2, 2') bipyridyl/tripropylamine system and the effect of halide ions on the emission intensity. *Analytical chemistry*, **72**(14), 3223-3232.

Protein synthesis with liposome fusion and fission by using the freeze-thaw method

Gakushi Tsuji[1], Takeshi Sunami[2], Satoshi Fujii[3] and Tetsuya Yomo[1, 3]

[1]Graduate School of Frontier Biosciences, Osaka University, 1-3 Yamadaoka, Suita, Osaka, 565-0871, Japan
[2] Institute for Academic initiatives, Osaka University, 1-5 Yamadaoka, Suita, Osaka, 565-0871, Japan
[3]Graduate School of Information Science and Technology, Osaka University, 1-5 Yamadaoka, Suita, Osaka, 565-0871, Japan
sunami-takeshi@ist.osaka-u.ac.jp, tetsuyayomo@gmail.com

Abstract

For the experimental construction of artificial cell, it is a challenge to simultaneously supply the nutrients and lipids required for protein synthesis, gene replication, membrane growth, and fission. Inner reactions of liposomes are not permanent because of nutrient exhaustion since liposomes do not have pores or channels on their membrane for acquisition of nutrients. In this study, we demonstrated that the liposome containing in vitro translation system was fused with liposome encapsulating RNA by freeze and thaw for GFP synthesis. The fusion mixed lipid molecules on the two kinds of liposomes, followed by the fission. Consequently, we observed GFP synthesis inside the liposomes after liposome fusion and fission. This freeze and thaw method can be repeated, for the sustainable supplement of nutrients with liposome growth. We hope this method would achieve the ultimate goal of establishing the artificial cells that can acquire nutrients sustainably and proliferate by coupling protein synthesis and gene replication compatible with membrane growth.

Introduction

It is important process to reconstruct life-like compartment with inner biochemical reaction for elucidating the border between life and non-life. The process could give us sights on the origin of life. Repetitive cycles of simple biochemical reaction in liposomes have already been achieved. Recently proliferation of liposomes was achieved with inner DNA replication systems (Kurihara et al., 2015). Also we reported sustainable RNA replication reaction with liposome proliferation by a freeze and thaw method similar to that in this work (Tsuji et al. 2016). Briefly, we mixed two types of liposomes, centrifuged, froze by liquid nitrogen, and thawed at room temperature, and it resulted in liposome fusion and fission. Although these reports succeeded in reconstruction of life-like phenomena, protein synthesis compatible with proliferation of liposomes has not been achieved yet. It has been reported that PURE system, in vitro translation system reconstituted with purified components, could be supplied by liposome fusion and protein synthesis occurred after fusion (Caschera et al., 2011). In this study, we show that PURE system can be supplied to liposomes also by freeze and thaw, and moreover, compatibly with the proliferation of liposomes.

Result

GFP synthesis induced by liposome fusion

We first tried to apply liposome fusion induced by freeze and thaw for supplying PURE system. Previously we reported liposome fusion by freeze and thaw (Tsuji et al., 2016). We prepared liposomes encapsulating RNA which encodes GFP (RNA liposomes), liposomes encapsulating proteins of the PURE system (+nutrient liposomes), and liposomes encapsulating buffer without RNA and the proteins (−nutrient liposomes). After mixing up the RNA liposomes and +nutrient or −nutrient liposomes, we centrifuged the liposomes to produce a liposomal pellet, and fused them by freeze and thaw. Then we incubated the liposomes for 3 hours to induce GFP synthesis. First, we analyzed the size distribution of liposomes before and after freeze and thaw by flow-cytometer (FCM). FCM can measure the size and the fluorescence of each liposome. The results showed that liposome size was hardly changed after freeze and thaw (data not shown) but the lipid markers on the nutrient liposome and RNA liposome were well mixed (Fig 1A, right). The lipid mixing without size change indicates that fusion and fission occurred during freeze and thaw. This result was consistent with our previous report (Tsuji et al., 2016). Second, we measured GFP fluorescence of liposomes by FCM and 8% of the total liposomes showed GFP signals only when +nutrient liposomes were fused with the RNA liposomes (Fig 1A middle, green dots). It should be noted that liposomes with GFP fluorescence appeared only in the region indicating the mixing of the two lipid markers (Fig 1A right, green dots). These data indicate that the two liposomes were fused and inner protein synthesis occurred by mixing of the inner solutions.

Then we observed the liposomes with a confocal laser microscope whether GFP synthesis occurred inside the liposomes. The images show that GFP fluorescence was observed inside the liposomes (Fig 1B). This fluorescence did not appear when RNA liposomes were fused with −nutrient liposomes. Therefore, we concluded that the components of PURE system can be supplied to the liposomes without severe defects of protein activities via freeze and thaw.
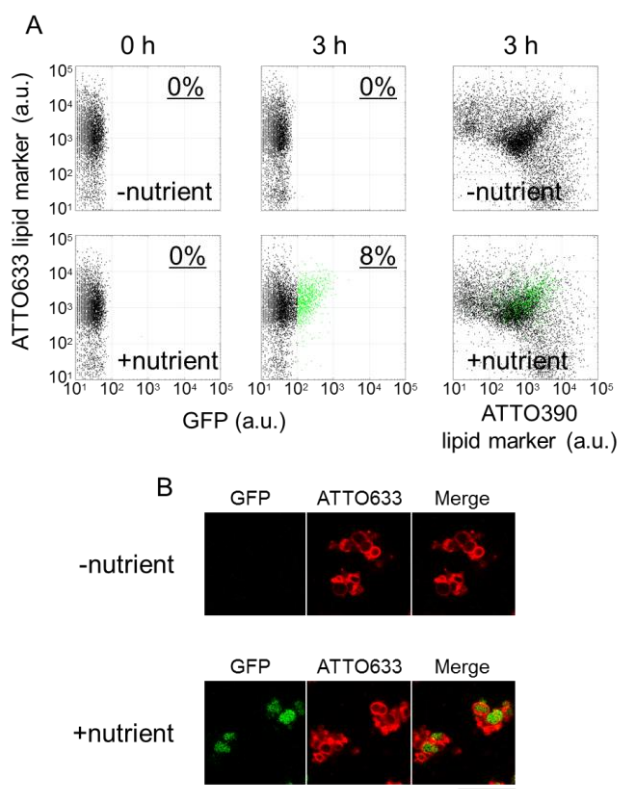
Figure 1 GFP synthesis in liposomes

(A) FCM analysis of GFP synthesis after liposome fusion. Vertical axes of all plots indicate the fluorescent intensity (F.I.) of ATTO633 lipid marker. Horizontal axes of left two columns show F.I of GFP and right column shows F.I. of ATTO390 lipid marker. Green dots indicate liposomes synthesizing GFP (F.I. >100). (B) Microscopic images of 3h incubated samples (scale bar 25μm).

Discussion

We reported the establishment of protein synthesis inside the liposomes by supplying nutrients from outer environment via liposome fusion. This study showed that our liposome fusion method by freeze and thaw can be applied for supplying PURE system. Therefore, we can design the artificial reaction system by introducing the requisite genes and will be able to reconstruct the flexible and extensible life-like structure.

In this report, only 8% of liposomes synthesized GFP, whereas the liposome fusion was observed in higher efficiency in our previous report (50%, Tsuji et al. 2016). This difference was in part because 39 elements were required for GFP synthesis, whereas only 5 elements were required for the previous work. Yet, it is noteworthy that macromolecules such as proteins and tRNAs, which are difficult to pass through the membrane, can be supplied by the freeze and thaw. By developing the gene replication system compatible with the presented freeze and thaw techniques, we will be able to perform "genotype-phenotype linked natural selection" in artificial cells, as a simplest form of "evolvable" protocell model.

Materials and Methods

Materials

1-Palmitoyl-2-oleoyl-sn-glycero-3-phosphocholine (POPC) was purchased from Avanti Polar Lipids (Alabaster, AL). Liquid paraffin (0.86-0.89 g/mL at 20°C) was purchased from Wako (Osaka, Japan). 1,2-dioleoyl-sn-glycero-3-phosphoethanolamine (DOPE) labeled with ATTO 633 (ATTO633) and DOPE labeled with ATTO 390 (ATTO390) were purchased from ATTO-TEC (Siegen, Germany).

GFP synthesis in liposome

The GFP synthesis in liposomes was induced by supplying the PURE system (Kazuta et al. 2014) via the freeze-thaw method. Liposome preparation and liposome fusion were performed as described in previous report (Tsuji et al., 2016). Liposome was prepared by using POPC. RNA encoding GFP (gfp-RNA) was prepared as described in the previous work by Kazuta et al. The RNA liposomes in this experiment were prepared by encapsulating 0.3 mM of each amino acid, 0.8 mM tRNA mix, 3.75 mM ATP, 2.5 mM GTP, 1.25 mM CTP, 1.25 mM UTP, 100 mM HEPES-KOH (pH 7.6), 280 mM potassium glutamate, 1.5 mM spermidine, 19 mM magnesium acetate, 2.5 mM phosphocreatine, 1.5 mM dithiothreitol, 0.01 μg/μl 10-formyl-tetrahydrofolate, 200 mM sucrose, and 2000 nM gfp-RNA. The nutrient liposomes were prepared by encapsulating all constituents of the PURE system, including ribosomes and other proteins required for protein synthesis. The outer solution before freeze contained same components of inner solution of RNA liposomes without RNA and sucrose. Instead of sucrose, 200 mM glucose was added to the outer solution. The outer solution during incubation contained the same components of the outer solution before freeze except that tRNA was not included. After freeze and thaw, liposome solutions were incubated at 37°C for 3 hours. The RNA liposomes and nutrient liposomes were labeled with the fluorescent lipid markers ATTO390 and ATTO633, respectively.

FCM and confocal microscopy analysis

We performed FCM analysis and microscope works as previously reported (Tsuji et al., 2016).

References

Kurihara K., Okura Y., Matsuo M., Toyota T., Suzuki K., and Sugawara T. (2015). A recursive vesicle-based model protocell with a primitive model cell cycle. *Nature comm.*, 6: 8352

Tsuji G., Fujii S., Sunami T., and Yomo T. (2016). Sustainable proliferation of liposomes compatible with inner RNA replication. *PNAS*, 113(3), 590-595.

Fillipo C., Sunami T., Matsuura T., Suzuki H., Martin M. H., and Yomo T. (2011). Program vesicle fusion triggers gene expression. *Langmuir*, 27 (21), 13082–13090.

Kazuta Y, Matsuura T, Ichihashi N, and Yomo T. (2014) Synthesis of milligram quantities of proteins using a reconstituted in vitro protein synthesis system. *J. Biosci Bioeng*. 118(5): 554-557

Self-Optimization and Automation

Evolving Cellular Automata to Perform User-Defined Computations

Paul Grouchy[1] and Gabriele M.T. D'Eleuterio[1]

[1]University of Toronto Institute for Aerospace Studies, Toronto, Ontario, Canada M3H 5T6
paul.grouchy@mail.utoronto.ca, gabriele.deleuterio@utoronto.ca

Abstract

A novel genetic algorithm for evolving both uniform and nonuniform cellular automata (CA) to perform user-defined computations is presented. Unlike previous approaches, the CAs evolved here can in general take as their input and their output only a subset of the cells, allowing for the design of CAs that are larger than the number of inputs required by the desired computation. It also provides greater flexibility compared with previous work in terms of the number of possible outputs. We test our algorithm by attempting to evolve both uniform and nonuniform 1D CAs of varying sizes to compute the sum of two 4-bit strings, a computation requiring 8 inputs and 5 outputs. Results demonstrate that while the algorithm is unable to discover solutions using 8-cell CAs, expanding the number of cells beyond the number of inputs enables the autonomous design of 4-bit adders.

Introduction

In their most basic form, cellular automata (CA) are a collection of simple identical components ("cells") whose behaviors are governed by local interactions (Von Neumann et al., 1966; Wolfram, 1984). Time in a CA is discrete and, at each timestep, a cell can be in one of k states. If $k = 2$, for example, a cell's state can be either 0 or 1 at a given timestep. To execute a CA, one must first "seed" each cell with an initial value. At each subsequent timestep, the CA's rule set determines how a given cell's state is updated based on the cell's current state and the states of its neighbors. The size of the neighborhood of surrounding cells that can influence a given cell's state is determined by the CA's radius r. If $r = 1$, for example, a cell's state is updated based on its current state and the states of its adjoining neighbors. Example rule sets for 1D CAs with $r = 1$ can be found in Figs. 1 and 3.

CAs typically have periodic boundary conditions, i.e., cells in 1D CAs are organized in a ring, cells in 2D CAs are organized in a toroid, etc. Canonical CAs are uniform (homogeneous), meaning that each cell's state is updated using the same rule set. However nonuniform (nonhomogeneous) CAs, where a cell's state can be updated based on one of two or more rule sets, were also studied (e.g., Sipper, 1996). Uniform 1D CAs with $k = 2$ and $r = 1$ are called elementary CAs and it has been shown that at least one such CA,

"rule 110," is Turing complete and thus capable of universal computation (Cook, 2004).

Owing to the size of the set of possible rule sets ($k^{k^{2r+1}}$ in the case of a uniform 1D CA), a brute-force search for CAs that perform a specific computation is often intractable. Therefore, an important area of research is the autonomous design of CAs via genetic algorithms (GAs) and other forms of evolutionary computation (Sapin et al., 2009; Cenek and Mitchell, 2009).

Here we present a novel, flexible genetic algorithm for evolving uniform and nonuniform CAs to perform user-defined computations. A key feature of this algorithm is that the number of inputs and outputs for the desired computation are each allowed to vary independently from each other, as well as from the number of cells in the CA, as long as the CA is at least the same size as the larger of the two. We demonstrate the capabilities of this algorithm by evolving 1D CAs of various sizes to successfully compute the sum of any two 4-bit strings.

Related Work

Nils Aall Barricelli was probably the first person to experiment with evolving CAs, using one of the first computers ever built (Barricelli, 1963). More recently, the work in (Packard, 1988), as well as by the Evolving Cellular Automata group at the Santa Fe Institute (Mitchell et al., 1996), sparked a flurry of research into the use of genetic algorithms to discover CAs capable of performing user-specified computations that continues to this day (for a review, see Iclănzan et al., 2011). In (Mitchell et al., 1993), for example, a GA was employed to discover rule sets for uniform 149-cell CAs with $k = 2$ and $r = 3$ that solve the density classification task (DCT). To solve the DCT, a CA that has its cells seeded with a vector of bits must, after a fixed number of iterations, resolve all of its cells to 1 in the case where the original vector contained more than 50% 1s, otherwise all of its cells should be 0. Thus the number of inputs to the CA is the same as its size, and the output of the computation is determined from the final state of all of the CA cells. In (Sipper, 1996), nonuniform CAs were evolved to

solve the DCT task using a coevolutionary algorithm where each cell had its own genome (rule set) and its own fitness (evaluated independently from the other cell's final states), and selection and reproduction occurred only within local neighborhoods. This work also demonstrated that for $r = 1$, a nonuniform CA could be evolved to achieve a high score on the DCT problem that is theoretically impossible to attain with a uniform CA.

CAs can be evolved to perform a wide range of computational tasks by first framing the problem in a manner similar to the DCT described above. For example, in the synchronization task (e.g., Oliveira et al., 2009), the CA is tasked with alternating between having all of its cells set to 1 and all cells set to 0, after having initially been iterated for a fixed number of timesteps starting from an arbitrary initial state. An interesting property of these types of CA computations is that global consensus among cells is achieved solely through local interactions. Besides searching the space of possible rule sets, genetic algorithms can also search for appropriate neighborhood topologies for a given task, as in (Darabos et al., 2013).

Conway's Game of Life, a 2D CA, is also Turing complete (Berlekamp et al., 2004). Computations in this case are performed by interpreting the interactions of patterns called "gliders" (Sapin et al., 2009). By searching for rules that produce these patterns, GAs can discover novel CAs for simulating logic gates such as AND and NOT gates (e.g., Sapin et al., 2004, 2009).

Finally, controllable CAs (CCAs) are evolvable 1D nonuniform CAs tasked with producing pseudorandom numbers (Guan and Zhang, 2003). At each iteration, the current state of a separate uniform CA (that uses a preselected rule set) is used to determine which of two other preprogrammed rule sets to apply to each cell in the CCA. In addition, a second separate uniform CA (using a fourth preselected rule set) is used to determine whether "controllable" cells act as normal cells, or operate according to a predefined behavior, such as "keep current state." The genetic algorithm is then tasked with determining which cells in the CCA are controllable, and which are considered to be output cells. The values of the output cells get translated into a number after each iteration of the CCA, and after many iterations the final set of numbers are evaluated on their randomness to determine the fitness of the CCA in question. In the algorithm presented below, the locations of input and output cells are evolved in a manner similar to how the locations of the controllable and output cells are evolved in CCAs. However, contrary to CCAs, cells in the work presented here use the same rule set at each iteration, and the number of possible rule sets that can be applied to a cell, as well as the rules themselves, are also evolvable. Furthermore, initial conditions are also evolvable in this work, whereas they are randomly generated in the case of CCAs.

EvoCA

In an effort to expand the range of problems one can tackle using evolved CAs, we present EvoCA, a genetic algorithm that allows users to evolve CAs to perform computations requiring a number of cells equal to *or greater than* the number of inputs. Furthermore, EvoCA allows for the number of output cells to vary from a single cell to having each cell in the CA be an output. The number of allowed output cells is independent from the number of inputs, and vice versa. EvoCA implements these additional features by allowing the genetic algorithm to choose a unique cell for each input and output. While a given cell can only be assigned to at most one input and at most one output, cells can be concurrently designated as both an input and an output.

EvoCA is capable of evolving both uniform and nonuniform CAs. If the user does not confine the search space to uniform CAs, the evolutionary process is free to select the number of potential rule sets that can be applied to a cell, from one (i.e., a uniform CA) to a user-defined maximum. The evolutionary process also selects which rule set is applied to each cell in the CA.

A CA evolved using EvoCA performs a computation as follows. The CA is initialized using the evolvable initial-value vector stored in its genome (see below and Fig. 1). For each cell that is linked to an input, its initial value is overwritten by the current value of its corresponding input. Thus if, for example, our computation takes one input and we set the CA to have five cells in total, then the cell assigned to the lone input will have its initial value defined by the value of the input, while the other four cells will have their initial values determined by the initial-value vector in the CA's genome. Once the CA is initialized in this fashion, it is run for a user-defined number of iterations, with each cell (including those designated as input and output cells) acting as "normal" cells, updating in parallel using the evolvable rule set assigned to them in the genome. At the end of the fixed number of CA iterations, the current values of the designated output cells are taken as the outputs of the computation. Example computations can be found in Fig. 2.

An island model is used as part of the canonical EvoCA algorithm. Island models enhance the canonical single-population GA to evolve multiple reproductively isolated populations in parallel, with periodic exchanges of genomes between islands through "migration." Implementing island models is straightforward, adds little computational overhead, and has been shown to significantly improve GA performance (see Cohoon et al., 1987; Grouchy et al., 2009).

Users must define several parameters before beginning the evolutionary process. A radius r must be provided, as well as the size of the CA and the number of CA iterations per computation. Users must also provide a maximum number of rule sets, where choosing 1 forces the algorithm to search only the space of uniform CAs. Besides these parameters that apply to the CAs themselves, a variety of typical GA
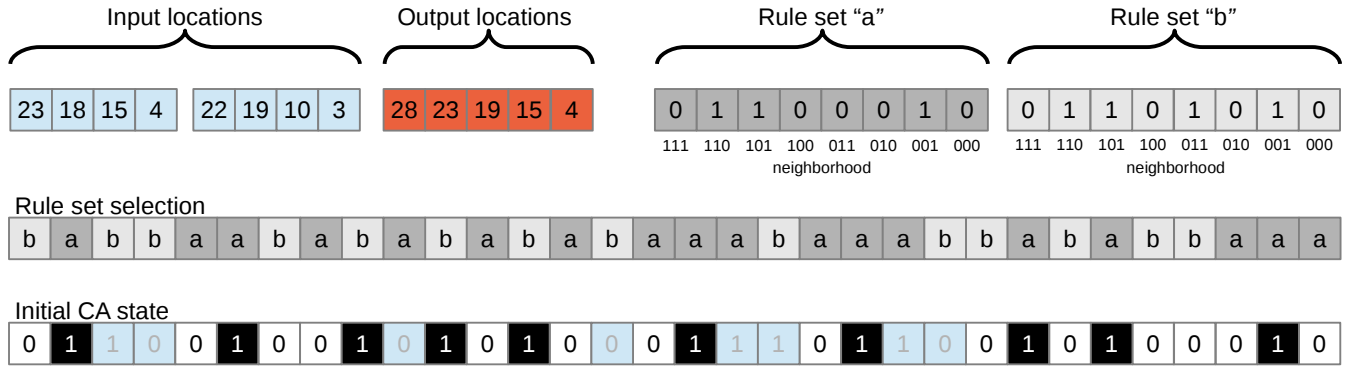
Figure 1: An example EvoCA genome that encodes a 32-cell CA with 8 inputs and 5 outputs. This genome also has two rule sets that differ by a single rule. This set of rule sets is one of the two most frequent sets to appear in discovered solutions. Input cell locations are highlighted in light blue on the initial CA state vector, indicating that the encoded initial values for these cells (shown in light gray) do not affect the CA's behavior. This genome encodes the 4-bit adder whose example behaviors are shown in Fig 2.

parameters must also be set (see below).

The algorithm presented here evolves 1D CAs with $k = 2$ and periodic boundary conditions, although extending EvoCA to evolve other types of CAs is possible.

Genome

We may regard a CA α as a map $\alpha : A^L \to A^L$, where A is the alphabet of states on which the CA operates (for binary CAs $A = \{0, 1\}$) and L is the length of the CA. The cardinality of A, $|A|$, is k and the radius of operation for each cell is set to a constant r, as described previously. However, the actual function of interest that we wish to evolve may be expressed as $\rho_\alpha : A^{L_I} \to A^{L_\Omega}$, where I represents the subset of cells from α that serve as inputs and Ω the subset of cells that serve as outputs; $|I| = L_I \leq L$ is accordingly the number of desired inputs and $|\Omega| = L_\Omega \leq L$ the number of desired outputs. In the present embodiment of this concept, the alphabet A, the radius r, the length of the CA L, and input and output sizes L_I and L_Ω are fixed. The location of input and output cells are mutable although their numbers are not.

Figure 1 shows an example EvoCA genome that encodes a 32-cell CA for a computation requiring 8 inputs and 5 outputs (in fact, this genome encodes a 4-bit adder CA, see below). There is one gene for each input and each output, with each of these genes containing the position of its associated cell represented as an integer in the range $[1, L]$. The genome also contains one or more rule sets, each of size 2^{2r+1}. Owing to the fact that additional rule sets can be added via mutation (see below), genomes in EvoCA are variable-length. To determine which rule set to apply for each cell (in the case of nonuniform CAs), genomes also contain a vector of genes of length L, where each gene corresponds to a cell in the CA and contains the unique identification tag of the rule set to be applied to that cell. Finally,

genomes contain a second vector of genes of length L containing initial values (either 0 or 1 for the work presented here) for each cell. Note that the genes in this initial-value vector that correspond to input cells are "neutral," meaning that they do not affect the behavior of the CA in any way. This is owing to the fact that these initial values get overwritten by input values before the initial iteration of the CA.

Initialization. Before the evolutionary process can begin (at generation 0), an initial population of genomes is needed. These genomes are generated randomly as follows: Input cells are randomly chosen one at a time without replacement from the set of all CA cells. Output cells are chosen in the same fashion. Initial genomes start with only a single rule set, as this was found to produce the best results (see Table 2). To generate a random rule set, a single value λ is chosen at random from the uniform distribution $[0, 1]$. Then for each rule in the rule set, a random value β is drawn from the same distribution as λ. If $\beta < \lambda$, then the current rule will be set to a 1, otherwise it will be set to 0. Using this λ term produces populations of genomes whose rule sets are uniformly distributed across different densities of 1s, as in (Sipper, 1996).

Since genomes are initialized with only a single rule set, the vector in the genome responsible for assigning rule sets to cells will be initialized with each gene pointing to the same initial rule set. However, in experiments where the number of initial rule sets is allowed to vary, a rule set is chosen at random (with uniform probability) for each cell.

Finally, the initial-value vector is randomly initialized in the same fashion as the initial rule set, i.e., using a randomly selected λ term to determine the expected density of 1s.

Reproduction. To produce the next generation of CA genomes (offspring) from the current (parent) generation,

relatively high fitness parent genomes are cloned. If crossover is to be used during the creation of an offspring genome, a second high-fitness parent genome is selected and merged with the offspring genome that was cloned from the first parent. The original offspring genome is preserved from its beginning (i.e., the top-left gene in Fig 1) to a randomly selected crossover point on its genome (moving from left to right and top to bottom along the various parts of the genome, as organized in Fig 1). From the crossover point onwards, the remaining genes on the offspring genome are replaced with their corresponding genes from the second parent's genome. Note that if the offspring genome contains more rule sets than the second parent's genome, its additional rule sets will be preserved regardless of the location of the crossover point. Conversely, if the second parent's genome contains more rule sets than the offspring genome, its additional rule sets will be ignored during crossover. Finally, when copying an input gene from the second parent to the offspring genome, if it is found that the second parent's gene points to a cell that is already associated with a previous input gene on the offspring genome, crossover does not occur for the gene in question, leaving the offspring's original gene unmodified. This rule is also applied when copying output genes.

Mutations. The resulting offspring genomes (either asexually cloned or generated via sexual reproduction, i.e., crossover) are then subject to one or more of a variety of potential mutations:

- Rules in an offspring genome's one or more rule sets can be modified by a bit-flip mutation.

- If the genome does not already contain the user-defined maximum number of rule sets, a new rule set can be created through a mutation. Rules in the new rule set are randomly generated as when rule sets are first initialized at generation 0 (see above).

- Input genes can be assigned to a new, randomly selected CA cell that is not already designed as an input cell. A mutation can also cause two input genes to swap associated cells. Finally, a mutation can cause an input gene to become associated with a cell adjacent to the cell that it is currently associated with. If the adjacent cell is already designated as an input, the next adjacent cell is selected instead (this process will repeat until a cell that is not currently associated with an input is found).

- Output genes are subject to the same three types of mutation as input genes.

- All genes in the initial-value vector are subject to bit-flip mutations.

- If a genome contains two or more rule sets, each gene that determines which rule set to apply to its associated cell is subject to a mutation that randomly selects a new rule set to govern its cell's behavior.

- If a genome contains two or more rule sets and at least one of these rule sets is not associated with any cells, the genome can undergo a mutation that removes all rule sets that are not in use.

Mutations to initial-value genes that are associated with cells designated as inputs are "neutral" in the sense that they will not modify the behavior of the CA. This is again owing to the fact that the initial values of input cells are overwritten by their associated input's value. These neutral genes may be expressed in future generations however, as a mutation may change the cell associated with an input, causing the former input cell to be initialized to its previously neutral initial-value gene at the beginning of a CA computation.

It should be noted that a mutation that adds a rule set to a EvoCA genome does not force it to be used by any of the CA's cells, and thus is also a neutral mutation. However, a future mutation may associate one or more cells with this new rule set, thus expressing the originally neutral mutation. Genomes with two or more rule sets may also end up with neutral rule sets through the accumulation of mutations that disassociate its cells from a previously used rule set (by associating them with other rule sets in the genome). When a rule set is not associated with any cells, its rules are still subject to bit-flip mutations. However, since these rules are not used by the CA, these mutations will also be neutral, perhaps getting expressed in future generations if genomes evolve to (re)use this rule set.

4-bit Adder Experiments

To demonstrate the capabilities of EvoCA, we attempt to evolve CAs that function as 4-bit adders[1]. This requires 8 inputs to the CA (the two 4-bit strings to be summed) and 5 outputs (the resulting 4-bit sum and the final carry bit). This problem is interesting in that at least four computational operations must be done *in the correct order* before a correct answer can be produced. Furthermore, to allow carry bits from previous operations to be passed to subsequent operations, a solution to this problem will necessitate some form of memory.

For all of the experiments described here, a minimal radius of $r = 1$ is used: Each cell has access to only its own state and that of its two adjacent neighbors. Each CA in the population is evaluated on all possible combinations of inputs ($2^8 = 256$ total training cases). For each input string, the CA being evaluated has the Hamming distance between its 5-bit output string and the correct 5-bit answer added to its fitness, which is initially set to 0. This is therefore a minimization problem, and a 4-bit adder CA will have a fitness of 0.

---

[1]The source code for these experiments is freely available at https://github.com/pgrouchy/EvoCA

We use 100 islands arranged in a ring, with each island having a population of 100 CAs, for a total population size of 10,000 CAs per run of EvoCA. Islands evolve in parallel and are reproductively isolated, except for occasional migration events. In the experiments presented here, migration is implemented as follows: islands receive 5 randomly selected genomes from the island to their right every 10 generations and these 5 incoming genomes overwrite randomly selected preexisting genomes in the population.

On each island, tournament selection, where the CA genome with the best fitness (smallest Hamming distance) is selected from a group of randomly chosen CA genomes, is used to select parents for reproduction. A tournament size of 10 is used for all experiments in this paper. To produce an offspring genome, a parent genome selected via tournament selection is cloned. If crossover is to be applied, tournament selection is run a second time to select a second parent whose genome will be spliced with the offspring genome. Offspring genomes are also subject to a variety of mutations (see above). The current top CA on an island is copied into the offspring population without modification (elitism). Otherwise, crossover is applied with a probability of p_c and each gene of offspring i is subject to mutation with a probability of $1/S_i$, where S_i is the size of the offspring's genome. There is a 1% chance that all unused rule sets (i.e., a rule set that is not applied to any of the CA's cells) will be removed from an offspring genome.

A variety of experiments are run to investigate the influence of various parameters of EvoCA on its performance on this problem. For each parameter configuration, 100 runs with varying initial populations are performed and all runs last for 1,000 generations.

Results and Discussion

The results from all performed experiments are summarized in Tables 1, 2, and 3. The fitness value of a given run is reported as the lowest fitness achieved by any genome throughout the entire evolutionary run. P values are calculated using the two-tailed Wilcoxon rank-sum test. Note that preliminary experiments (see Table 3) determined that EvoCA performs better without crossover enabled (i.e., $p_c = 0$), therefore all experiments reported here are with $p_c = 0$, except where noted.

The results in Table 1 demonstrate the need for additional cells beyond the number of inputs. When L is equal to the number of inputs (i.e., $L = 8$), EvoCA is unable to find a CA that can correctly compute the addition of every possible combination of two 4-bit strings. Furthermore, increasing the number of CA iterations has no effect on performance. By increasing L beyond the number of inputs however, significant performance improvements are achieved and EvoCA is able to discover nonuniform CAs that are 4-bit adders (no solutions employing uniform CAs emerged from any of the reported experiments). These results also
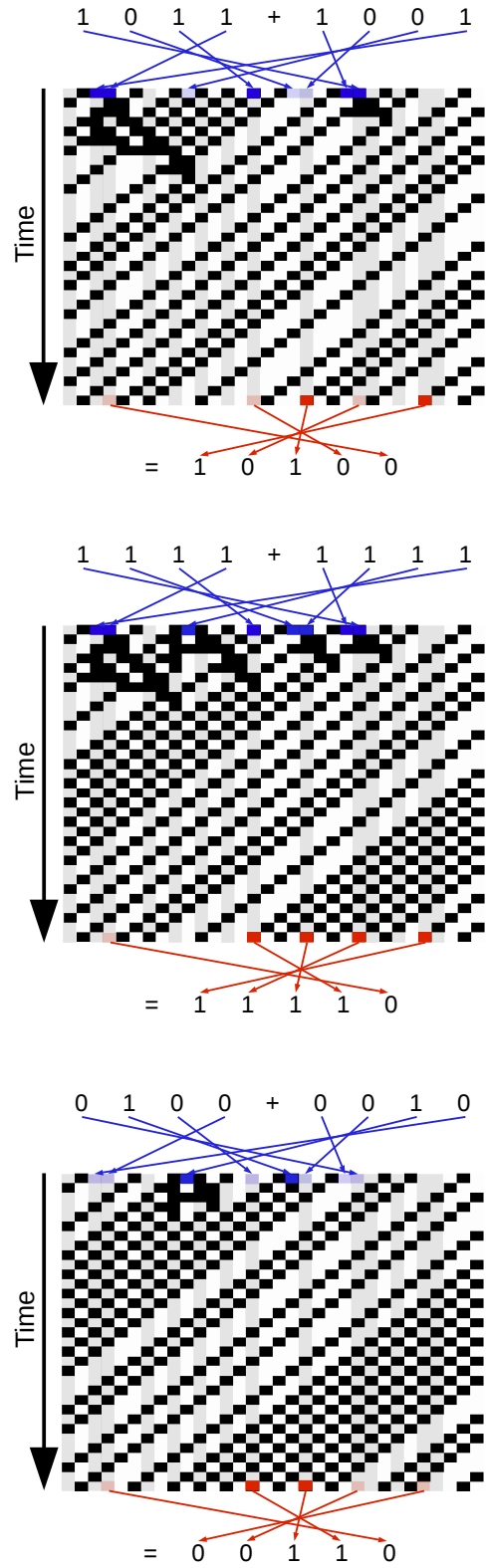
Figure 2: Three different computations from the successfully evolved 4-bit adder CA described by the genome in Fig 1. Cells whose states are updated using rule set "a" are white, while cells whose states are updated using rule set "b" are gray.

CA size	# CA iterations	# success	Fitness		
			μ	σ	Rank-sum
8	8	0	158.43	41.06	
					$P > 0.999$
8	32	0	156.16	38.37	
					$P < 0.001$
24	24	0	70.36	52.51	
					$P < 0.020$
24	32	1	51.77	42.92	
					$P < 0.015$
32	24	11	80.76	62.95	
					$P < 0.060$
32	32	21	64.09	59.50	
					$P < 0.017$
32	40	10	82.31	61.94	
					$P > 0.965$
40	40	11	87.12	60.67	
					$P < 0.016$
40	32	6	108.51	53.79	

Table 1: Summary of the results from experiments where the CA size and the number of CA iterations were varied. Mean and standard deviation are labeled as μ and σ respectively. All runs are mutation-only (i.e., $p_c = 0$) and use the island model described in the main text.

show that there is a point at which adding additional cells no longer affects performance (e.g., when going from 32 cells to 40 cells), while continuing to increase computational costs.

Why would EvoCA only be able to fully solve this problem using CAs with more cells than inputs? One reason could be that with only 8 cells, the GA gets stuck in local minima. By adding additional cells, the GA can exploit added dimensions in the search space to escape such minima. Another possibility is that additional cells beyond the number of inputs allow EvoCA to better control the order and timing of key computations during the CA iterations, e.g., to control when the carry bit from the addition of the least significant input bits is added into the addition of the next two input bits. Lending evidence to this hypothesis is the fact that in all successful results one finds that while two inputs representing the same bit from the two 4-bit input strings (e.g., one input is the most significant bit from the first input string and the second input is the most significant bit from the second input string) are often found associated with adjoining cells, there are always at least two noninput cells separating inputs from different locations in the bit strings (e.g., Figs. 1, 2, and 4). Since $r = 1$ in all ex-

periments, this separation ensures that cells associated with different bit locations in the input strings will not influence each other's initial state updates. Moreover, by having additional cells beyond the number of inputs, the EvoCA algorithm can exert greater control over which rules get applied at the initial iteration via the evolvable initial values. It is likely that all of the aforementioned hypotheses contribute to EvoCA's success when the number of CA cells is greater than the number of inputs, although further experimentation is required before conclusions can be drawn.

The data in Table 1 also yield no obvious rule for choosing the number of CA iterations to maximize EvoCA performance given a specific CA size. This indicates that future versions of EvoCA should consider allowing the number of iterations to be evolvable, alongside the other parameters already incorporated into the genome.

The experiments summarized in Table 2 explore EvoCA performance with and without restrictions on the use of nonuniform CAs. The top two rows show the results from experiments where nonuniform CAs are allowed to evolve, the difference being that the data in the first row are from experiments where populations were initialized with uniform CAs only, while the data in the second row are from experiments where populations are initialized with a variety of uniform and nonuniform CAs. The third row of data are from experiments where only uniform CAs could be evolved. These results clearly demonstrate that allowing nonuniform CAs to evolve is necessary for the success of the EvoCA algorithm on this task. Furthermore, significant performance improvements are achieved by restricting the initial population of CAs to be uniform only. This is to be expected, however, as additional rule sets increase the size

CA type	# init. rule sets	# success	Fitness		
			μ	σ	Rank-sum
n-u	1	21	64.09	59.50	
					$P < 0.001$
n-u	[1-8]	2	129.20	45.27	
					$P < 0.001$
u	1	0	175.18	46.43	

Table 2: Summary of the results from experiments that explored GA performance with and without restrictions on the use of nonuniform CAs. Note that the top-most row of data are reproduced from Table 1 for comparison. Mean and standard deviation are labeled as μ and σ respectively. Experiments with a CA type of "n-u" allow both uniform and nonuniform CAs to evolve, while experiments with a CA type of "u" are restricted to uniform CAs only. All runs are mutation-only (i.e., $p_c = 0$) and use the island model described in the main text.

p_c	Island model	# success	Fitness		
			μ	σ	Rank-sum
0.0	yes	21	64.09	59.50	
					$P < 0.001$
0.3	yes	3	96.56	58.40	
					$P > 0.853$
0.7	yes	8	96.84	66.02	
					$P < 0.001$
0.7	no	1	203.11	65.14	
					$P < 0.003$
0.0	no	1	174.41	58.09	

Table 3: Summary of the results from experiments where the CA size and the number of CA iterations were both fixed at 32, while the percent of offspring genomes created using crossover p_c and the type of GA used (island model or canonical) are varied. Note that the top-most row of data are reproduced from Table 1 for comparison. Mean and standard deviation are labeled as μ and σ respectively.

of the genome, and thus the size of the search space (e.g., Stanley and Miikkulainen, 2002).

The experiments summarized in Table 3 explore EvoCA performance when varying whether or not an island model and/or crossover are used. These results demonstrate that significant performance improvements are achieved using the island model described here. Surprisingly, these results also show that using the crossover method described here significantly *reduces* performance, and thus best results are achieved using mutation-only (i.e., $p_c = 0$) EvoCA. There are many possible reasons for such a result. Perhaps the single-point crossover mechanism as described needs to be

Rule set "c"

0	0	1	1	1	0	0	0
111	110	101	100	011	010	001	000

neighborhood

Rule set "d"

0	1	1	1	1	0	0	0
111	110	101	100	011	010	001	000

neighborhood
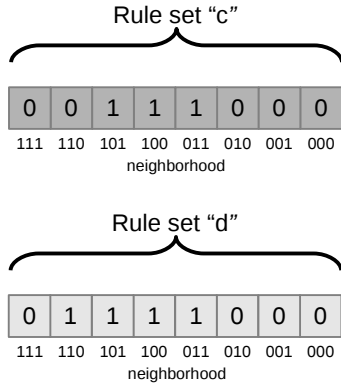
Figure 3: This set of rule sets and the one shown in Fig. 1 are the two most common sets among the 4-bit adder CAs discovered by EvoCA. An example behavior of a solution that employs these rule sets is shown in Fig. 4.
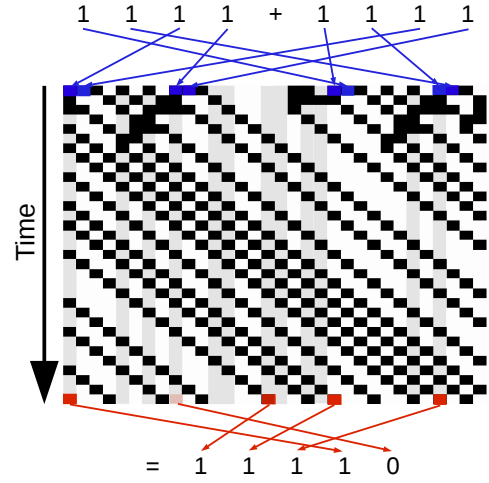
Figure 4: An example behavior from an evolved 4-bit adder that employs the set of rule sets shown in Fig. 3. Cells whose states are updated using rule set "c" are white, while cells whose states are updated using rule set "d" are gray.

refined. Or perhaps single-point crossover works better with a different arrangement of genes in the genome than the one used here (see Fig. 1). Better results might also be achieved by developing an alternative method of crossover, perhaps based on uniform crossover (where crossover is applied at each gene independently, thus the arrangement of genes in the genome has no effect on performance). Finally, it is possible that this problem domain, or even the search spaces engendered by the EvoCA algorithm in general, are best searched with mutation-only GAs. Again, conclusions cannot be drawn without further experimentation.

While the positions of the input and output cells vary considerably between solutions (which is to be expected considering the CA has periodic boundary conditions and addition is commutative), 85.3% of all successfully evolved 4-bit adders contain one of two sets of rule sets: the two rule sets shown in Fig. 1, and the two rule sets shown in Fig. 3. A sample behavior from a solution that employs the set of rule sets shown in Fig. 3 is shown in Fig. 4. Two types of solutions with three rule sets were also discovered, as well as an additional solution with two rule sets for a 40-cell CA. No uniform solutions or solutions with four or more rule sets emerged.

Conclusions and Future Work

We have presented EvoCA, a novel genetic algorithm for evolving both uniform and nonuniform CAs to perform user-defined computations. This algorithm allows the size of the CA to be greater than the number of inputs, a necessity when searching for solutions to the 4-bit adder problem described here, and contrary to previous work. Furthermore, the number of outputs is free to vary from 1 to the size of the CA,

independent of the number of inputs. The canonical version of EvoCA uses an island model to significantly improve the performance of the GA, although the presented crossover mechanism was found to be significantly detrimental to performance. Thus, future work should look to further investigate crossover mechanisms. Another avenue of research is to explore the trade-offs between increased computational complexity and increased performance when incorporating GA variants such as novelty search (Lehman and Stanley, 2008) and speciating GAs (e.g., Grouchy et al., 2009) into EvoCA. Of course, EvoCA has only proven itself on a single problem thus far, therefore future work should apply EvoCA to many other problems.

Finally, the EvoCA runs presented here were computationally intensive, with each run testing 10,000 CAs on 256 different training cases per generation, for 1,000 generations. Thus an important step towards applying EvoCA to more challenging problems will be to implement it on GPUs, something that should be relatively straightforward owing to the inherently parallel nature of both genetic algorithms and CAs (Žaloudek et al., 2010).

References

Barricelli, N. A. (1963). Numerical testing of evolution theories. *Acta Biotheoretica*, 16(3-4):99–126.

Berlekamp, E. R., Conway, J. H., and Guy, R. K. (2004). Winning ways for your mathematical plays.

Cenek, M. and Mitchell, M. (2009). Evolving cellular automata. *Encyclopedia of Complexity and Systems Science*, pages 3233–3242.

Cohoon, J. P., Hegde, S. U., Martin, W. N., and Richards, D. (1987). Punctuated equilibria: a parallel genetic algorithm. In *Proceedings of the second international conference on genetic algorithms and their application*, pages 148–154. L. Erlbaum Associates Inc.

Cook, M. (2004). Universality in elementary cellular automata. *Complex systems*, 15(1):1–40.

Darabos, C., Mackenzien, C. O., Tomassini, M., Giacobini, M., and Moore, J. H. (2013). Cellular automata coevolution of update functions and topologies: A tradeoff between accuracy and speed. In *Advances in Artificial Life, ECAL*, volume 12, pages 340–347.

Grouchy, P., Thangavelautham, J., and D'Eleuterio, G. M. (2009). An island model for high-dimensional genomes using phylogenetic speciation and species barcoding. In *GECCO'09*, pages 1355–1362. ACM.

Guan, S.-U. and Zhang, S. (2003). An evolutionary approach to the design of controllable cellular automata structure for random number generation. *Evolutionary Computation, IEEE Transactions on*, 7(1):23–36.

Iclănzan, D., Gog, A., and Chira, C. (2011). Enhancing the computational mechanics of cellular automata. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2011)*, pages 267–283. Springer.

Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pages 329–336.

Mitchell, M., Crutchfield, J. P., Das, R., et al. (1996). Evolving cellular automata with genetic algorithms: A review of recent work. In *EvCA'96*. Moscow.

Mitchell, M., Hraber, P., and Crutchfield, J. P. (1993). Revisiting the edge of chaos: Evolving cellular automata to perform computations. *arXiv preprint adap-org/9303003*.

Oliveira, G. M., Martins, L. G., de Carvalho, L. B., and Fynn, E. (2009). Some investigations about synchronization and density classification tasks in one-dimensional and two-dimensional cellular automata rule spaces. *Electronic Notes in Theoretical Computer Science*, 252:121–142.

Packard, N. H. (1988). *Adaptation toward the edge of chaos*. University of Illinois at Urbana-Champaign, Center for Complex Systems Research.

Sapin, E., Bailleux, O., and Chabrier, J.-J. (2004). Research of complex forms in cellular automata by evolutionary algorithms. In *Artificial Evolution*, pages 357–367. Springer.

Sapin, E., Bull, L., and Adamatzky, A. (2009). Genetic approaches to search for computing patterns in cellular automata. *Computational Intelligence Magazine, IEEE*, 4(3):20–28.

Sipper, M. (1996). Co-evolving non-uniform cellular automata to perform computations. *Physica D: Nonlinear Phenomena*, 92(3):193–208.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127.

Von Neumann, J., Burks, A. W., et al. (1966). Theory of self-reproducing automata. *IEEE Transactions on Neural Networks*, 5(1):3–14.

Wolfram, S. (1984). Cellular automata as models of complexity. *Nature*, 311(5985):419–424.

Žaloudek, L., Sekanina, L., and Šimek, V. (2010). Accelerating cellular automata evolution on graphics processing units. *International Journal on Advances in Software*, 3(1 & 2).

Expansion of Perception Area in Cellular Automata Using Recursive Algorithm

Yoshihiko Kayama

Department of Media and Information, BAIKA Women's University
2-19-5 Shukuno-sho, Ibaraki 567-8578, Osaka, Japan
y_kayama@ieee.org

Abstract

This study aims to present a new idea to expand the perception area of each cell in cellular automaton (CA) using a recursive algorithm known as "Recursive Estimation of Neighbors." An intelligent cellular process defined by the algorithm makes it possible to introduce an extra radius of the perception area, in addition to the radius of the CA neighborhood. A basic CA rule is extrapolated into rules with larger radii, which form a sequence indexed by the extra radius containing the basic CA as the first term of the sequence. The patterns formed in some typical sequences of extrapolated ECA and Life-like CA rules are presented. Contrasting pattern activities contained in homogeneous and heterogeneous CAs are discussed by applying mean field analysis. Some symmetrical arrangements of composites of cells with different radii are used in order to discuss the heterogeneous CA. The new perspective presented here offers several possible applications for CA.

Introduction

Cellular automaton (CA) is characterized by a large number of cells and a synchronous update of all cell states according to a local rule. As a result, CA has been used to describe the complexity emerging from interactions among simple individuals following simple rules. The original concept of CA was introduced by von Neumann and Ulam for modeling biological self-reproduction (Neuman, 1966). In the 1970s, Conway developed a two-dimensional CA rule, which he called "the Game of Life," that exhibited complex behaviors evoking biological activities (Gardner, 1970 and Berlekamp, Conway, and Guy, 1982). In the 1980s, Wolfram studied one-dimensional CAs (Wolfram, 1983, 1984, 1986, 2002), and proposed that CA could be grouped into four classes of complexity: homogeneous (class I), periodic (class II), chaotic (class III), and complex (class IV). This study mainly discusses a possibility to extend intelligence of each cell and presents a new perspective of CA for discussing relationship between information processing and pattern formation.

If intelligence can be described as the ability to perceive information and use it to form adaptive behaviors within an environment, each cell of CA seems not intelligent because it does not incorporate a process for organizing the information to determine its own future state. It is rather important that even such a simple model can simulate complex patterns reminding biological activities. In contrast, in a flocking boids simulation, which is a typical model of multi-agent system developed by Reynolds (Reynolds, 1987, and Banks, Vincent, and Anyakoha, 2007), each boid obtains the motion information of other boids within its perception area and using a simple algorithm, alters its own motion according to an analysis of this information. Boids easily organize themselves into a large, orderly group and move as a single organism without a central commander, i.e., their information processing leads to a collective control of the group motion. There seem to be a crucial difference between the two models, CA and the boids.

Actually, a framework inspired by that of boids, named the *Recursive Estimation of Neighbors* (REN), was introduced into CA to construct a model for studying the relationship between information processing and pattern formation in collective systems at *the 21st AROB international conference* (Kayama, 2016). In the present study, this idea is clarified and the formations of some interesting patterns in the studied CAs are investigated. It can be seen that the new model does not surpass the framework of CA but its reconstruction or reinterpretation. A basic CA rule with a unit rule radius is extrapolated into rules with larger radii through the REN algorithm. The extrapolated rules form a sequence indexed by an extra radius that represents the size of the perception area of a cell. The sequence contains the basic CA as its first term. When we call a CA model comprising cells with different values of the extra radius as *heterogeneous*, some models show interesting pattern formations. Contrasting examples in two-dimensional CA are discussed through the mean field analysis.

The next section shows how the intelligent process of each cell can be implemented using the REN algorithm, through the introduction of the extra radius in addition to the radius of the basic CA neighborhood. In Section 3, some typical sequences of the extrapolated CA rules in one-dimensional elementary CA (ECA) and two-dimensional eight-neighbor

outer-totalistic CA including Conway's Game of Life (Life-like CA) are presented. Sequences of ECA #22 and #110 modify their complexity between periodic and chaotic patterns depending on the even-odd parity of their extra index. In the extrapolated Game of Life sequence, there is a positive correlation between the extra radius and the average convergence speed to a rest state; however, in another sample sequence, there is a negative correlation. Such contrasting activities are interpreted by applying the mean field analysis in Section 4. Some interesting patterns are formed in heterogeneous models; their lattices are composed of cells that follow different extrapolated rules over a given sequence. Some geometrical arrangements of such different types of cells form composite models, which provide new application possibilities for CA.

Recursive Estimation of Neighbors

Practice is an activity to improve skills. When acquiring a skill, we initially try to recognize it as a set or some sequence of small actions. After repeating practice, such a process becomes unconscious, and the skill can be used just as a reflex action, in which no intelligent activities seem to be involved. An experienced person can deal with a lot of information almost automatically, whereas a beginner is likely to be at a standstill in front of it. If a CA rule is considered similar to such psychological insight, it might be possible to represent the rule by a set or some sequence of processes.

In case of boids, each boid acquires information regarding the positions and velocities of boids within its perception area and determines its own movement in order to follow the representative values of the neighbors. The radius of the perception area can be treated as a parameter expressing the differences between individual elements. In order to incorporate a similar scenario in CA, the perception area of a cell should be separated from the neighborhood determined by the CA rule, so that the size of the area can be treated as an attribute of each cell. Under the CA framework, however, the neighborhood of each cell is defined by the CA rule, and there is no possibility of expanding the sensory area of a cell. For example, each cell of ECA acquires the states of the three cells within its radius-one neighborhood to determine its state in the next timestep. The above psychological discussion suggests that such separation can be possible if the update process of each cell has an intermediate process of estimation of next states of neighboring cells as follows:

Acquire information of neighbors $\Rightarrow$ estimate their next states $\Rightarrow$ determine its own next state

Estimation and determination of states are assumed to be processed by only a basic CA rule because if other rules or mechanisms were introduced, the present framework would become complicated and difficult to find a reasonable selection method. Moreover, here we assume "self-similarity," which means that all cells use the same update algorithm. Then, the basic CA rule is expected to be used recursively.

Following the above discussion, the target framework includes the perception area for each cell in addition to the basic CA neighborhood. The states of all cells within the area are perceived in each timestep. Here we assume that the basic CA neighborhood and the perception area are both isotropic and can be parametrized by their radii r and R:

r: radius of the basic CA neighborhood shared by all cells.

R: radius depending on the size of the perception area of each cell.

The intelligent process of each cell is implemented by the REN algorithm, in which the basic CA rule is recursively used to estimate the states of the neighboring cells. The perceptual information of the states of cells inside the perception area of a cell should be consumed in an intelligent manner in the update process of the cell. The recursive usage of the basic CA rule continues until the state of the target cell is subsequently determined. Note that in the estimation process the values of the extra radius R of neighboring cells are assumed by the target cell and the assumed values are not necessarily identical with their actual ones.

The REN algorithm is defined as follows:

(i) In the estimation process, it is assumed that the same algorithm of recursive estimation is used by all neighboring cells (self-similarity). Within the perception area, each neighboring cell is assumed to have a perception area that is as large as possible.

(ii) In cases where the assumed perception area of a cell is smaller than the neighborhood of the basic CA rule ($R < r$), the cell in the next timestep is assumed to remain equal to the current state (termination condition).

Demonstrating the implementation in ECA, which is the simplest one-dimensional binary CA with $r = 1$, will help clarifying the REN concept. Here we suppose that an ECA model is *homogeneous*, which means that all cells have the same value of R.

The state of the i-th cell at timestep t and the ECA rule function are denoted by $x_i^{(t)}$ and f, respectively. The standard time evolution of the state is expressed by

$$x_i^{(t+1)} = f(x_{i-1}^{(t)}, x_i^{(t)}, x_{i+1}^{(t)}). \qquad (1)$$

The new framework requires that the states of all neighboring cells at $t + 1$ are estimated by the ECA rule. Then the above expression changes to

$$\varphi_{R_0,i}^{(t+1)} = f(\varphi_{R_0-1,i-1}^{(t+1)}, x_i^{(t)}, \varphi_{R_0-1,i+1}^{(t+1)}), \qquad (2)$$

where $\varphi_{R_0,i}^{(t+1)}$ is an estimated state of the i-th cell at $t + 1$ with radius $R = R_0$, and $\varphi_{R_0-1,i\pm1}^{(t+1)}$ are estimated states of the adjacent neighbors at $t + 1$ with an *assumed* radius

$R_0 - 1$; the value of the neighbors' radius stems from the above definition (i) because $R_0 - 1$ is the maximum value of the perception area for the neighbors within the perception area of the i-th cell with radius R_0. Note that $\varphi_{R_0,i}^{(t+1)}$ is assigned to the actual state $x_i^{(t+1)}$, but $\varphi_{R_0-1,i\pm1}^{(t+1)}$ are not necessarily equal to their respective actual states $x_{i\pm1}^{(t+1)}$ because the neighbors' true radius is not $R_0 - 1$, but R_0. Subsequently, the definition (i) leads to the following recursive expressions of the estimated states of the neighbors:

$$\varphi_{R_0-j,i-j}^{(t+1)} = f(\varphi_{R_0-j-1,i-j-1}^{(t+1)}, x_{i-j}^{(t)}, \varphi_{R_0-j-1,i-j+1}^{(t+1)}), \quad (3)$$
$$\varphi_{R_0-j,i+j}^{(t+1)} = f(\varphi_{R_0-j-1,i+j-1}^{(t+1)}, x_{i+j}^{(t)}, \varphi_{R_0-j-1,i+j+1}^{(t+1)}), \quad (4)$$

where $j = 1, 2, \ldots, R_0 - 1$. Because $j = R_0$ implies that the estimated value of R equals 0 ($< r = 1$), the definition (ii) gives the following conditions:

$$\varphi_{0,i\pm R_0}^{(t+1)} = x_{i\pm R_0}^{(t)}, \quad \varphi_{0,i\pm R_0\mp2}^{(t+1)} = x_{i\pm R_0\mp2}^{(t)}, \quad (5)$$

which terminate the above recursiveness.

As the first step of a concrete demonstration, let us consider the case $R_0 = r = 1$. Equations 2 and 5 give $x_i^{(t+1)} = f(\varphi_{0,i-1}^{(t+1)}, x_i^{(t)}, \varphi_{0,i+1}^{(t+1)})$ and $\varphi_{0,i\pm1}^{(t+1)} = x_{i\pm1}^{(t)}$. Accordingly, the ECA model with $R_0 = 1$ is identical to the basic ECA (Eq. 1). This discussion is not restricted to ECA; all CA models with $R_0 = r$ are identical to their basic CA. We next discuss the case $R_0 = 2$. Equation 2 gives $x_i^{(t+1)} = f(\varphi_{1,i-1}^{(t+1)}, x_i^{(t)}, \varphi_{1,i+1}^{(t+1)})$, and the recursive expressions 3 and 4 give

$$\varphi_{1,i-1}^{(t+1)} = f(\varphi_{0,i-2}^{(t+1)}, x_{i-1}^{(t)}, \varphi_{0,i}^{(t+1)}), \quad (6)$$
$$\varphi_{1,i+1}^{(t+1)} = f(\varphi_{0,i}^{(t+1)}, x_{i+1}^{(t)}, \varphi_{0,i+2}^{(t+1)}), \quad (7)$$

Because $\varphi_{0,i}^{(t+1)} = x_i^{(t)}$ and $\varphi_{0,i\pm2}^{(t+1)} = x_{i\pm2}^{(t)}$ from the termination conditions (Eqs. 5), the CA model with $R_0 = 2$ is expressed by

$$x_i^{(t+1)} = f(f(x_{i-2}^{(t)}, x_{i-1}^{(t)}, x_i^{(t)}), x_i^{(t)}, f(x_i^{(t)}, x_{i+1}^{(t)}, x_{i+2}^{(t)})), \quad (8)$$

which corresponds to a five-neighbor CA rule, $x_i^{(t+1)} = g(x_{i-2}^{(t)}, x_{i-1}^{(t)}, x_i^{(t)}, x_{i+1}^{(t)}, x_{i+2}^{(t)})$. Equation 8 is an "intelligent" expression of the rule g. The cases for larger values of R can be derived similarly.

The above discussion indicates that the REN algorithm with increasing sizes of perception areas indexed by R extrapolates the basic CA rule to rules with larger radius $r = 2 \times R + 1$. The extrapolated rules from the basic CA form a *sequence* parametrized by R. Namely, a CA rule included in the sequence can be reconstructed from the basic CA through REN, where each cell has its own perception area and acts as an intelligent agent.

Sequence of Extrapolated CA rules

When a basic CA is assigned to a code $\mathcal{N}$, a sequence of extrapolated rules from the basic CA with index R is represented by $[\mathcal{N}]$. If each CA included in the sequence should be identified, its code is shown as the basic CA code followed by the letter "R" and its value n. The sequence is represented by $[\mathcal{N}]=\{\mathcal{N}\text{R1}, \mathcal{N}\text{R2}, \mathcal{N}\text{R3}, \ldots\}$, where $\mathcal{N}\text{R1}$ is identical with the basic CA. Each cell that belongs to $\mathcal{N}\text{R}n$ is called an "$\mathcal{N}\text{R}n$ cell."

The implementation of REN discussed in the previous section can be applied to Life-like CA. A pattern that does not change from one generation to the next is known as a "still life" in the Game of Life and other CA. The following property of the sequence of extrapolated CAs is proved:

- A still life in $\mathcal{N}\text{R1}$ is also a still life in any CA included in the sequence $[\mathcal{N}]$, when it is sufficiently isolated.

"Sufficiently isolated" means that any perception area of cells that form the located still life contains no active cells other than the member of the still life. To prove this property, we adopt the notation of the previous section. The generalization to other cases, e.g., Life-like CA, is straightforward.

(1) A pattern *SL* in $\mathcal{N}\text{R1}$ is assumed to be a still life, which means that for any cell i in *SL*,

$$\begin{aligned} x_i^{(t+1)} &= \varphi_{1,i}^{(t+1)} = f(x_{i-1}^{(t)}, x_i^{(t)}, x_{i+1}^{(t)}) \\ &= x_i^{(t)}. \end{aligned}$$

(2) In $\mathcal{N}\text{R2}$, any state x_i follows

$$x_i^{(t+1)} = \varphi_{2,i}^{(t+1)} = f(\varphi_{1,i-1}^{(t+1)}, x_i^{(t)}, \varphi_{1,i+1}^{(t+1)}).$$

From (1) and the assumption "sufficiently isolated," $x_{i\pm1}^{(t+1)} = \varphi_{1,i\pm1}^{(t+1)} = x_{i\pm1}^{(t)}$ are also satisfied in *SL*. Then,

$$\begin{aligned} x_i^{(t+1)} &= f(x_{i-1}^{(t)}, x_i^{(t)}, x_{i+1}^{(t)}) \\ &= x_i^{(t)}. \end{aligned}$$

(3) If *SL* is assumed to be a still life in $\mathcal{N}\text{R}n$, then

$$x_i^{(t+1)} = \varphi_{n,i}^{(t+1)} = x_i^{(t)}.$$

(4) In $\mathcal{N}\text{R}(n + 1)$, any state x_i follows

$$x_i^{(t+1)} = \varphi_{n+1,i}^{(t+1)} = f(\varphi_{n,i-1}^{(t+1)}, x_i^{(t)}, \varphi_{n,i+1}^{(t+1)}).$$

From (3) and the assumption "sufficiently isolated," $x_{i\pm1}^{(t+1)} = \varphi_{n,i\pm1}^{(t+1)} = x_{i\pm1}^{(t)}$ are also satisfied in *SL*. Then from (1),

$$\begin{aligned} x_i^{(t+1)} &= f(x_{i-1}^{(t)}, x_i^{(t)}, x_{i+1}^{(t)}) \\ &= x_i^{(t)}. \end{aligned}$$

Although pattern formations in sequences of extrapolated ECA and Life-like CA were already demonstrated in Kayama, 2016, some typical examples are presented in the following subsections.
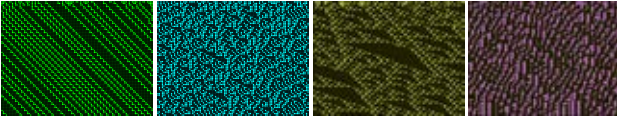
Figure 1: Patterns in [#134]: R1, R2, R3, R4 (left to right).
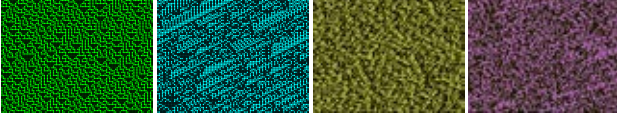
Figure 2: Patterns in [#30]: R1, R2, R3, R4 (left to right).

Figure 4: Patterns in [#22]: R1, R2, R3, R4 (left to right).

Figure 5: Patterns in [#110]: R1, R2, R3, R4 (left to right).

Extrapolated ECA

ECA is the simplest nontrivial CA with $r = 1$; its $2^3 = 8$ different neighborhood configurations result in $2^8 = 256$ possible rules. We follow the standard naming convention invented by Wolfram (Wolfram, 1983, 2002), which assigns each ECA rule a number from #0 to #255. The equivalency of the CA rules under mirror and complementary transformations reduces the number of independent rules to 88 (Li and Packard, 1990, and Kayama, 2011). In the simulations used in this subsection, we set the maximum value of R to 20. Among the sequences generated from independent ECA rules, eight are based on class I rules and all of the rules contained in these sequences also belong to class I. In contrast, various pattern formations can be found in sequences based on class II rules. The sequence [#134] shows changes between periodic and chaotic patterns depending on the index R (Fig. 1), where the colored dots are live cells and the black ones are dead. The patterns originate from pseudo-randomly generated initial configurations. The initial probability of live cells is set to 0.5. Pattern formations in sequences based on class III and IV rules are also attractive. Class III rules are sometimes exemplified by rule #30, and its sequence [#30] exhibits chaotic patterns (Fig. 2). Especially, the patterns with large R values (#30R3 and R4) are typical ones. For example, the pattern of #90R2 in Fig. 3a cannot be distinguished from them. But the pattern of rule #18 is sparser (Fig. 3b). Some sequences show periodic changes between periodic and chaotic patterns depending on the even-odd parity of R, e.g., [#22] (Fig. 4) and [#110] (Fig. 5). In these cases, no simple correlations are found between fluctuation control and the radius R, even when the amount

of information each cell acquires increases monotonically with R.

Extrapolated Life-like CA

In the descriptions below, all Life-like CA rules are specified in the Golly/RLE format (Adamatzky (Ed.), 2010, and Eppstein, 2010). The Game of Life is denoted by B3S23 in this notation, where "B" stands for "birth" and "S" stands for "survival." In the Game of Life, many complex patterns and activities can emerge (Callahan, 1995, and Flammenkamp, 1998). After a long transient process, a randomly generated initial configuration is transferred to a rest state that can include various patterns: still lifes (e.g., blocks, beehives, or ships) and oscillators (e.g., blinkers, toads, or beacons). Any isolated still life is also still life in any extrapolated rule, as proved above. Figure 6a shows the rest states of [B3S23]. In contrast, [B23S234] has a stable state only in B23S234R1 and random states in all others (Fig. 6b). Their randomness gradually increases with R. These contrasting examples are investigated in the mean field analysis in the next section.

Mixing Cells with Different R

Thus far, only homogeneous CAs were discussed, in which all the cells follow the same rule with the same value of R. If we note that the extra radius R is an index of the amount of
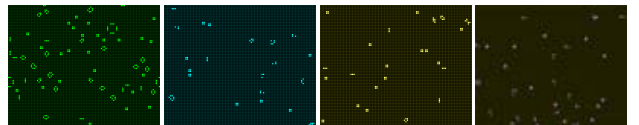
(a) [B3S23]: R1, R2, R3, R4 (left to right).
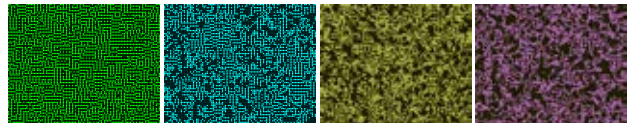
(b) [B23S234]: R1, R2, R3, R4 (left to right).

Figure 6: Patterns in homogeneous (a) [B3S23] and (b) [B23S234]

(a) Left: #90R1, right: #90R2   (b) Left: #18R1, right: #18R2

Figure 3: Patterns in (a) [#90] and (b) [#18].

95

(a) B3S23R1⇒R2⇒R3⇒R4
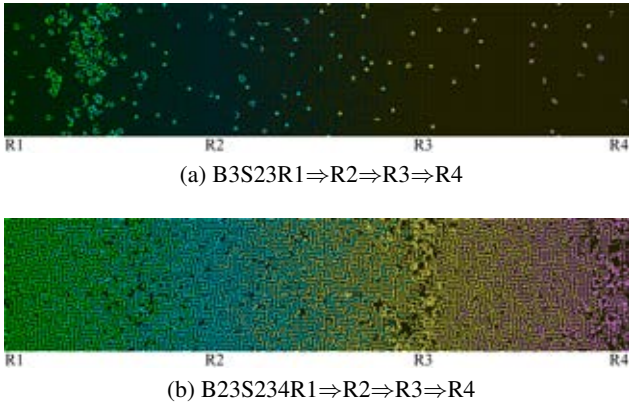
(b) B23S234R1⇒R2⇒R3⇒R4

Figure 7: Mixings of cells with different R in (a) [B3S23] and (b) [B23S234]
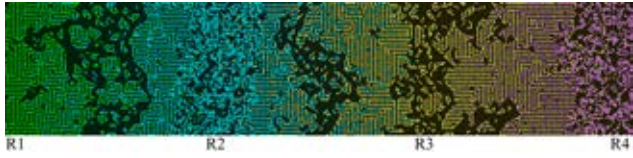
Figure 8: Mixing of cells with different R in [B4S1234]: R1⇒R2⇒R3⇒R4.

information each cell can acquire, R can be recognized as a characteristics of a cell and heterogeneous CA composed of cells with different values of R become meaningful. Figures 7a and 7b present the mixing of cells with different values of R in [B3S23] and [B23S234], respectively. The mixing ratio between two kinds of cells, Rn:R($n + 1$), changes linearly from 1:0 to 0:1. Although homogeneous B3S23R1 and R2 have rest states as shown in Fig. 6a, Fig. 7a shows an emergence of unstable states in their intermediate mixing area. In contrast, Fig. 7b shows that all mixing areas of [B23S234] become stable. The difference between them is also discussed in the next section.

A heterogeneous CA in [B4S1234] is also interesting. Figure 8 shows a complex pattern change; areas of islands and random states are sandwiched between walls. Their mixing ratio is the same as in the case of Fig. 7, but the initial probability of live cells is set to 0.2.

Mean Field Analysis

Mean field analysis assumes that the iterative application of a rule does not introduce correlations between states of cells in different positions when applied to CA (Wolfram, 1983, Schulman and Seiden, 1978). This assumption is generally not valid, but allows the derivation of a simple formula for investigating the qualitative behavior of CA dynamics and an estimation of the limit density of the possible states of a cell. Mean field analysis is applied to the contrasting examples of [B3S23] and [B23S234] presented in the previous

(a) B3S23R1

(b) B3S23R4
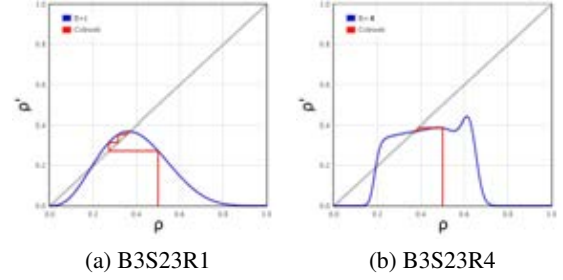
Figure 9: CobWeb plots of [B3S23]

section.

Homogeneous CA

A Life-like CA rule is determined by the numbers of "births" and "survivals." Here they are symbolized by *bth* and *svl*, respectively. The time evolution relationship (Eq. 2) is rewritten in the mean field analysis as follows:

$$\rho'_{R_0} = (1 - \rho_{R_0})B(\rho'_{e\,R_0 - 1}, bth) + \rho_{R_0}S(\rho'_{e\,R_0 - 1}, svl), \tag{9}$$

where ρ and ρ' are the densities of the live cells at the present and at the next timestep, respectively. $\rho'_{e\,R_0 - 1}$ is the estimated density of live cells relating to $\varphi_{R_0 - 1}^{(t+1)}$. The functions B and S refer to the contributions from the eight neighboring cells according to the rule. The recursive expressions, Eqs. 3 and 4, lead to

$$\rho'_{e\,R_0 - j} = (1 - \rho_{R_0})B(\rho'_{e\,R_0 - j - 1}, bth) + \rho_{R_0}S(\rho'_{e\,R_0 - j - 1}, svl). \tag{10}$$

where $j = 1, 2, \ldots, R_0 - 1$. Because the termination conditions (Eqs. 5) give $\rho'_{e0} = \rho_{R_0}$, the above Eq. 10 leads to

$$\rho'_{e1} = (1 - \rho_{R_0})B(\rho_{R_0}, bth) + \rho_{R_0}S(\rho_{R_0}, svl). \tag{11}$$

A recursive use of Eq. 10 and Eq. 11 in Eq. 9 derive a relational expression between ρ'_{R_0} and ρ_{R_0}. In B3S23, the Game of Life , the functions B and S are expressed by

$$
\begin{aligned}
B(\rho, 3) &= \binom{8}{3}\rho^3(1 - \rho)^5, \\
S(\rho, 23) &= B(\rho, 3) + \binom{8}{2}\rho^2(1 - \rho)^6.
\end{aligned}
$$

Mean field diagrams and cobweb plots of B3S23R1 and R4 are presented in Fig. 9, where the number of timesteps for reaching the limit density from the initial density 0.5 decreases with the value of R. This result means that there can be a positive correlation between the average convergence
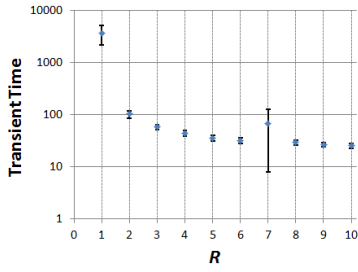
Figure 10: Semi-log plots of transient times in [B3S23]. Each error bar indicates the standard deviation.

(a) B23S234R1　　　　(b) B23S234R4
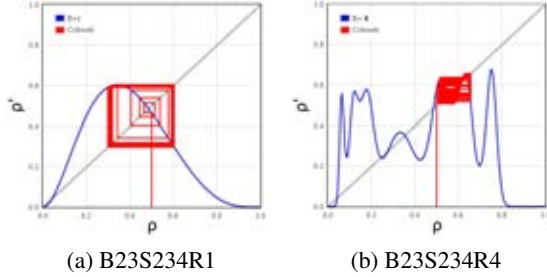
Figure 11: CobWeb plots of [B23S234]

speed and the value of R, namely the size of the perception area. Actually, Fig. 10 shows that the average transient time from fifty pseudo-randomly generated initial configurations on a 200×160 lattice to rest states decreases[1]. In contrast, the mean field diagrams and the cobweb plots of [B23S234] become complex with the value of R. Accordingly, the cell states become unstable (Fig. 7b). In other words, the REN algorithm in [B3S23] consumes information of cell states efficiently to control the fluctuations of the states, but that in [B23S234] disturbs the cell states.

Heterogeneous CA

As pointed out in Section 3.3, the pattern formations in the heterogeneous CAs in [B3S23] and [B23S234] are contrasting. In order to discuss them through mean field analysis, some geometrical symmetry is required to the arrangement of cells with different R.

The definition (i) of REN results in a specific composite configuration of cells. If an R2 cell is surrounded by eight R1 cells (Fig. 12a), the estimation of the states of the neighboring cells from the center cell is always correct, which means that the nine cells can be considered as one composite with 2^9 states. In the symmetrical arrangement of Fig. 12b, all R1 cells have seven R1 and one R2 neighbors, and all R2 cells have eight R1 neighbors. Then these two kinds of cells can be represented by two densities of cells ρ_1 and

----

[1]The long transient time at $R = 6$ is the only exception owing to a glider (Kayama, 2016).

(a)　　　　(b) R1:R2=8:1　　　　(c) R1:R2=4:1
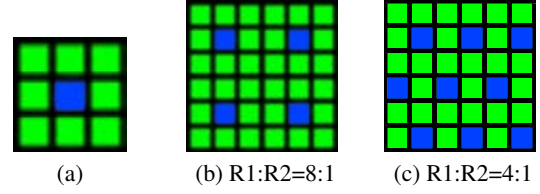
Figure 12: (a) Composite configuration comprising one center cell ($R = 2$; blue) and eight neighboring cells ($R = 1$; green), and its symmetrical arrangements; the ratios of the numbers of R1 and R2 cells are (b) 8:1 and (c) 4:1.

ρ_2 in the mean field analysis, respectively. ρ_1 satisfies the following expression:

$$\rho'_1 = (1 - \rho_1)B(\rho_1, \rho_2, bth) + \rho_1 S(\rho_1, \rho_2, svl), (12)$$

where the termination conditions are taken into account. B and S are expressed in [B3S23] as follows:

$$
\begin{aligned}
B(\rho_1, \rho_2, 3) &= \binom{7}{2}\rho_1^2(1 - \rho_1)^5\rho_2 \\
&\quad + \binom{7}{3}\rho_1^3(1 - \rho_1)^4(1 - \rho_2), \\
S(\rho_1, \rho_2, 23) &= B(\rho_1, \rho_2, 3) + \binom{7}{1}\rho_1(1 - \rho_1)^6\rho_2 \\
&\quad + \binom{7}{2}\rho_1^2(1 - \rho_1)^5(1 - \rho_2).
\end{aligned}
$$

The expression of ρ_2 comes from Eq. 9 as follows:

$$\rho'_2 = (1 - \rho_2)B(\rho_{e1}, bth) + \rho_2 S(\rho_{e1}, svl), \quad (13)$$

where the density of the estimated state ρ_{e1} is identical with ρ'_1, because the estimation of the states of the neighboring R1 cells from the R2 cell is always correct. The numerical results of iterations of the above expressions 12 and 13 in [B3S23] lead to Fig. 13, which shows that the symmetrical arrangement of Fig. 12b has no essential differences with the homogeneous B3S23R2. In contrast, the plots of the numerical results of the above expressions of the arrangement of B23S234R1 and R2 cells and homogeneous B23S234R2 are totally different. Correlation between R1 and R2 cells suppresses their fluctuation mutually and their densities approach a limit value. These results correspond to pattern formations in the symmetrical arrangements of B3S23R1 and R2 cells and B23S234R1 and R2 cells. The former reaches a rest state and the latter a stable one.

In order to understand the emergence of unstable states in the mixing area between B3S23R1 and R2 in Fig. 7a, we adopt another symmetrical arrangement of Fig. 12c. Actually, the pattern formation in this arrangement is unstable

(a)   (b)
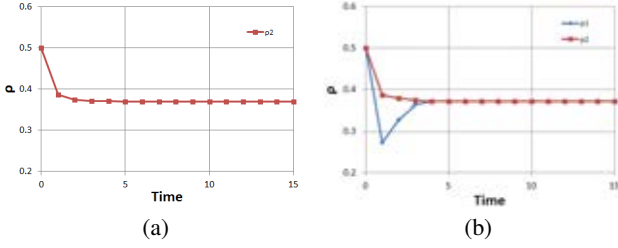
Figure 13: Plots of iterations in (a) homogeneous B3S23R2 and (b) composite of R1:R2=8:1

(a)   (b)
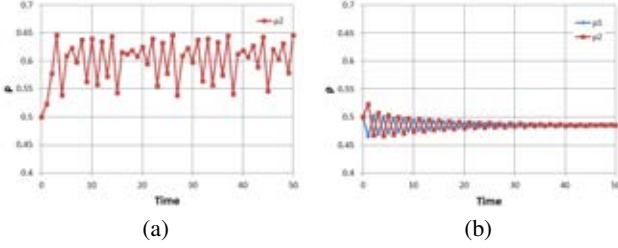
Figure 14: Plots of iterations in (a) homogeneous B23S234R2 and (b) arrangement of B23S234R1:R2=8:1

(Fig. 15a). The arrangement requires two kinds of R1 cells and one R2 cell. If the two kinds of R1 cells are represented by Ra and Rb, they are distinguished by their neighbors; Ra cell has six Rb and two R2 neighbors, and Rb cell has three Ra, two Rb and three R2 neighbors. R2 cell has two Ra and six Rb neighbors. If the densities of the three types of cells are denoted by ρ_a, ρ_b, and ρ_2, they satisfy the following expressions:

$$\rho'_a = (1-\rho_a)B_a(\rho_b,\rho_2,3) + \rho_a S_a(\rho_b,\rho_2,23), \quad (14)$$
$$\rho'_b = (1-\rho_b)B_b(\rho_a,\rho_b,\rho_2,3) + \rho_b S_b(\rho_a,\rho_b,\rho_2,23), \quad (15)$$
$$\rho'_2 = (1-\rho_2)B_2(\rho_a,\rho_b,3) + \rho_2 S_2(\rho_a,\rho_b,23), \quad (16)$$

(a)   (b)
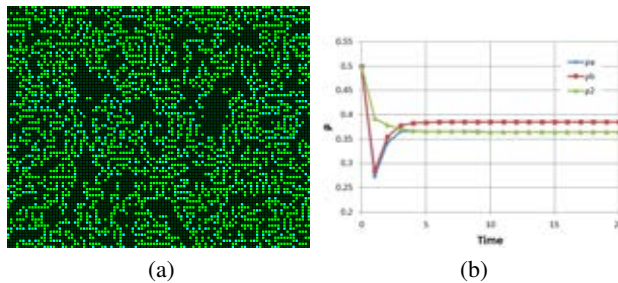
Figure 15: (a) Unstable pattern in the symmetrical arrangement (Fig. 12c) of B3S23R1:R2=4:1 composites and (b) its plot of iterations.

where the above functions, Bs and Ss, are presented in Table 1. The plot of the numerical results is shown in Fig. 15b. The existence of a gap between ρ_a and ρ_b represents the existence of the unstable area.

From the above discussion, mean field analysis can be effective to investigate the qualitative behavior of heterogeneous CA, if some symmetrical arrangements of the composites are adopted.

Conclusions

The REN algorithm allows for defining the intelligent process of each cell by introducing an extra index R that represents the radius of the perception area of a cell in addition to the radius of the CA neighborhood. A basic CA rule with a unit rule radius is extrapolated into rules with larger radii $r = 2 \times R + 1$, which form a sequence indexed by R containing the basic CA as the first term of the sequence. Pattern formations in some typical sequences of the extrapolated ECA and Life-like CA rules are presented. It is proven that a still life in the basic CA is also a still life in any extrapolated rule over the same sequence when the still life is sufficiently isolated. The sequence of the Game of Life ([B3S23]) and [B23S234] exhibit the contrasting activities of cell states, which are discussed through mean field analysis. Those mean field diagrams show opposite effects of their extrapolations in accordance with R. Namely, the REN algorithm in [B3S23] consumes information about the cell states efficiently in order to control the fluctuations of the states, but that in [B23S234] disturbs the cell states. The pattern activities of the heterogeneous models of mixing R1 and R2 cells in [B3S23] and [B23S234] are also contrasting. Some symmetrical arrangements of the composites containing R2 cells surrounded by eight R1 cells are examined to discuss the models in the mean field analysis. The unstable area between B3S23R1 and R2 may be emerged from the gap between the limit densities of two kinds of R1 cells. The stable area between B23S234R1 and R2 comes from mutually suppressing their fluctuations. Correspondingly, mean field analysis of their densities shows that they approach to the same limit value. Furthermore, mean field analysis is effective in investigating qualitative behavior of heterogeneous CA, not just homogeneous CA.

The new perspective of CA presented here has several different potential applications. Heterogeneous CAs with combinations of cells with different values of R show unexpected pattern activities. Such phenomena appear to act like mixing two different materials; the end material has a changed state by the chemical process. The symmetrical arrangements of composites could be related to crystal substances in solid-state physics. If interactions between cells and an evolutionary algorithm are introduced in the heterogeneous CA, a new theoretical field, like "CA Chemistry," could be established just as the boids theory was developed to "Swarm Chemistry" (Sayama, 2007, 2009, 2010).

98

$$B_a(\rho_b, \rho_2, 3) = \binom{6}{1}\rho_b(1-\rho_b)^5\rho_2^2$$
$$+\binom{6}{2}\rho_b^2(1-\rho_b)^4\binom{2}{1}\rho_2(1-\rho_2) + \binom{6}{3}\rho_b^3(1-\rho_b)^3(1-\rho_2)^2$$

$$S_a(\rho_b, \rho_2, 23) = B_a + (1-\rho_b)^6\rho_2^2$$
$$+\binom{6}{1}\rho_b(1-\rho_b)^5\binom{2}{1}\rho_2(1-\rho_2) + \binom{6}{2}\rho_b^2(1-\rho_b)^4(1-\rho_2)^2$$

$$B_b(\rho_a, \rho_b, \rho_2, 3) = (1-\rho_a)^3(1-\rho_b)^2\rho_2^3$$
$$+(1-\rho_a)^3\binom{2}{1}\rho_b(1-\rho_b)\binom{3}{2}\rho_2^2(1-\rho_2)$$
$$+\binom{3}{1}\rho_a(1-\rho_a)^2(1-\rho_b)^2\binom{3}{2}\rho_2^2(1-\rho_2)$$
$$+\left\{(1-\rho_a)^3\rho_b^2 + \binom{3}{2}\rho_a^2(1-\rho_a)(1-\rho_b)^2\right\}\binom{3}{1}\rho_2(1-\rho_2)^2$$
$$+\binom{3}{1}\rho_a(1-\rho_a)^2\binom{2}{1}\rho_b(1-\rho_b)\binom{3}{1}\rho_2(1-\rho_2)^2$$
$$+\left\{\binom{3}{1}\rho_a(1-\rho_a)^2\rho_b^2 + \rho_b^3(1-\rho_b)^2\right\}(1-\rho_2)^3$$
$$+\binom{3}{2}\rho_a^2(1-\rho_a)\binom{2}{1}\rho_b(1-\rho_b)(1-\rho_2)^3$$

$$S_b(\rho_a, \rho_b, \rho_2, 23) = B_b + (1-\rho_a)^3(1-\rho_b)^2\binom{3}{2}\rho_2^2(1-\rho_2)$$
$$+(1-\rho_a)^3\binom{2}{1}\rho_b(1-\rho_b)\binom{3}{1}\rho_2(1-\rho_2)^2$$
$$+\binom{3}{1}\rho_a(1-\rho_a)^2(1-\rho_b)^2\binom{3}{1}\rho_2(1-\rho_2)^2$$
$$+\left\{(1-\rho_a)^3\rho_b^2 + \binom{3}{2}\rho_a^2(1-\rho_a)(1-\rho_b)^2\right\}(1-\rho_2)^3$$
$$+\binom{3}{1}\rho_a(1-\rho_a)^2\binom{2}{1}\rho_b(1-\rho_b)(1-\rho_2)^3$$

$$B_2(\rho_a, \rho_b, 3) = (1-\rho_a)^2\binom{6}{3}\rho_b^3(1-\rho_b)^3$$
$$+\binom{2}{1}\rho_a(1-\rho_a)\binom{6}{2}\rho_b^2(1-\rho_b)^4 + \rho_a^2\binom{6}{1}\rho_b(1-\rho_b)^5$$

$$S_2(\rho_a, \rho_b, 23) = B_2 + (1-\rho_a)^2\binom{6}{2}\rho_b^2(1-\rho_b)^4$$
$$+\binom{2}{1}\rho_a(1-\rho_a)\binom{6}{1}\rho_b(1-\rho_b)^5 + \rho_a^2(1-\rho_b)^6$$

Table 1: Contribution functions Bs and Ss in mean field analysis for the symmetrical arrangement of B3S23R1 and R2 cells (Eqs. 14-16).

Acknowledgments

References

Adamatzky, A., editor (2010). *Game of life cellular automata*. London:: Springer.

Banks, A., Vincent, J., and Anyakoha, C. (2007). A review of particle swarm optimization. part i: background and development. *Natural Computing*, 6.4:467–484.

Berlekamp, E. R., Conway, J. H., and Guy, R. K. (1982). *Winning Ways for Your Mathematical Plays*. Academic, New York.

Callahan, P. (1995). Patterns, programs, and links for conway's game of life. "http://www.radicaleye.com/lifepage/". Retrieved at February 1, 2011.

Eppstein, D. (2010). Growth and decay in life-like cellular automata. In Adamatzky, A., editor, *Game of Life Cellular Automata*, pages 71–98. Springer.

Flammenkamp, A. (1998). Achim's game of life. "http://wwwhomes.uni-bielefeld.de/achim/gol.html". Retrieved at December 12, 2011.

Gardner, M. (1970). Mathematical games. *Scientific American*, 223:102–123.

Kayama, Y. (2011). Network representation of cellular automata. In *2011 IEEE Symposium on Artificial Life (IEEE ALIFE 2011) at SSCI 2011*, pages 194–202.

Kayama, Y. (2016). Extension of cellular automata by introducing an algorithm of recursive estimation of neighbors. In *Proceedings of the 21-st International Symposium on Artificial Life and Robotics*, pages 73–77.

Li, W. and Packard, N. (1990). The structure of the elementary cellular automata rule space. *Complex Systems*, 4:281–297.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM Siggraph Computer Graphics*, 21.4:25–34.

Sayama, H. (2007). Decentralized control and interactive design methods for large-scale heterogeneous self-organizing swarms. *Advances in Artificial Life*, 15(1):105–114.

Sayama, H. (2009). Swarm chemistry. *Artificial Life*, 15.1:105–114.

Sayama, H. (2010). Robust morphogenesis of robotic swarms. *Computational Intelligence Magazine, IEEE*, 5(3):43–49.

Schulman, L. S. and Seiden, P. E. (1978). Statistical mechanics of a dynamical system based on conway's game of life. *Journal of Statistical Physics*, 19(3):293–314.

von Neumann, J. (1966). The theory of self-reproducing automata. In Burks, A. W., editor, *Essays on Cellular Automata*. University of Illinois Press.

Wolfram, S. (1983). Statistical mechanics of cellular automata. *Rev. Mod. Phys.*, 55:601–644.

Wolfram, S. (1984). Universality and complexity in cellular automata. *Physica D*, 10:1–35.

Wolfram, S. (1986). *Theory and Applications of Cellular Automata*. World Scientific, Singapore.

Wolfram, S. (2002). *A New Kind of Science*. Wolfram Media, Inc.

Ant Geometers

Sean Luke, Katherine Russell, and Bryan Hoyle

George Mason University
sean@cs.gmu.edu

Abstract

Just how much can a pheromone-enabled swarm do? Motivated by robotic construction, we set out to show that a swarm of computationally simple ants, communicating only via pheromones, can in fact perform classic compass-straightedge geometry, and thus can make many shapes and perform many nontrivial geometric tasks. The ants do not need specially-designed stigmergic building materials, a prepared environment, local or global direct communication facilities (such as radio or line-of-sight signaling), or any localization beyond initial starting points for drawing. We describe the proof of concept in replicable detail. We then note that its accuracy and efficiency can be greatly improved through augmentation with a simple embeddable broadcast mechanism.

Introduction

One of the biggest difficulties in swarm robotics, and in swarm agent simulation in general, lies in how to communicate and coordinate. Due to their large numbers, swarm agents often cannot communicate through a common broadcast medium such as radio, both because they would overwhelm the medium, and because it would require every agent to receive and deal with messages from all N other agents. Instead artificial swarm schemes often use either local communication or *indirect* communication, whereby agents leave messages for one another — virtual breadcrumbs, if you will.

The biggest source of inspiration for indirect communication in robotics is surely the use of pheromones by ants, termites, etc. to coordinate behaviors. These insects are known to use pheromones for many tasks: but most swarm robotics and swarm simulation literature has focused on their most famous use, namely establishing foraging trails.

In prior work we have demonstrated trail optimization, adaptation to environmental changes, and even multi-waypoint, self-intersecting tours using only pheromones and swarms of very simple agents (Panait and Luke, 2004). Our later work extended the pheromone model to swarm foraging robots which store and read pheromone information in intelligent breadcrumbs (in the form of wireless sensor motes), and then deploy, move, and retrieve these devices from the environment (Hrolenok et al., 2010; Russell et al., 2015).

The use of pheromones to build foraging trails is straightforward and well studied. We are instead interested in showing that pheromones can be used for something much more ambitious. Our research area is collective building construction, and among the first tasks in construction is the laying out of survey lines to define the form of the object being built. As such, we have chosen to demonstrate, as an elaborate proof of concept, that swarms of very simple agents, communicating only via pheromones, can achieve all the operations necessary to perform collective **compass-straightedge construction** (or "classical construction") from geometry, and thus can build many nontrivial geometric shapes.

This is not easy: some of the basic operations are challenging to achieve and, while we show that such operations are possible, they can be costly. We will compare against agents that differ only in their communications medium (broadcast beacons), but nevertheless can do the task much more rapidly.

In this paper we first discuss existing literature in pheromones, swarm robotics, and collaborative construction. We then explain classic compass-straightedge geometric construction and its background. We then detail the pheromone and swarm agent model being used, and describe the basic procedures necessary to do compass-straightedge construction. Finally, we compare this approach against similar agents using broadcast beacons instead of pheromones.

Previous Work

Swarm robotics and swarm agent research is naturally inspired by social insects (Brambilla et al., 2013) and stigmergic approaches to collective behavior Dorigo et al. (2000). Swarms are highly parallel, can be built with simple agents, and are robust in the face of noise, agent failure, and dynamic environments (Bonabeau, 1996). Swarms are commonly used in tasks such as exploration and foraging (Wodrich and Bilchev, 1997; Panait and Luke, 2004; Russell et al., 2015; Prabhakar et al., 2012), but one recent application has been in collective construction (Ardiny et al., 2015). Swarms are typically limited to indirect, stigmergic, and local communication, which leads to one of the three following implementation trends:

First: agents may use inert, local, environmental features, which can be sensed but do not directly communicate with the agents or with each other. Landmarks or the presence of other agents are common features which can be used to clear ground for site preparation (Parker and Zhang, 2006) and build circles around given locations (Pitonakova and Bullock, 2013). Agents using these techniques do not need accurate localization and can use a variety of building materials to perform their tasks. However, the agents cannot easily do planning or coordination as they neither know where they are nor what has been accomplished so far.

Second: agents may exist in or create "smart" environments which can be used to localize the agents, such as writable blocks (Allwright et al., 2014; Werfel and Nagpal, 2008; Sugawara and Doi, 2014) or countable building materials (Werfel et al., 2011). This allows grid-world models to be directly implemented with robots and has produced swarms capable of making 3D user-defined shapes. The use of such environments allows agents to be fully localized relative to the structure they are building, but they must use highly specialized building materials or contrived environments.

Third: agents may lay temporary stigmergic markers in the environment, such as breadcrumbs or pheromones (Deneubourg et al., 1990; Russell, 1999; Panait and Luke, 2004; Chibaya and Bangay, 2007). This technique has produced behaviors such as circle building (Pitonakova and Bullock, 2013), exploration (Wodrich and Bilchev, 1997), and wall building (Stewart and Russell, 2006). One difficulty with the method lies in the medium in which these stigmergic markers are placed. In prior work we have attempted to address this with portable local beacons (Russell et al., 2015) but other methods, such as RFID tags (Mamei and Zambonelli, 2007; Ziparo et al., 2007), lights (Stewart and Russell, 2006), and chemical dispersion (Kowadlo and Russell, 2008) have also been tried.

Our work in this paper is most comparable to that of "smart" environments, such as in Werfel et al. (2011), as it allows the collective construction of a very large number of possible structures: but instead of using specially-made stigmergic construction materials, we explore whether such tasks may be performed solely through a general-purpose indirect communication method such as pheromones.

Compass-Straightedge Construction

The *compass-straightedge* technique has existed since the ancient Greeks and has a long history of constructive proofs to build complex shapes and do many nontrivial geometric tasks. Traditionally the only two tools permitted are a collapsing compass and an arbitrarily long straightedge. The compass loses its angle as soon as it is lifted off the drawing surface, and so one cannot preserve distance by raising the compass and moving it somewhere else. However, this is an artificial limitation, as there is a way to transfer distance between two points using a finite number of axiomatic steps (Sarhangi,
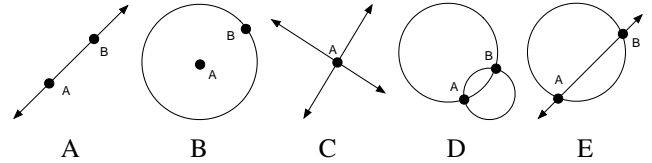
Figure 1: The five basic compass-straightedge procedures. (A) Drawing a line between two points. (B) Drawing a circle centered at one point and passing through another. (C) Identifying the point at the intersection of two lines. (D) Identifying and distinguishing the two points at the intersection of two circles (one point if they are tangent). (E) Identifying and distinguishing the two points at the intersection of a circle and a line (one point if they are tangent).

2007). It has also been shown that the straightedge is not required at all, only a compass, (Mohr, 1672; Mascheroni, 1797) but the resulting proofs can be much more complicated.

Figure 1 shows the five basic abstract procedures sufficient to do all compass-straightedge construction: drawing a line through two points, drawing a circle centered at one point and passing through another, identifying the intersection of two lines, identifying the intersections of a line and a circle, and identifying the intersections of two circles. Note that for the last two procedures, not only must one identify the points, but one must also *uniquely* identify and distinguish them from one another. This problem does not arise for human proofs, which are visual: but ants have only local information and so must include ways to distinguish the points.

Composing these simple techniques, one can build much more complicated constructions. One simple example for constructing an equilateral triangle is as follows: start with a line segment representing the desired base; use a compass to construct two circles centered at the endpoints with radius equal to the segment length; and, finally, draw two more line segments connecting the desired intersection point to the two endpoints of the base. Other basic things which can be constructed include: bisecting arbitrary angles with a line; constructing a square with twice the area of another square; constructing a circle tangent to another circle; trisecting an arbitrary line segment; and building any regular polygon whose number of edges is equal to some power of 2 times the product of zero or more primes of the form $2^{2^n} + 1$, for some integer n; and many, many more. Overall the first ten constructible polygons have 3, 4, 5, 6, 8, 10, 12, 15, 16, and, thanks to Gauss, 17 sides (Gauss, 1801).

There are many things that *cannot* be constructed: any regular polygon not of the form above (such as the heptagon or the nonagon), trisections of arbitrary angles, squares with equal area to arbitrary circles (the legendary *squaring the circle*), and many other classes of shapes. Several extensions have been proposed: for example, the ability to make marks on the straightedge permits angle trisection and the construction of additional regular polygons (Gleason, 1988).

A: 10%   B: 1%   C: 0.1%   D: 0.01%
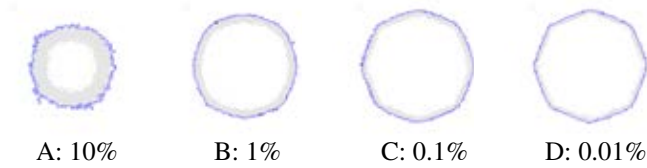
Figure 2: Effect of Evaporation on Circle Development. Too high evaporation (A) produces incorrectly small and overly noisy circles, but too low evaporation (D) produces octagons.

The Ant Model

Out ants' world is a non-toroidal, bounded 200×200 square grid environment holding 1000 ants. Any number of ants may share the same grid cell, and can move horizontally, vertically, or diagonally. Each grid cell can also hold multiple kinds of *pheromones*. The ants may move about in the environment, may read pheromone values in their local (9-cell) neighborhood, and may write or update pheromone values only to their current cell. As shown later, a grid world is not critical to the model, but was chosen for simplicity.

Pheromones are used by the ants for various tasks: to mark points of interest, to establish gradients to and from those points of interest, to build up estimates of shapes, and to make final line drawings in the environment. Each pheromone has a *pheromone type* and a current numerical *value* ≥ 0 which by default automatically reduces (*evaporates*) at a rate of 0.5% per timestep. A pheromone in a cell can also be set to be *non-evaporating*. Non-evaporating pheromones are used to draw lines and circles and to mark points of interest, which in turn serve as permanent maxima in a pheromone gradient. We call cells holding non-evaporating pheromones *sites*.

Because we ultimately will migrate this model to robots, the pheromones in the model do not diffuse on their own, but rather pheromone information can only be spread from one cell to another by an ant. This is because, while diffusion is used in biological pheromone models, it is not easy to employ with physical robots, as it requires chemical sensors and dispersion methods (Kowadlo and Russell, 2008), or cells or breadcrumbs which communicate with one another.

Lacking diffusion, evaporation becomes very useful for a grid-world model as it causes the ants to naturally build gradients with more circular and less octagonal cross-sections. This results in straight paths at any angle, not just in the eight compass directions, and more circular circles (see for example Figure 2). Additionally, evaporation can be seen as a potential benefit in more interesting dynamic environments where old information should be treated skeptically. The disadvantage of evaporation is that it is a major source of noise: agents cannot rely on pheromone gradients being consistent at infrequently visited locations as neighboring cells may have been allowed to evaporate while the current cell has just recently been "topped off". This significantly complicates our task and makes our procedures less efficient.

The Ants Each ant is a simple machine which iteratively *updates* pheromone values in its cell, then either follows a *procedure* to perform some task, or (with 0.1 probability) *moves randomly*. Random movement encourages exploration and nondeterminism: some procedures may temporarily disable it so as to reduce noise.

An ant can move in any of the eight compass directions. If an ant moves horizontally or vertically, it must wait one timestep before it may continue. If an ant moves diagonally, it must wait 1.5 steps (a discrete approximation of $\sqrt{2}$, though neither has any real impact on the results over just using 1.0).

An ant can also sense pheromones in its grid cell or any of the eight neighboring cells, and can determine if those pheromones are set to evaporate or not. An ant also knows where it was last timestep, and can perceive relative orientation ("to the left of me", etc.). An ant can set a temporary *timer* to know roughly how long it has been doing a task. Finally, an ant is capable of storing a single pheromone value in its sole *register* for later retrieval.

Updating Pheromones Every timestep, the ant first updates all the pheromones at its cell location. If a pheromone P at the ant's location i is marked as non-evaporating, the ant leaves it alone. Otherwise, it is updated as:

$$P_i \leftarrow \max_{j \,\in\, \text{Neighbors}(i)} \begin{cases} \alpha^{\sqrt{2}} P_j & \text{if } j \text{ is a diagonal neighbor of } i \\ \alpha P_j & \text{otherwise} \end{cases}$$

We set α to 0.88 based on experiment. As the ants wander about, this equation effectively builds a pheromone gradient away the "tent-pole" locations in the grid (where a non-evaporating value for P has been set). Note that even though $\alpha^{\sqrt{2}}$ is used to (properly) cut down diagonal neighboring cells, this is not sufficient to create a true circular gradient in a grid world.

The Procedures

As this paper is a proof of concept of a challenging task, and the procedures below are nontrivial, we describe them in detail for replicability, and beg the reader's forgiveness.

Though we describe the ant's procedures below in algorithmic pseudocode, we in fact implement each of them as a finite-state automaton (a DFA) which iterates through its process one step at a time as it is pulsed. Each state in the automaton is associated with some behavior to iteratively perform, which may be some simple action (like laying a pheromone) or a call to a lower-level procedure (such as wall following). Such a recursive DFA is known as a *Hierarchical Finite-State Automaton* (HFA). Any procedure can signal *done*, which informs its calling procedure that it has finished.

The procedures make heavy use of building gradients from various points in order to establish loci. They take two kinds of arguments: *pheromones*, which are capitalized (like *PointA*), and simple numerical *values*, which are lowercase (like *direction* or *m*). A pheromone typically establishes

102

a gradient leading to a maximal spot in the environment. For brevity, we often use the pheromone name (like *PointA*) to also refer to the location of its maximum (typically a *site*). There is always a single starting location for the ants: *Home*.

Some procedures assume a world border, however in an environment with no border it could just as well be replaced with following along the locus of points for which the *Home* gradient is equal to some very small value. Only cells within the world border are considered valid places to move and read pheromones from.

We first describe various basic procedures, then the five geometric procedures. Here is a summary of the five:

- **Draw a Circle**   Draw a line segment from the center to the edge point and use it to determine the radius (in terms of pheromone gradient from the center). Build a gradient from the center. Identify points whose gradient value matches the radius. Trace a line through those points.

- **Mark Line/Line Intersection**   This is simple: fan out until the intersection is discovered, then mark it.

- **Mark Circle/Circle Intersections**   Identify the intersections *and* determine which is which. To do this, each ant randomly chooses to go to one or the other of the two circle centers. It then heads towards the other center until it finds the circle edge of its initial center, then follows clockwise along the circle until it reaches an intersection. The intersection is uniquely labeled according to which circle was followed when it was discovered. The process is illustrated below:
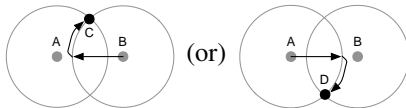
- **Mark Line/Circle Intersections**   Again, the trick is to distinguish the intersections from each other. To do this, the ants go to a certain extreme point on the line outside the circle, then follow down the line, and as they find intersections, mark them in the order they were found.

- **Draw an Extended ("Infinite") Line**   This is tricky. Noise and the restriction to local information make the obvious approaches impossible for drawing an extended line, such going down the gradient away the circle centers, or walking straight using some notion of momentum. Instead we set up a perpendicular bisector. The ants draw a circle centered at the first point and passing through the second point, then a circle centered at the second point, passing through the first. They identify the two circle/circle intersections, then draw the extended line that is the the perpendicular bisector between the two points. The various elements are shown below:
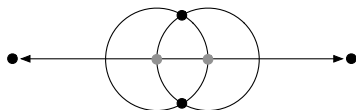
Basic Procedures

Backup( )   Back up one step, to the ant's previous location. Further backups are not possible (the ant has no history).

GoUp(*P*)   Iteratively, in the current nine-cell neighborhood around the ant, move to the cell where the value of pheromone *P* is highest. Break ties randomly. If the highest value is zero, *MoveRandomly( )*. When at a local maximum, signal *done*.

GoHome( )   The same as *GoUp(Home)*.

GoDown(*P, m (default = 0)*)   Iteratively, in the current nine-cell neighborhood around the ant, move to the cell where the value of pheromone *P* is lowest. Break ties randomly. If the lowest value is at or below *m* or the ant cannot move because the surrounding area has a higher gradient (as happens at the world border), signal *done*.

MoveRandomly( )   Move to a random location in the nine-cell neighborhood around the ant. Because our environment is bounded, we include an additional protective measure: if the ant reaches a world border, *GoHome( )*.

MarkSite(*P*)   Set the value of pheromone *P* at the ant's location to maximal, and mark it non-evaporating.

LoadRegister(*P*)   Store in the ant's register the value of pheromone *P* at the ant's location.

FollowWorldBorder(*direction*)   Temporarily turn off randomness so as not to lose the border. Head along the border of the world in *direction* (clockwise or counterclockwise).

FollowLine(*P, End*)   Temporarily turn off randomness to not to lose the line. Among the eight-cell neighborhood around the ant, iteratively move to the cell where *P* is highest. Break ties by preferring forward-facing directions. When the ant has reached a cell with the *End* pheromone, signal *done*.

WallFollow(*P*)   Head clockwise such that pheromone *P* is always maximal in the cell immediately to the ant's right (that is, follow along a "wall" of cells marked as sites for *P*).

MakeLine(*PointA, PointB*)   This procedure sets up the gradients for straight line from *PointA* to *PointB*, but does not draw it. Iterate: *GoUp(PointA)*, then *GoUp(PointB)*. This causes the ants to go back and forth between the points, optimizing the trail until it is straight. After some time (perhaps 5000 steps), signal *done*.

BuildGradientDown(*P, m (default = 0)*)   Build a gradient away from *P*, stopping when the gradient is well established down to value *m*. This is done by repeatedly iterating: *GoUp(P)*, then *GoDown(P, α× m)*. The α makes the ant go one cell further than needed. The ant initially doesn't go straight down, but makes many random moves: and so we only signal *done* when *GoDown(...)* signals *done* and *all* the neighboring cells around the ant have nonzero values for *P*, indicating that it has likely built out the gradient well.

DrawLine(*PointA, PointB, Trace*)   This procedure draws a straight line of pheromone *Trace* from *PointA* to *PointB*. First, *MakeLine(PointA, PointB)*. Then *GoUp(PointA)*. Temporarily turn off random moves to make a straight line, then *GoUp(PointB)* while calling *MarkSite(Trace)* on each new grid cell. Upon reaching *PointB*, signal *done*.

MakePerpendicularBisectorLine(*PointA, PointB, MarkA, MarkB, Temp*)   The bisector line is the locus of cells where the gradients from points *PointA* and *PointB* are equal. Its ends are defined by *MarkA* and *MarkB*. This splits the swarm into two groups to build the two gradients in parallel.

Randomly do either: (1) *GoUp(PointA)*, then *MakeBisectorHalf(PointA, PointB, MarkA, MarkB, Temp)*; or (2) *GoUp(PointB)*, then *MakeBisectorHalf(PointB, PointA, MarkB, MarkA, Temp)*. Then signal *done*.

MakeBisectorHalf(*MyPoint, OtherPoint, MyMark, OtherMark, Temp*)   This handles one sub-swarm. First, *BuildGradientDown(MyPoint)*. While doing so, when the pheromone value of *MyPoint* is less than or equal to the pheromone value for *OtherPoint*, *MarkSite(Temp)*; and whenever *Temp* is set at the ant's current location but the pheromone value of *MyPoint* is greater than the pheromone value of *OtherPoint*, remove it, as it has been set incorrectly due to pheromone evaporation.

Occasionally (with 0.1 probability) stop gradient-building and do the following. *GoDown(OtherPoint)* until the agent hits the world border, then *FollowWorldBorder(clockwise)* until one of three things happens: (1) If the value of *OtherPoint* is greater than *MyPoint*, the ant is too far: *GoUp(MyPoint)*, then continue *BuildGradientDown(MyPoint)* as before. (2) If the ant finds a cell with *Temp* set, this is the far end of the line. *MarkSite(MyMark)*, *FollowWorldBorder(counterclockwise)* a short distance (perhaps 100 steps), *GoUp(MyPoint)*, *GoUp(OtherPoint)*, *GoUp(Home)*, and *GoUp(MyPoint)*, then continue to *BuildGradientDown(MyPoint)* as before. This spreads the *MyMark* pheromone.   (3) If the ant finds a non-zero *MyMark* gradient, the task is already completed: *GoUp(MyPoint)* and continue to *BuildGradientDown(MyPoint)* as before. Whenever both *MyMark* and *OtherMark* have been set at the ant's cell, *GoUp(MyPoint)*, then *BuildGradientDown(MyPoint)*: this erases *Temp* in all cells. After some time (perhaps 1000 steps), signal *done*.

The Five Geometric Construction Procedures

DrawCircle(*Center, EdgePoint, Temp, Circle*)   This procedure draws a circle with pheromone *Circle*, centered at *Center*, and which passes through *EdgePoint*. A temporary and much thicker circle is marked first, then the outer border traced, since fluctuations in pheromones and random movements of the ants can cause variation in what the ants perceive as being the correct distance from the center.

MakeLine(Center, EdgePoint), then *GoUp(Center)*, then *GoUp(EdgePoint)*. At this point, *LoadRegister(Center)* to measure the gradient from *Center*, which will define the

radius of the circle. *GoUp(Center)*. Next, repeatedly iterate through *BuildGradientDown(Center, register)*, then *GoDown(Center)*, then *MarkSite(Temp)*. This causes the ant to mark the outer edge of the circle with *Temp*. After some time (perhaps 3000 steps), enough marks have been set to form a solid circular wall. At this point, the ant must find its way to the outside of this wall. To do this, *GoUp(Center)*, then *GoDown(Center, $\beta \times$ register)*, which causes the agent to move out well beyond circle whose radius is defined by *register*, then *GoUp(Center)* until it finds a cell with a *Temp* pheromone value (the wall). We set $\beta = 0.01$ based on experiment. Finally, trace the circle: *WallFollow(Temp)* while simultaneously doing *MarkSite(Circle)* on each new grid cell. After some time (perhaps 500 steps), signal *done*.

MarkCircleCircleIntersections(*CenterA, CenterB, TempA, TempB, CircleA, CircleB, Mark1, Mark2*)   This procedure identifies and uniquely distinguishes the intersections of two circles, one centered at *CenterA* and traced with *CircleA*, and one centered at *CenterB* and traced with *CircleB*. The *Temp* pheromones were those used to generate the original circles: they are called upon again to assist in wall-following.

First randomly *GoUp(CenterA)* or *GoUp(CenterB)*. If the ant chose *CenterA*, then *GoUp(CenterB)* until it finds *CircleA*, then *WallFollow(TempA)* until the ant reaches a cell with both *CircleA* and *CircleB*: then *MarkSite(Mark1)*. If the ant chose *CenterB*, then *GoUp(CenterA)* until it finds *CircleB*, then *WallFollow(TempB)* until the ant reaches a cell with both *CircleA* and *CircleB*: then *MarkSite(Mark2)*. In either case, *GoHome( )* and stay there for a while (perhaps 500 steps) to ensure other ants have seen the markings, then signal *done*.

MarkCircleLineIntersections(*PointA, PointB, Circle, Line, Mark1, Mark2*)   This procedure identifies and uniquely distinguishes the (up to) two intersections of a circle traced out with *Circle*, and a line delimited by *PointA* and *PointB*, and traced out with *Line*. To make things simple, we assume that *PointA* and *PointB* are at the extrema of the line: this is not a problem, as the procedure for marking extended lines will mark the points at the borders of the environment.

First *GoUp(PointA)*, then *FollowLine(Line, PointB)*, and as the ant is doing so, *MarkSite(...)* the first intersection (which has both *Line* and *Circle* pheromones) with *Mark1*, and the second such intersection, if any, with *Mark2*.

MarkLineLineIntersection(*LineA, LineB, Mark*)   This identifies the intersection between two lines. One simple (and inefficient) approach is to search the space until we have discovered the unique intersection, then mark it.

GoUp(Mark) (which moves randomly unless *Mark* is non-zero) until the agent has found a cell marked as both *LineA* and as *LineB*: at this point, *MarkSite(Mark)* then *GoHome( )*, then signal *done*. If another ant finds the intersection first, we may see the *Mark* pheromone already, in which case *GoHome( )* to tell the other ants, then signal *done*.
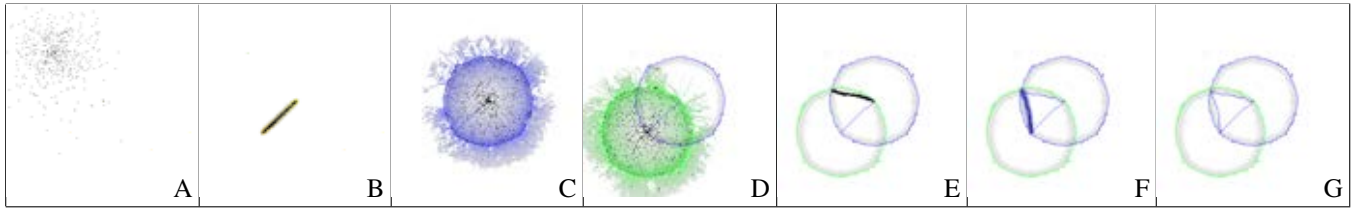
Figure 3: Building an Equilateral Triangle from Two Provided Points. (A) Emerging from Home. (B) Building the first line between the provided points. (C) Building the circle. (D) Building the second circle. (E) Determining the intersections, then building the second line. (F) Tracing the third line. (G) Finished.

DrawExtendedLine(*PointA, PointB, TempA, TempB, CircleA, CircleB, IntersectionA, IntersectionB, MarkA, MarkB, Line*) This procedure requires *nine* distinct pheromones because of its approach to drawing an extended (arbitrarily long) line between two points *PointA* and *PointB*. It first draws circles with *PointA* and *PointB* each as centers and passing through the other point, respectively. It then identifies the intersections of these circles. The extended line is the perpendicular bisector of these two intersection points.

The procedure is as follows. First *DrawLine(PointA, PointB, Line)*. Then *DrawCircle(PointA, PointB, TempA, CircleA)*. Then *DrawCircle(PointB, PointA, TempB, CircleB)*. Then *MarkCircleCircleIntersections(PointA, PointB, TempA, TempB, CircleA, CircleB, IntersectionA, IntersectionB)*. Next, *MakePerpendicularBisectorLine(IntersectionA, IntersectionB, MarkA, MarkB)* to identify the extrema of the extended line. Finally, *DrawLine(MarkA, PointA, Line)*, then *DrawLine(MarkB, PointB, Line)*.

Demonstration and Comparison

At this stage, if a shape can be provably built with compass-straightedge geometry, the ants can theoretically build it, given a finite-state automaton coded with the steps necessary. As a simple example, Figure 3 shows the process of building an equilateral triangle from two prespecified points. Using this approach, we have built a variety of structures: see for example the hexagon and angle bisector in Figure 4. Noise can cause some failures: in our simulator triangles presently have a 95% success rate, angle bisection is 91%, and hexagons are 87%: we believe these rates can still be improved.

While we have demonstrated that a basic pheromone model is *capable* of a nontrivial task such as this, we admit that it is not *efficient*: the agents build a variety of gradients throughout the environment, and if the system must use evaporation, then they must also maintain gradients until they have finished a subtask. Because of the locality of the pheromone model, and the need to move randomly, the approach is also quite *noisy*. As can be seen in Figures 3 and 4, the resulting lines, shapes, and angles are not ideal.

Broadcasting With *small* amount of global embedded communication, we can dramatically outperform a pheromone
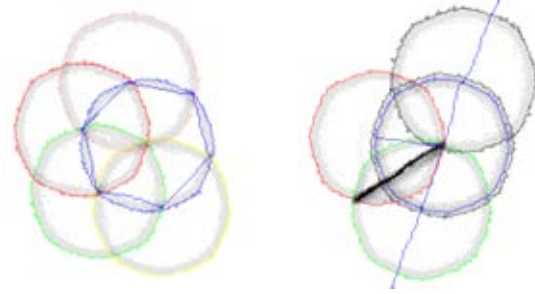
Figure 4: Hexagon (left) and Angle Bisection (right).

model. To show this, we compare the model against the same model augmented with *broadcast beacons*: objects which an ant may deploy at any time and associate with a pheromone. Ants can detect the presence, distance to, and relative angle to a broadcast beacon anywhere in the environment. As a result, they can easily follow along a circle (the locus of points a certain distance from a beacon), or head down a line passing through two beacons (the locus of points where both beacons are at the same relative angle or opposite angles). This eliminates the need to build and maintain gradients, and so ants with broadcast beacons can generally complete most tasks over dramatically faster. Furthermore, as they do not use local updating on a square grid, broadcast beacons' "gradients", so to speak, are circular without evaporation.

Except for the addition of broadcast beacons deployment and sensing, the ants are the same: in fact the revised procedures work largely the same way as the all-pheromone approach. One exception: because relative angle to beacons is reliable, the ants can do *DrawExtendedLine(...)* by simply going away from both beacons (that is, having both beacons directly behind the ant), though the original perpendicular bisector approach could still have been used.

For some examples, consider Figure 5, and compare against the same tasks done in Figures 3 and 4: there is something to be said for globally accessible information.[1]

Here we describe the additional basic procedures, then the geometric construction procedures, for ants using beacons.

---

[1]With three beacons, you could just do triangulation! But we want to show the ants' capability without sophisticated trigonometry.

Further Basic Procedures with Broadcast Beacons

GotoOrPlaceBeacon(*P*)   If a beacon for *P* exists, head towards the beacon, and signal *done* on arrival. If the beacon does not exist, *GoUp(P)* until a site is discovered marked with *P*, then place a beacon for *P* at that site, and signal *done*.

GotoOrPlaceBeaconShortCircuit(*P*)   If a beacon for *P* exists, simply signal *done*. If the beacon does not exist, *GoUp(P)* until a site is discovered marked with *P*, then place a beacon for *P* at that site, and signal *done*.

DrawLine(*LineA, LineB, P*)   *GotoOrPlaceBeaconShortCircuit(LineB). GotoOrPlaceBeacon(LineA)*. This ensures that beacons are located at both *LineA* and *LineB*, and that the ant is at *LineA*. Then head along the line between *LineA* and *LineB*, towards *LineB*, drawing the line with pheromone *P*. When at *LineB*, signal *done*.

Geometric Construction with Broadcast Beacons

MarkLineLineIntersection(*Line1A, Line1B, Line2, B*)
GotoOrPlaceBeacon(Line1A). Then head down the line which passes through *Line1A* and *Line1B*. It doesn't matter what direction. If the ant has gone "too far" (we define this as being at the world border, but any large measure is fine), turn around and head the other direction. When the ant discovers the pheromone *Line2*, denoting the trace of the second line, this is the intersection of the two lines. Place beacon *B* at this position and signal *done*.

MarkCircleLineIntersections(*LineA, LineB, Circle, A, B*)
GotoOrPlaceBeacon(LineA). Then head down the line which passes through *LineA* and *LineB* in the direction of (and past) *LineB*. When the ant has gone "too far", turn around and head the other direction. When the ant discovers the pheromone *Circle*, denoting the first intersection of the line and circle, place beacon *A* at this position. Then head down the line past *LineA*. When the ant has again gone "too far", turn around and head the other direction. When the ant discovers the pheromone *Circle*, denoting the second intersection, place beacon *B* at this position and signal *done*.

MarkCircleCircleIntersections(*CenterA, CenterB, CircleA, CircleB, A, B*)   *GotoOrPlaceBeacon(CenterA)*, then *GotoOrPlaceBeacon(CenterB)*. Next, randomly head either to *CenterA* from *CenterB* or to *CenterB* from *CenterA*. If the ant goes to *CenterA*, then when the ant finds the circle traced with *CircleB*, follow clockwise along the *CircleB* pheromone until an intersection point is found (having both *CircleA* and *CircleB*), then *PlaceBeacon(A)*. Continue along *CircleB* until another intersection point is found, then *PlaceBeacon(B)*. On the other hand, if the ant goes to *CenterB*, then when the ant finds the circle traced with *CircleA*, follow clockwise along the *CircleA* pheromone until an intersection point is found (having both *CircleA* and *CircleB*), then *PlaceBeacon(B)*. Continue along *CircleB* until another intersection point is found, then *PlaceBeacon(A)*. Either way, finally signal *done*.
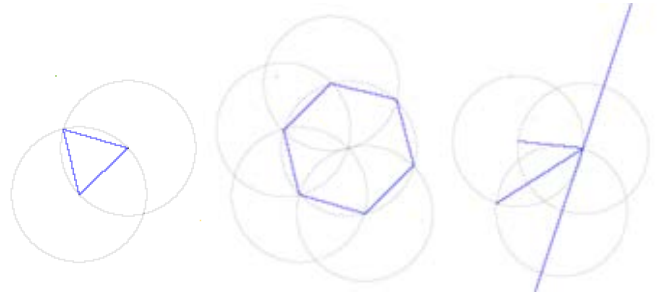
Figure 5: Equilateral Triangle (left), Hexagon (center), and Angle Bisection (right) using Broadcast Beacons. Compare with Figures 3 and 4.
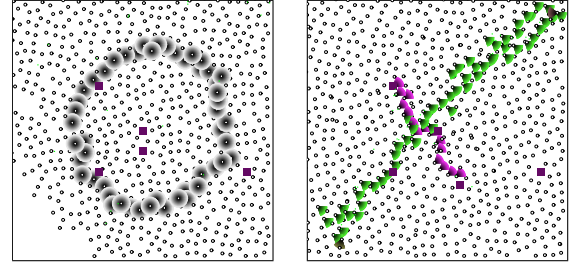
Figure 6: Circle (left) and Perpendicular Line Bisector (right) using a simulator for robots with deployable sensor motes.

DrawExtendedLine(*LineA, LineB, P*)   *GotoOrPlaceBeaconShortCircuit(LineA)*. Then *GotoOrPlaceBeacon(LineB)*. This ensures that beacons are located at both *LineA* and *LineB*, and that the ant is at *LineB*. Then head along the line between *LineB* and *LineA*, towards and ultimately past *LineA*, drawing the line with pheromone *P*. When the ant has gone "too far", *GotoOrPlaceBeacon(LineA)*. Once at *LineA*, head along the line between *LineA* and *LineB*, towards and ultimately past *LineB*, drawing the line with pheromone *P*. When the ant has again gone "too far", signal *done*.

DrawCircle(*Center, EdgePoint, P*)   *GotoOrPlaceBeaconShortCircuit(Center)*. Then *GotoOrPlaceBeacon(EdgePoint)*. This ensures that beacons are located at both *Center* and *EdgePoint*, and that the ant is at *EdgePoint*. *LoadRegister(Center)*. Then head along the path (either direction) where the *Center* pheromone is equal to the register (this essentially traces along the circle), while drawing the line with pheromone *P*. When the ant is back at *EdgePoint*, signal *done*.

Conclusion

We have demonstrated through proof of concept that, using only non-diffusing pheromones as a communication model, a swarm of ants can work together to perform compass-straightedge construction. This is a nontrivial task, but it demonstrates that indirect communication can enable sophisticated collaborative work.

This demonstration also illustrates a potential weakness in our pheromone model: it can be *very* inefficient, as whole ar-

eas must be painted with pheromones and perhaps constantly updated. For this reason we feel our demonstration falls near the limit of what these models are realistically capable of supporting. The broadcast beacons method, on the other hand, still allows for efficient constructions while using only sparse, robot-deployable devices.

Both methods are applicable to real robots, and immediate future work is to demonstrate this on an actual robot swarm. We have gathered preliminary (and noisy) results for the pheromone model in a simulator in which robots use deployable wireless sensor motes to create a sparse pheromone graph (as in Hrolenok et al. (2010) and Russell et al. (2015)); this is shown in Figure 6. To use such capabilities in a real-world scenario, however, will require solutions to a number of additional issues, including: obstacles in the environment, dynamic environments where marked areas might be removed, and environments without a known border.

Acknowledgments

References

Allwright, M., et al. (2014). SRoCS: Leveraging stigmergy on a multi-robot construction platform for unknown environments. In *Swarm Intelligence*, 158–169.

Ardiny, H., Witwicki, S., and Mondada, F. (2015). Construction automation with autonomous mobile robots: A review. In *RSI International Conference on Robotics and Mechatronics (ICROM)*, 418–424.

Bonabeau, E. (1996). Marginally stable swarms are flexible and efficient. *Journal de Physique I*, 6(2):309–324.

Brambilla, M., et al. (2013). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41.

Chibaya, C. and Bangay, S. (2007). A probabilistic movement model for shortest path formation in virtual ant-like agents. In *Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries*, 9–18.

Deneubourg, J.-L., et al. (1990). The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior*, 3(2):159–168.

Dorigo, M., Bonabeau, E., and Theraulaz, G. (2000). Ant algorithms and stigmergy. *Future Generation Computer Systems*, 16(8):851–871.

Gauss, C. (1801). *Disquisitiones Arithmeticae*, chap. 366.

Gleason, A. (1988). Angle trisection, the heptagon, and the triskaidecagon. *The American Mathematical Monthly*, 95(3):185–194.

Hrolenok, B., et al. (2010). Collaborative foraging using beacons. In *AAMAS*, 1197–1204.

Kowadlo, G. and Russell, R. A. (2008). Robot odor localization: A taxonomy and survey. *International Journal of Robotics Research*, 27(8):869–894.

Mamei, M. and Zambonelli, F. (2007). Pervasive pheromone-based interaction with RFID tags. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 2(2):4.

Mascheroni, L. (1797). *La Geometria del Compasso*.

Mohr, G. (1672). *Euclides Danicus*.

Panait, L. and Luke, S. (2004). A pheromone-based utility model for collaborative foraging. In *AAMAS*, 36–43.

Parker, C. A. C. and Zhang, H. (2006). Collective robotic site preparation. *Adaptive Behavior*, 14(1):5–19.

Pitonakova, L. and Bullock, S. (2013). Controlling ant-based construction. In *ALIFE*, 151–158.

Prabhakar, B., Dektar, K. N., and Gordon, D. M. (2012). The regulation of ant colony foraging activity without spatial information. *PLoS Computational Biology*, 8(8):1–7.

Russell, K., et al. (2015). Swarm robot foraging with wireless sensor motes. In *AAMAS*, 287–295.

Russell, R. (1999). Ant trails — an example for robots to follow? In *ICRA*, 2698–2703.

Sarhangi, R. (2007). Geometric constructions and their arts in historical perspective. In *Proceedings of the Bridges Donostia Conference*, 233–240.

Stewart, R. L. and Russell, R. A. (2006). A distributed feedback mechanism to regulate wall construction by a robotic swarm. *Adaptive Behavior*, 14(1):21–51.

Sugawara, K. and Doi, Y. (2014). Collective construction of dynamics structure: collaboration between semi-active blocks and simple robots. In *2014 IEEE/SICE International Symposium on System Integration (SII)*, 118–121.

Werfel, J. and Nagpal, R. (2008). Three-dimensional construction with mobile robots and modular blocks. *International Journal of Robotics Research*, 27(3-4):463–479.

Werfel, J., Petersen, K., and Nagpal, R. (2011). Distributed multi-robot algorithms for the TERMES 3d collective construction system. In *Workshop on Reconfigurable Modular Robotics, (at IROS)*.

Wodrich, M. and Bilchev, G. (1997). Cooperative distributed search: The ants' way. *Control and Cybernetics*, 26.

Ziparo, V., et al. (2007). RFID-based exploration for large robot teams. In *ICRA*, 4606–4613.

Evolving Real-time Heuristic Search Algorithms

Vadim Bulitko

Department of Computing Science
University of Alberta
Edmonton, Alberta, T6G 2E8, Canada
bulitko@ualberta.ca

Abstract

Heuristic search is a core area of Artificial Intelligence, successfully applied to planning, constraint satisfaction and game playing. In real-time heuristic search autonomous agents interleave planning and plan execution and access environment locally which make them more suitable for Artificial Life style settings. Over the last two decades a large number of real-time heuristic search algorithms have been manually crafted and evaluated. In this paper we break down several published algorithms into building blocks and then let a simulated evolution re-combine the blocks in a performance-based way. Remarkably, even relatively short evolution runs result in algorithms with state-of-the-art performance. These promising preliminary results open exciting possibilities in the field of real-time heuristic search.

1 Introduction and Related Work

Artificial Life (ALife) settings afford a researcher an intuitive testbed to study autonomous agents. In particular, ALife has been used to study emergence of various cognitive mechanisms, including a source of rewards (Ackley and Littman, 1991) in a Reinforcement Learning setting (Sutton and Barto, 1998). In this paper we propose to use ALife to study heuristic search algorithms. Heuristic search is a core area of Artificial Intelligence with a long history (Hart et al., 1968) and a broad applicability to planning, game-playing and constraint-optimization tasks. Heuristic search algorithms take a search graph and a start and goal states and output a path through the graph that connects the start and the goal. A canonical example is pathfinding on a road map: one can ask their in-car GPS to find a route from Edmonton to Cancun. Shorter routes may be preferred and quick computation times are valued.

We focus on real-time heuristic search — a subclass of agent-centered algorithms (Koenig, 2001) where the agent has to act before a full solution to the search problem is computed and has access only to the environment and data in the vicinity of the agent's current state. In other words, plan execution must be interleaved with the planning process and there is no global view of the world. These constraints would be important in the context of a self-driving car: its steering algorithm needs to issue commands to the steering wheel so many times per second while the GPS is computing the full route, *regardless* of how distant the goal is. Another application of real-time heuristic search is distributed search such as routing in *ad hoc* sensor networks (Bulitko and Lee, 2006).

Starting with LRTA* (Korf, 1990) real-time heuristic search agents interleave three processes: local planning, heuristic learning and move selection. In over the two decades since LRTA*, researchers have explored different methods for looking ahead during the planning stage of each cycle (Koenig and Sun, 2009); different heuristic learning rules (Bulitko; Hernández and Meseguer; Bulitko and Lee; Rayner et al.; Koenig and Sun; Rivera et al., 2004; 2005; 2006; 2007; 2009; 2015) and different move selection mechanisms (Ishida; Shue and Zamani; Shue and Zamani; Shue et al.; Bulitko and Lee; Hernández and Baier, 1992; 1993a; 1993b; 2001; 2006; 2012). Finally, information in addition to the heuristic has been learned during (Bulitko et al.; Sturtevant et al.; Sturtevant and Bulitko; Sharon et al., 2007; 2010; 2011; 2013) and before (Bulitko et al.; Bulitko et al.; Botea; Bulitko et al.; Lawrence and Bulitko, 2008; 2010; 2011; 2012; 2013) the search.

The number of techniques proposed by the researchers in the field of real-time heuristic search is overwhelming. More importantly, the interactions between these techniques are difficult to analyze empirically (Bulitko and Lee, 2006) or theoretically (Sturtevant and Bulitko, 2014). In this paper, we frame the problem of finding a high-performance combination of real-time heuristic search techniques as a survival task. To do so we set up a colony of autonomous real-time heuristic search agents whose genes determine their operation in their life-time. Finding high-quality solutions sustains an agent and eventually allows it to mate and reproduce. During the reproduction new agents are born, each with a slightly different genetic code.

Unlike human researchers in the field of real-time heuristic search, such a simulated evolution has no prior expectations, intuitions or biases. It simply conducts a randomized parallel search of a large space of real-time heuristic search algorithms. Yet, preliminary results in the standard

testbed of pathfinding on video-game maps are promising. An evolution run on a desktop computer in under a day led to a new real-time heuristic search algorithm that outperforms the state of the art. The emergence of such new high-performance algorithms appears to be a robust phenomena as we have repeated the evolution process several times, with similar results.

However, evolution has two downsides relative to the traditional process of designing real-time heuristic search algorithms: (i) it does not prove any theoretical properties of the new algorithms and (ii) it does not intuitively explain their performance. Thus, we suggest using the evolution as a computer-assisted exploratory step, to be followed by theoretical analysis and additional manual design.

The rest of the paper is organized as follows. Section 2 formally defines the problem we are attempting to solve. We then review the common framework of heuristic-learning real-time heuristic search algorithms in Section 3 and describe the building blocks in Section 4. We conduct a simulated evolution in a class of real-time heuristic search algorithms in Section 5 and present the empirical results in Section 6. We then conclude with directions for future work.

2 Problem Formulation

In line with previous research, we define a *search problem* $\mathbf{S}$ as the tuple (S, E, c, s_0, s_g, h) where S is a finite set of *states* and $E \subset S \times S$ is a set of *edges* between them. S and E jointly define the search graph which is assumed to be undirected: $\forall s_a, s_b \in S\,[(s_a, s_b) \in E \implies (s_b, s_a) \in E]$ and has no self-loops: $\forall s \in S\,[(s, s) \notin E]$. The graph is weighted by the strictly positive edge costs $c : E \to \mathbb{R}^+$ which are symmetric: $\forall s_a, s_b \in S\,[c(s_a, s_b) = c(s_b, s_a)]$. Two states s_a and s_b are *immediate neighbors* iff there is an edge between them: $(s_a, s_b) \in E$; we denote the set of immediate neighbors of a state s by $N(s)$. A *path* P is a sequence of states $(s_0, s_1, \ldots, s_n)$ such that for all $i \in \{0, \ldots, n-1\}$, $(s_i, s_{i+1}) \in E$. We assume that the search graph (S, E) is connected (i.e., any two vertices have a path between them) which makes it safely explorable.

At all times $t \in \{0, 1, \ldots\}$ the agent occupies a single state $s_t \in S$, called the *current state*. The state s_0 is the start state and is given as a part of the problem. The agent can change its current state, that is, move to any immediately neighboring state in $N(s)$. The traversal incurs a travel *cost* of $c(s_t, s_{t+1})$. The agent is said to solve the search problem at the earliest time T it arrives at the goal state: $s_T = s_g$. The *solution* is a path $P = (s_0, \ldots, s_T)$: a sequence of states visited by the agent from the start state until the goal state. The cumulative cost of all edges in a solution is called *solution cost* and is formally defined as $c_A(\mathbf{S}) = \sum_{t=0}^{T-1} c(s_t, s_{t+1})$ for algorithm A. The cost of the shortest possible path between states $s_a, s_b \in S$ is denoted by $h^*(s_a, s_b)$. We abbreviate $h^*(s, s_g)$ as $h^*(s)$. We define *suboptimality* of the agent on a problem as the ratio of the

solution cost the agent incurred to the cost of the shortest possible solution: $\alpha(A, \mathbf{S}) = \frac{c_A(\mathbf{S})}{h^*(s_0)}$. For instance, suboptimality $\alpha(\text{LRTA*}, \mathbf{S}) = 2$ means that the agent driven by the LRTA* algorithm found a solution to $\mathbf{S}$ twice as long as optimal. Lower values are preferred; 1 indicates optimality.

The other performance measure we are concerned with in this paper is the *scrubbing complexity* (Huntley and Bulitko, 2013). It is defined as the average number of state visits the agent makes while solving a problem. Formally, let $v_A^{\mathbf{S}} : S \to \mathbb{N} \cup \{0\}$ be the number of state visits the agent driven by algorithm A made while solving a problem $\mathbf{S}$. The scrubbing complexity is then defined over the subset of states that the agent visited at least once: $S_{\text{visited}} = \{s' \in S \mid v_A^{\mathbf{S}}(s') \geq 1\}$ as $\tau(A, \mathbf{S}) = \frac{1}{|S_{\text{visited}}|} \sum_{s \in S_{\text{visited}}} v_A^{\mathbf{S}}(s)$. For instance, $\tau(\text{LRTA*}, \mathbf{S}) = 7.5$ means that while solving problem $\mathbf{S}$, on average the agent driven by the LRTA* algorithm visited a state 7.5 times (states that were not visited at all do not contribute to the average). Lower values of $\tau(\mathbf{S})$ are preferred since re-visiting states tends to look irrational to an external observer. This is a major reason why real-time heuristic-search methods are hardly used for pathfinding in actual video games. Instead, game developers prefer non-real-time heuristic search such as variants of path-refinement A* (Sturtevant, 2007).

In its operation the agent has access to a heuristic $h : S \to [0, \infty)$. The heuristic function is a part of the search problem specification and is meant to give the agent an estimate of the remaining cost to go. Unlike much literature in the field, we do *not* assume admissibility or consistency of the initial heuristic but require that $h(s_g) = 0$. The search agent can modify the heuristic as it sees fit as long as it remains non-negative and the heuristic of the goal state s_g remains 0. The heuristic at time t is denoted by h_t; $h_0 = h$.

We say that a search agent is *real time* iff its computation time between its moves is upper-bounded by a constant independent of the number of states in the search space. We will additionally require our search algorithms to be *agent centered* (Koenig, 2001) insomuch as they have access to the heuristic, states and edges only in a bounded vicinity of the agent's current state and the bound is independent of the number of states.

We say that a search agent is *complete* iff it solves any search problem as defined above. That is, it is required to terminate in the goal state s_g at time $T < \infty$. Since we deal with randomly generated algorithms which may or may not be complete, we impose an upper bound on their travel cost. Any algorithm whose suboptimality on a problem exceeds $\alpha_{\max}$ is said to not solve a problem. We implement this by monitoring the agent's travel cost and terminating an agent as soon as it reaches or exceeds $\alpha_{\max} h^*(s_0)$. The resulting suboptimality is then recorded and contributes to the average suboptimality of the agent over a set of problems.

The problem we tackle in this paper is to develop a real-

time heuristic search algorithm that has low suboptimality (α) and low scrubbing complexity (τ).

3 Basic Real-time Heuristic Search

As Section 1 presented, many ways of improving on LRTA* (Korf, 1990) towards the two measures have been proposed. In this paper we specifically focus on heuristic learning and movement rules. To isolate the problem, we fix the lookahead at 1 (i.e., allow the agent to consider only the immediate neighbors of its current state during the planning stage) and allow the agent to update its heuristic only in its current state. In other words, our local search space and the local learning spaces are limited to the agent's current state. With these limitations, LRTA* becomes Algorithm 1.

Algorithm 1: Basic Real-time Heuristic Search

input : search problem (S, E, c, s_0, s_g, h)
output: path $(s_0, s_1, \ldots, s_T), s_T = s_g$
1   $t \leftarrow 0$
2   $h_t \leftarrow h$
3   **while** $s_t \neq s_g$ **do**
4     $s_{t+1} \leftarrow \arg \min_{s \in N(s_t)} (c(s_t, s) + h_t(s))$
5     $h_{t+1}(s_t) \leftarrow \max \left\{ h_t(s_t), \min_{s \in N(s_t)} (c(s_t, s) + h_t(s)) \right\}$
6     $t \leftarrow t + 1$
7   $T \leftarrow t$

A search agent following the algorithm begins in the start state s_0. It then executes a fixed loop until it reaches the goal s_g (line 3). At each iteration of the loop, the agent expands the current state s_t by generating its immediate neighbors $N(s_t)$ (local planning). It computes its action by selecting the next state s_{t+1} among the neighbors to minimize the estimated cost of traveling to the goal through that neighbor (line 4). Ties among neighbors that have the same $c + h$ values are broken with a tie-breaking schema that is consistent over state revisits. Then, in line 5, the agent updates (learns) its heuristic in the current state from $h_t(s_t)$ to $h_{t+1}(s_t)$. Note that the explicit maximum of the state's old heuristic value and the new value causes the heuristic to never decrease. Such a maximum is unnecessary if the heuristic is consistent and is commonly omitted in the literature. We do not assume our heuristic to be consistent and hence put the maximum in explicitly. The agent then changes its current state to the neighbor and the cycle repeats.

4 Building Blocks

To define the space of real-time heuristic search algorithms, we first abstract the base algorithm (Algorithm 1) into a search algorithm template (Algorithm 2). The template still has the main loop (line 3) which the agent executes until it gets to the goal state. Within the loop the agent repeatedly executes the movement (line 4) and the learning (line 5) rules. The rules can include the following building blocks.

Algorithm 2: Search Algorithm Template

input : search problem (S, E, c, s_0, s_g, h)
output: path $(s_0, s_1, \ldots, s_T)$ such that $s_T = s_g$
1   $t \leftarrow 0$
2   $h_t \leftarrow h$
3   **while** $s_t \neq s_g$ **do**
4     $s_{t+1} \leftarrow$ new s due to a movement rule
5     $h_{t+1}(s_t) \leftarrow \max \{h_t(s_t), \text{new } h \text{ due to a learning rule}\}$
6     $t \leftarrow t + 1$
7   $T \leftarrow t$

4.1 Movement Rule Building Blocks

For the movement rule we use line 4 in the base Algorithm 1 with the following possible blocks on top:

Backtracking (Shue and Zamani; Shue and Zamani; Shue et al.; Bulitko and Lee, 1993a; 1993b; 2001; 2006) causes the agent to move back to the previous state on its path when the heuristic is updated. For simplicity we do not consider the learning quota T parameter of SLA*T and LRTS and move the agent back to the previous state as soon as the heuristic function is updated (i.e., learning takes place). The intuition of backtracking is that upon detecting an inaccurate heuristic value the agent should not only update it in the current state but also in the previous states whose heuristic values may be dependent on it.

Depression avoidance (Hernández and Baier, 2012) detects whether the current state is a part of a heuristic depression — an area of the state space where the heuristic values are inaccurately low — and tries to guide the agent out of such a depression. The problem of heuristic depressions was identified early on (Ishida, 1992) and linked to state-revisitation and increased solution cost (Huntley and Bulitko, 2013). In our building block we use the method implemented in daLRTA* by Hernández and Baier (2012). Specifically, to select a neighbor of the current state as its next state, the agent considers only the states $N_{\text{min learning}}(s_t)$ where the amount of learning to-date is minimal:

$$N_{\text{min learning}}(s_t) = \{s \in N(s_t) \mid |h_0(s) - h_t(s)| = \mu\}$$
$$\mu = \min_{s \in N(s_t)} |h_0(s) - h_t(s)|. \quad (1)$$

Then the agent selects its next state s_t from the set $N_{\text{min learning}}(s_t)$ in the usual fashion (i.e., the one that minimizes $c(s_t, s) + h_t(s)$ in line 4, Algorithm 1).

Removing expendable states (Sharon et al., 2013) is meant to reduce the size of the search space by eliminating a state whose immediate neighbors can be all reached from each other within the immediate neighborhood. Such states are called locally expendable. In line with Sharon et al. (2013) we remove a locally expendable state from the search graph only when the heuristic of the state is updated by the agent.

4.2 Learning Rule Building Blocks

For the learning rule, we extend the basic mini-min update of LRTA* (line 5 in the base Algorithm 1) with the following

possible blocks:

Heuristic weighting is meant to accelerate the learning process and thus discourage the agent from re-visiting states (a major problem with heuristic depressions). We modify the weighting rule of Rivera et al. (2015):

$$h_{t+1}(s_t) \leftarrow \max\left\{h_t(s_t), \min_{s \in N(s_t)} (w \cdot c(s_t, s) + h_t(s))\right\} \quad (2)$$

by moving the weight outside of the $c + h$:

$$h_{t+1}(s_t) \leftarrow \max\left\{h_t(s_t), \ w \cdot \min_{s \in N(s_t)} (c(s_t, s) + h_t(s))\right\} \quad (3)$$

which further increases the updates to the heuristic of the current state. We do so as our preliminary tests have shown the modified rule to be more robust and thus easier to tune the weight for. Note that $w \geq 1$ means that the heuristic may become inadmissible and inconsistent.

Learning operators other than the $\min$ can be used in the learning rule (3) if admissibility/consistency is not a requirement. We allow our agents to replace the $\min$ with avg, median or $\max$.

Lateral learning allows the learning operator ($\min$, avg, median, $\max$) in the learning update rule (3) to be taken over a portion of the neighborhood $N(s_t)$. Exploratory experiments have shown that doing so can improve the agent's performance. We define the portion using a parameter b called *beam width*. Specifically, the partial neighborhood N_b^f of a state s_t is defined as the b fraction of the neighborhood $N(s_t)$ with the lowest f values:

$$N_b^f(s) = \left(s^1, \ldots, s^{\lfloor b|N(s_t)|\rfloor}\right) \quad (4)$$

where $\left(s^1, \ldots, s^{\lfloor b|N(s_t)|\rfloor}, \ldots, s^{|N(s_t)|}\right)$ is the immediate neighborhood sorted in the ascending order by their $f = c + h$. For instance, s^1 has the lowest $f(s^1) = c(s_t, s^1) + h(s^1)$ value in the set $\{f(s) \mid s \in N(s_t)\}$ whereas $s^{|N(s_t)|}$ has the highest f value in that set. Clearly, for $b = 1$ we get the full neighborhood: $N_1^f(s_t) = N(s_t)$. For $b = 0$ we define $N_0^f(s_t)$ as the neighbor with the lowest f: $\{s^1\}$.

4.3 Putting the Building Blocks Together

With these building blocks, the template Algorithm 2 becomes Algorithm 3. The main loop is the same as before (line 3). Inclusion of the building blocks is determined by the control parameters w, b, lop, da, $\mathtt{expendable}$, $\mathtt{backtrack}$ as follows. If the depression avoidance block is present in the agent ($\mathrm{da} = true$) then line 5 temporarily sets the neighborhood to only the states where the amount of learning $|h_t(s) - h_0(s)|$ is minimal. The learning rule in line 6 covers heuristic weighting, learning operator and lateral learning, using the control parameters w, lop and b.

If the expendable block is present in the agent ($\mathtt{expendable} = true$) then in line 8 the current state is

---

Algorithm 3: Real-time Heuristic Search w/ Building Blocks

> **input** : search problem (S, E, c, s_0, s_g, h), control parameters w, b, lop, da, $\mathtt{expendable}$, $\mathtt{backtrack}$
> **output**: path $(s_0, s_1, \ldots, s_T)$, $s_T = s_g$

1   $t \leftarrow 0$
2   $h_t \leftarrow h$
3   **while** $s_t \neq s_g$ **do**
4   **if** da **then**
5         $N(s_t) \leftarrow N_{\mathrm{min\ learning}}(s_t)$
6      $h_{t+1}(s_t) \leftarrow$
       $\max\left\{h_t(s_t), \ w \cdot \operatorname*{lop}_{s \in N_b^f(s_t)} (c(s_t, s) + h_t(s))\right\}$
7   **if** $\mathtt{expendable}$ & $h_{t+1}(s_t) > h_t(s_t)$ & $\mathcal{E}(s_t)$ **then**
8   remove s_t from the search graph
9   **if** $\mathtt{backtrack}$ & $h_{t+1}(s_t) > h_t(s_t)$ **then**
10        $s_{t+1} \leftarrow s_{t-1}$
11   **else**
12        $s_{t+1} \leftarrow \arg\min_{s \in N(s_t)} (c(s_t, s) + h_t(s))$
13      $t \leftarrow t + 1$
14 $T \leftarrow t$

---

removed from the graph if there was learning in it and it is indeed locally expendable (denoted by the predicate $\mathcal{E}$). Finally, if the agent learned in the current state and the backtracking block is present ($\mathtt{backtrack} = true$) then the agent will move back to the previous state in line 10. Otherwise it moves forward in line 12. If there is no previous state (i.e., $s_t = s_0$) then the agent stays put.

If at any time the neighborhood $N(s_t)$ becomes empty (i.e., the agent has no moves to pick from) then the agent quits without producing a solution. Such unsolved problems contribute $\alpha_{\max} h^*(s_0) + \min_{s_a, s_b \in S} c(s_a, s_b)$ to the agent's statistics on the solution cost/suboptimality.

In the rest of the paper we compactly denote any such algorithm by listing its building blocks as $w \cdot \mathrm{lop}_b(c + h) + \mathtt{backtrack} + \mathrm{da} + \mathrm{E}$ where the last three parts are optional.

5 Simulated Evolution

We had briefly experimented with an ERL-style asynchronous evolution (Ackley and Littman, 1991) without an explicit fitness function or generations. However, given our current heuristic search code base, it was computationally prohibitive and we switched to the traditional style evolution with discrete generations and an explicit fitness function. Section 7 briefly discusses the alternative.

The simulated evolution proceeds as per Algorithm 4. It starts in line 2 with forming the initial population P_0 of K agents. Each agent is represented by its gene which encodes the building blocks used by the agent. Technically the gene is a vector $(w, b, \mathrm{lop}, \mathrm{da}, \mathtt{expendable}, \mathtt{backtrack})$. Each of six components of the initial agents' genes is picked

Algorithm 4: Evolution of Search Agents

---

input : search problems, batch size B, max suboptimality $\alpha_{\max}$, population size K, number of generations M

output: genome of p_{oldest}

1 $t \leftarrow 0$
2 create population P_0 of size K with random genes
3 **for** $t = 1, \dots, M$ **do**
4   **for** $p \in P_{t-1}$ **do**
5        $\phi(p) \leftarrow \alpha(p)$ over B problems, truncated at $\alpha_{\max}$
6   sort P_{t-1} by ϕ
7     $C \leftarrow \text{children}(P_{t-1}(1, \dots, K/2))$
8     $P_t \leftarrow P_{t-1}(1, \dots, K/2) \cup C$
9   update p_{oldest}

---

uniformly randomly from their respective ranges. Some of the ranges are domain specific and we will list them in Section 6.2. Binary gene components (e.g., `da` which takes on the values of $true$ and $false$) are represented as a continuous value in $[0, 1]$. The algorithm converts it to binary by rounding. For instance, an agent with the `da` gene of 0.4 will not have depression avoidance block as $\text{round}(0.4) = 0$. However, an agent with the `da` gene of 0.7 will perform depression avoidance as $\text{round}(0.7) = 1$. The same schema is used for the learning operator `lop` gene which takes on a continuous value in $[1, 4]$ but is rounded to the nearest discrete value inside the search agent to select the operator ($1, 2, 3, 4$ encode `min`, `avg`, `median` and `max` correspondingly).

During each of the M generations each agent of the population is evaluated on B problems picked randomly from search problems given to the evolution. The fitness of the individual p is denoted by $\phi(p)$ and is the average suboptimality of the agent p on the B problems (line 5). Each run is truncated if the agent exceeds the suboptimality $\alpha_{\max}$ as explained earlier in the paper.

The population is then sorted by the agents' fitness in line 6. The top half of the population, $P_{t-1}(1, \dots, K/2)$, are included in the next generation's population P_t and are allowed to reproduce. The bottom half are removed and are replaced by $K/2$ children in line 8.

The set of children C is created by randomly picking two parents from the top half of P_{t-1} and performing a cross over on their genes in line 7. In other words, each gene value of the child has a $50/50\%$ chance of coming from the father or the mother. The child's gene values are then mutated by adding a Gaussian noise of zero mean and the standard deviation of $1/100$th of the size of the gene range (e.g., $(4 - 1)/100 = 0.03$ for the `lop` gene). If adding Gaussian noise pushes the gene value outside of the valid range then the value is clipped. For instance, if a child inherited $\text{lop} = 4$ from its parent but the mutation noise made it $4 + 0.03 = 4.03$ then it will be brought back down to 4.

Finally, line 9 updates the running oldest agent p_{oldest} (i.e., the one who has survived the most generations so far) with an older one, if such is found in the current generation. The

genome of the earliest oldest agent is the output of the evolution. Ties between agents of the same age are broken in the favor of lower suboptimality.

6 Empirical Evaluation

The traditional testbed for real-time heuristic search is pathfinding in video games, where the planning time per move is limited to a few milliseconds for *all* simultaneously operating agents. The standard way (Sturtevant, 2012) of representing game maps is as a two-dimensional discrete grid where each grid cell is either available for the agent to pass through (i.e., vacant, shown white in Figure 1) or blocked by an obstacle (black). At each moment of time, the agent occupies a single vacant cell of the map which determines the agent's current state. The agent changes its state by moving from its current grid cell to one of its vacant neighbors, incurring a travel cost. In this paper we use the standard eight-connected maps where cardinal moves cost 1 and diagonal moves cost $\sqrt{2}$. A problem is solved when the agent enters the goal cell. At the beginning of each problem, the agent starts with the octile distance as the heuristic. Octile distance is the cost of the shortest path between a given cell and the goal cell *if* no cells are blocked by obstacles.
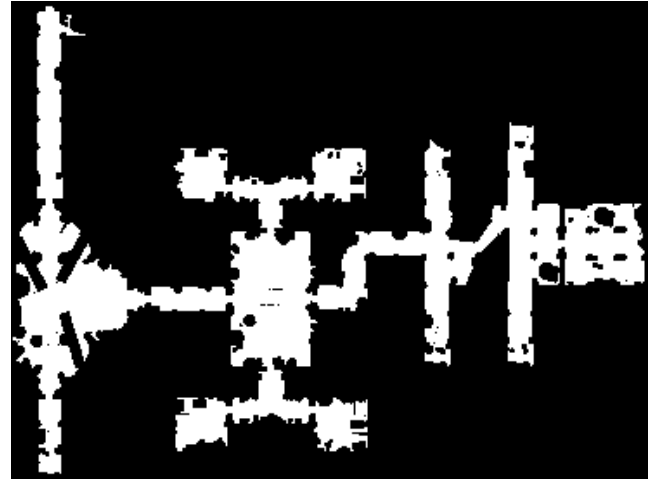
Figure 1: A video-game map from *Dragon Age: Origins*.

We implemented all algorithms in a mixture of C and MATLAB code run on Intel i5/i7-based desktop computers. Parts of the code were run in parallel on 4-6 CPU cores.

6.1 Problem Set

We used the benchmark problems from the Moving AI set (Sturtevant, 2012). We treated the water terrain type as obstacle and excluded all problems which thereby became unsolvable (e.g., the start state is in an obstacle cell). This resulted in 493298 problems situated on 342 maps. The maps were from the video games *StarCraft*, *WarCraft III*, *Baldur's*

Gate II (maps scaled up to 512×512) and *Dragon Age: Origins* (Figure 1).

6.2 Evolution

We conducted three evolution runs. The first run was of 50 generations each of 200 agents. Each agent of each generation was evaluated on 200 random problems. The maximum suboptimality $\alpha_{\max}$ was set to 1000. The maximum age of any agent was 14 generations and it was first achieved in generation 47 by the agent $8.223 \cdot \min_{0.341}(c + h)$+E. The run took approximately 6 hours, with the agents solving 92 problems/second.

The second run was of 25 generations each of 200 agents. Each agent of each generation was evaluated on 400 random problems. The maximum suboptimality $\alpha_{\max}$ was set to 1000. The maximum age of any agent was 11 generations and it was first achieved in generation 12 by the agent $7.952 \cdot \min_{0.720}(c + h)$+da+E. The run took approximately 5 hours, with the agents solving 110 problems/second.

The third run was of 200 generations each of 100 agents. Each agent of each generation was evaluated on 100 random problems. The maximum suboptimality $\alpha_{\max}$ was set to 1000. The maximum age of any agent was 16 generations and it was first achieved in generation 72 by the agent $8.061 \cdot \text{avg}_{0.029}(c + h)$+E. The run took approximately 9 hours, with the agents solving 63 problems/second.

We evaluated the resulting algorithms on non-overlapping sets of 10000 random problems. The suboptimality cutoff was set to $\alpha_{\max} = 10^5$ which allowed all of them to solve all problems. The results are found in Table 1 and suggest that the evolved individuals are similar. For the subsequent evaluation we picked $8.223 \cdot \min_{0.341}(c + h)$+E algorithm.

Table 1: Results of the three evolution runs. Means and standard errors of the mean are listed.

Algorithm	Suboptimality α	Scrubbing τ
$8.223 \cdot \min_{0.341}(c + h)$+E	20.70 ± 0.4782	1.17 ± 0.0036
$7.952 \cdot \min_{0.720}(c + h)$+da+E	21.17 ± 0.5220	1.18 ± 0.0037
$8.061 \cdot \text{avg}_{0.029}(c + h)$+E	21.35 ± 0.5366	1.18 ± 0.0038

6.3 Systematic Search

We also ran systematic search in the space of the possible algorithms. To do so, we tabulated the weight space w in 15 increments from 1 to 10. We tabulated the beam width b in 15 increments from 0 to 1. For each of those combinations we tried all four learning rule operators (min, avg, median and max) and all three movement rule blocks: depression avoidance, backtracking and removing expendable states. In total we created $15 \times 15 \times 4 \times 2 \times 2 \times 2 = 7200$ agents. Each of them was run on the same 100 random problems. The three agents with the lowest suboptimality were: $\max_{0.714}(c + h)$+E, $\max_{0.143}(c + h)$+da+E and $2.286 \cdot \text{avg}_{0.286}(c+h)$+da+E. Their suboptimalities over 100 problems were 14.56, 14.87 and 15.14 respectively.

As before, we evaluated the resulting algorithms on non-overlapping sets of 10000 random problems. The suboptimality cutoff was again set to $\alpha_{\max} = 10^5$ which allowed all three algorithms to solve all problems. The results are found in Table 2 and suggest that the evaluation on the 100 problems during the systematic search is not representative of the algorithms' true performance. For instance, the second algorithm was selected for its second-best suboptimality of 14.87 as measured on the 100 problems. However, as measured on 10000 problems, it performed substantially worse (suboptimality of 30.44).

Table 2: Results of the three evolution runs. Means and standard errors of the mean are listed.

Algorithm	Suboptimality α	Scrubbing τ
$\max_{0.714}(c + h)$+E	19.56 ± 0.5237	1.16 ± 0.0035
$\max_{0.143}(c + h)$+da+E	30.44 ± 1.4305	1.22 ± 0.0075
$2.286 \cdot \text{avg}_{0.286}(c + h)$+da+E	20.47 ± 0.5109	1.13 ± 0.0026

Such misleading estimates of the algorithm's performance appear a necessary downside of the systematic search. Indeed, to evaluate all 7200 algorithms in a reasonable amount of time, each can be evaluated only on a small set of problems. In contrast, our evolution looks for the oldest individuals which had to remain in the top half of the performance in *each* generation to make it to the next. For instance, the best algorithm found by the evolution, $8.223 \cdot \min_{0.341}(c+h)$+E, lasted for 14 generations and thus had its performance evaluated on 14×200 problems.

For the subsequent evaluation we picked $\max_{0.714}(c + h)$+E as the output of the systematic search.

6.4 Competing Algorithms

We selected two groups of published algorithms as competitors. In the first group, we have LRTA* (Korf, 1990) and its weighted variant wLRTA* (Rivera et al., 2015). To select the weight in wLRTA*, we ran $w \in \{1, 2, 3, 4, 5, 6, 7, 8, 16, 32, 64, 128, 256, 512, 1024, 2048\}$ on 6000 non-overlapping problems each and found that $w = 128$ gave the lowest suboptimality.

In the second group we have existing algorithms that explicitly aim to escape heuristic depressions quicker than LRTA*. Specifically, we evaluated the myopic versions (lookahead of 1) of aLRTA*, daLRTA* (Hernández and Baier, 2012), its weighted version w-daLRTA* (Rivera et al., 2015) and its combination with removing expendable states, daLRTA*+E (Sharon et al., 2013). We did not include f-LRTA* or f-LRTA*+E in our evaluation since both versions were found inferior to daLRTA* by Sharon et al. (2013). To select the weight parameter for w-daLRTA*, we ran $w \in \{1, 2, 3, 4, 5, 6, 7, 8, 16, 32, 64, 128, 256, 512, 1024, 2048\}$ on 6000 non-overlapping problems each and found that $w = 7$ gave the lowest suboptimality.

6.5 Competition Results

We ran the eight algorithms, each on 30000 non-overlapping problems randomly selected from the benchmark set. For each algorithm we measured its suboptimality α and scrubbing τ. The suboptimality cutoff was set to $10^5 \cdot h^*(s_0)$ and all algorithms solved all problems under the cutoff. For the systematic search we used algorithm $\max_{0.714}(c + h)$+E. For the evolution we used $8.223 \cdot \min_{0.341}(c + h)$+E. The means and standard errors of the mean are listed in Table 3.

Table 3: Performance of the algorithms: sample mean $\pm$ standard error of the mean.

Algorithm	Suboptimality α	Scrubbing τ
LRTA*	455.58 $\pm_{7.2984}$	16.24 $\pm_{0.1423}$
aLRTA*	349.21 $\pm_{6.3649}$	11.04 $\pm_{0.1175}$
daLRTA*	44.98 $\pm_{0.7425}$	1.98 $\pm_{0.0104}$
wLRTA*	39.81 $\pm_{0.5458}$	1.83 $\pm_{0.0078}$
wdaLRTA*	30.95 $\pm_{0.4521}$	1.48 $\pm_{0.0050}$
daLRTA*+E	31.82 $\pm_{0.7782}$	1.21 $\pm_{0.0041}$
systematic search	19.75 $\pm_{0.2836}$	1.17 $\pm_{0.0020}$
evolution	20.79 $\pm_{0.2809}$	1.17 $\pm_{0.0021}$

The new algorithms found by evolution and systematic search appear to produce shorter solutions and scrub less than the classical and contemporary competing algorithms.

Compared to the existing algorithms, the new algorithms use a combination of more aggressive learning rules, $\max_{0.714}(c + h)$ or $8.223 \cdot \min_{0.341}(c + h)$, with marking expendable states. Naturally, either of these combinations could have been found manually by the researchers but since it is time-consuming and tedious to try all combinations of building blocks, researchers tend to focus on a few, guided by their experience and intuition. Evolution and systematic search lack either and operate entirely based on algorithm performance (within the human-defined algorithm space).

7 Future Work

The promising preliminary results reported above open a few avenues for future work. First, while we tested the evolved algorithms on problems that they may not have seen during the evolution process, it is still possible that the resulting algorithms work so well merely because they overfit to the video-game pathfinding maps. Thus, the next step is to explore how effective the process is in other real-time heuristic search domains and across domains.

Second, additional real-time heuristic search techniques such as deeper lookahead of LSS-LRTA* (Koenig and Sun, 2009) can be made available to the evolution as building blocks. To make the competition fair, additional planning time per move should be incorporated into the fitness function. Another possibility to curb computational complexity is to assume that computations are subject to errors. For instance, one can introduce access errors to the heuristic

which is realistic when heuristic values are stored in the environment (a la pheromone in ant-colony algorithms) and the agents may not always read/write them perfectly. Then robust algorithms are likely to evolve (Ackley, 2013).

Finally, we can attempt to evolve robust search agents specifically for a class of problems. Following Ackley and Small (2014) we can have the agents continuously evolving in an asynchronous environment, similarly to the ERL testbed of Ackley and Littman (1991). In such a setting there would be no discrete generations and no explicit fitness function. Instead, each agent will maintain a health value which is depleted during the search and replenished when the agent reaches its goal, before taking on the next search problem. An agent which lived long enough and healthy enough is given an opportunity to reproduce. The domain-specificity would allow us not only to evolve the algorithms but also their innate heuristic function which can be encoded in the genes using function approximation.

8 Conclusions

We presented possibly the first application of evolutionary search to real-time heuristic search algorithms. We did so by breaking down classical and contemporary real-time heuristic search algorithms into building blocks for the evolution to pick from. In a large-scale evaluation in video-game pathfinding, evolved real-time heuristic search agents outperformed manually designed published algorithms.

Acknowledgments

References

Ackley, D. and Littman, M. (1991). Interactions between learning and evolution. *Artificial life II*, 10:487–509.

Ackley, D. H. (2013). Beyond efficiency. *Communations of the ACM*, 56(10):38–40.

Ackley, D. H. and Small, T. R. (2014). Indefinitely scalable computing = artificial life engineering. In *Proceedings of The Fourteenth International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14) 2014*, pages 606–613. MIT Press.

Botea, A. (2011). Ultra-fast optimal pathfinding without runtime search. In *Proceedings of the Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 122–127.

Bulitko, V. (2004). Learning for adaptive real-time search. *CoRR*, cs.AI/0407016.

Bulitko, V., Björnsson, Y., and Lawrence, R. (2010). Case-based subgoaling in real-time heuristic search for video game pathfinding. *Journal of Artificial Intelligence Research*, 39:269–300.

Bulitko, V. and Lee, G. (2006). Learning in real time search: A unifying framework. *Journal of Artificial Intelligence Research*, 25:119–157.

Bulitko, V., Luštrek, M., Schaeffer, J., Björnsson, Y., and Sigmundarson, S. (2008). Dynamic control in real-time heuristic search. *Journal of Artificial Intelligence Research*, 32:419–452.

Bulitko, V., Rayner, D. C., and Lawrence, R. (2012). On case base formation in real-time heuristic search. In *Proceedings of Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 106–111.

Bulitko, V., Sturtevant, N., Lu, J., and Yau, T. (2007). Graph abstraction in real-time heuristic search. *Journal of Artificial Intelligence Research*, 30:51–100.

Hart, P., Nilsson, N., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

Hernández, C. and Baier, J. A. (2012). Avoiding and escaping depressions in real-time heuristic search. *Journal of Artificial Intelligence Research*, 43:523–570.

Hernández, C. and Meseguer, P. (2005). LRTA*(k). In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1238–1243.

Huntley, D. A. and Bulitko, V. (2013). Search-space characterization for real-time heuristic search. *CoRR*, abs/1308.3309.

Ishida, T. (1992). Moving target search with intelligence. In *Proceedings of the National Conference on Artificial Intelligence*, pages 525–532.

Koenig, S. (2001). Agent-centered search. *Artificial Intelligence Magazine*, 22(4):109–132.

Koenig, S. and Sun, X. (2009). Comparing real-time and incremental heuristic search for real-time situated agents. *Journal of Autonomous Agents and Multi-Agent Systems*, 18(3):313–341.

Korf, R. (1990). Real-time heuristic search. *Artificial Intelligence*, 42(2–3):189–211.

Lawrence, R. and Bulitko, V. (2013). Database-driven real-time heuristic search in video-game pathfinding. *IEEE Transactions on Computational Intelligence and AI in Games*, 5(3):227–241.

Rayner, D. C., Davison, K., Bulitko, V., Anderson, K., and Lu, J. (2007). Real-time heuristic search with a priority queue. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 2372–2377.

Rivera, N., Baier, J. A., and Hernández, C. (2015). Incorporating weights into real-time heuristic search. *Artificial Intelligence*, 225:1–23.

Sharon, G., Sturtevant, N. R., and Felner, A. (2013). Online detection of dead states in real-time agent-centered search. In *Proceedings of the Symposium on Combinatorial Search*, pages 167–174.

Shue, L.-Y., Li, S.-T., and Zamani, R. (2001). An intelligent heuristic algorithm for project scheduling problems. In *Annual Meeting of the Decision Sciences Institute*.

Shue, L.-Y. and Zamani, R. (1993a). An admissible heuristic search algorithm. In *International Symposium on Methodologies for Intelligent Systems (ISMIS-93)*, volume 689 of *LNAI*, pages 69–75.

Shue, L.-Y. and Zamani, R. (1993b). A heuristic search algorithm with learning capability. In *ACME Transactions*, pages 233–236.

Sturtevant, N. (2007). Memory-efficient abstractions for pathfinding. In *Proceedings of Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 31–36.

Sturtevant, N. R. (2012). Benchmarks for grid-based pathfinding. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(2):144 – 148.

Sturtevant, N. R. and Bulitko, V. (2011). Learning where you are going and from whence you came: H- and G-cost learning in real-time heuristic search. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 365–370.

Sturtevant, N. R. and Bulitko, V. (2014). Reaching the goal in real-time heuristic search: Scrubbing behavior is unavoidable. In *Proceedings of the Symposium on Combinatorial Search*, pages 166–174.

Sturtevant, N. R., Bulitko, V., and Björnsson, Y. (2010). On learning in agent-centered search. In *Proceedings of the Conference on Autonomous Agents and Multiagent Systems*, pages 333–340.

Sutton, R. and Barto, A. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Comparing Human and Automated Evaluation of Open-Ended Student Responses to Questions of Evolution

Michael J. Wiser[1,2], Louise S. Mead[1,2,3], James J. Smith[1,2,3,4,5], and Robert T. Pennock[1,2,4,6,7]

[1]BEACON Center for the Study of Evolution in Action

[2]Program in Ecology, Evolutionary Biology and Behavior, Michigan State University, East Lansing, MI, USA

[3]Department of Integrative Biology, Michigan State University, East Lansing, MI, USA

[4]Lyman Briggs College, Michigan State University, E. Lansing, MI, USA

[5]Department of Entomology, Michigan State University, East Lansing, MI, USA

[6]Department of Philosophy, Michigan State University, East Lansing, MI, USA

[7]Department of Computer Science & Engineering, Michigan State University, East Lansing, MI, USA

mwiser@msu.edu

Abstract

Written responses can provide a wealth of data in understanding student reasoning on a topic. Yet they are time- and labor-intensive to score, requiring many instructors to forego them except as limited parts of summative assessments at the end of a unit or course. Recent developments in Machine Learning (ML) have produced computational methods of scoring written responses for the presence or absence of specific concepts. Here, we compare the scores from one particular ML program – EvoGrader – to human scoring of responses to structurally- and content-similar questions that are distinct from the ones the program was trained on. We find that there is substantial inter-rater reliability between the human and ML scoring. However, sufficient systematic differences remain between the human and ML scoring that we advise only using the ML scoring for formative, rather than summative, assessment of student reasoning.

Background

The central importance of evolution to teaching and learning in the biological sciences has been clearly established in all science education reform (States, 1900; Brewer and Smith, 2011). Adequate formative assessment instruments – administered during the course of instruction to gauge student understanding and reasoning in order to provide feedback for future instruction, instead of to assign a grade at the end of a unit – that measure student understanding of evolutionary concepts (Bishop and Anderson, 1990; Anderson et al., 2002), however, have until recently been rather limited (Nehm and Schonfeld, 2008). Part of the challenge in designing an effective instrument comes from the fact that student understanding of evolutionary concepts is complex, and constantly changing. Studies find that students hold both scientifically accurate and naive or non-scientific explanations simultaneously (Andrews et al., 2012; Hiatt et al., 2013) and that accurately identifying alternative conceptions can be difficult (Rector et al., 2012). Data also suggest students reason differently than experts, especially in response to different contextual elements of the sample questions. Undergraduates employ more naive concepts when applying explanations of natural selection to plants as compared to animals; trait loss as compared to trait gain; and unfamiliar taxa as compared to familiar taxa (Nehm and Ha, 2011). Furthermore, ascertaining the meaning of student responses is often very difficult. One study found that 81 percent of students incorporated lexically ambiguous language in their responses to open ended questions about evolutionary mechanisms (Rector et al., 2012).

Despite these challenges, assessing student knowledge is important, particularly in evaluating pedagogical practices designed to improve student understanding. In an effort to identify effective assessment strategies, we have been investigating the applicability of a new tool, EvoGrader (Moharreri et al., 2014).

Open-ended student responses can provide a wealth of data about student reasoning. Unfortunately, they can also be time- and labor-intensive to score. One study found that it took an average of four minutes for a human grader to score a single response for the nine ideas we analyze in this study (Moharreri et al., 2014). For even a class of 30 students, scoring five such questions would take ten hours, which quickly becomes prohibitive. If an instructor wants to get a general sense of student understanding on a formative assessment, a more rapid method is highly desirable.

An appealing potential solution to this problem would be if instructors had an automated system that was sufficiently sophisticated to evaluate student answers to such open-ended questions. Of course, this is not a simple task. Even setting aside the difficulty of parsing open-ended natural language responses in general, one still has the further problem of interpreting the appropriateness of answers in relation to content knowledge and overarching concepts. For instance, a science teacher may want to know whether a student's response demonstrates incorrect naive notions or whether it demonstrates concrete scientific understanding. Machine Learning systems have begun taking the first steps to accomplishing this difficult task.

Use of Machine Learning in Education

There is growing interest in using tools and techniques from Machine Learning in the classroom environment (Butler et al., 2014). In fact, a chapter has been written about using Machine Learning in educational science (Kidziński et al., 2016) within the context of a book on educational technologies. One area of particular interest is language processing. Machine learning techniques have been used to classify instructor questions according to Bloom's taxonomy (Yahya et al., 2013). Perhaps the biggest use of Machine Learning in an educational environment is in the automated scoring of student writing (reviewed in (Nehm et al., 2012b)).

One domain-specific example of ML techniques in language processing is provided by the web portal EvoGrader, discussed below. EvoGrader was designed to assess student understanding of natural selection, using a particular set of questions, consisting of a brief scenario and asking the students how a biologist would explain this scenario of evolutionary change or patterns. Our study seeks to measure how similar of scores this ML procedure provides to human scoring for questions on which the application has not been trained but which are written in the same style.

EvoGrader

EvoGrader (http://www.evograder.org) is a free, online service that analyzes open-ended responses to questions about evolution and natural selection, and provides users with formative assessments. It is described in detail in (Moharreri et al., 2014), but a brief description follows.

EvoGrader works by supervised machine learning. Participants (n=2,978) wrote responses to ACORNS assessment items (Nehm et al., 2012a) and ACORNS-like items (Bishop and Anderson, 1990), generating 10,270 student responses. These items consist of a prompt describing a short scenario relevant to natural selection, and ask students to write how a biologist would explain this situation. Participants spanned many different levels of expertise, including non-majors, undergraduate biology or anthropology majors, graduate students, postdocs, and faculty in evolutionary science. Each response was scored independently by two human raters for each of six Key Concepts (KC) and three Naive Ideas (NI) (see Box 1). These consensus scores were used to train EvoGrader, based on the supervised machine learning tools of LightSIDE (Mayfield and Rosé, 2013). LightSIDE provides feature extraction, model construction, and model validation, based on the human-scored responses.

EvoGrader's authors chose different methods to optimize the scoring algorithm for feature extraction for the 9 scoring models (one model for each concept) – all considered the dictionary of words used in a particular response, and reduced words to their stems; most removed high frequency low information words (e.g., the, of, and, it); some also included pairs of consecutive words (e.g., "had to", "passing on"), or removing misclassified data (see Moharreri et. al. (Moharreri et al., 2014) Table 2 for details).

After feature extraction, each response was converted to a set of vectors containing frequencies of words or word pairs. These vectors were then passed to a binary classifier, which underwent Sequential Minimal Optimization (SMO) (Platt 1999) for each of the 9 models. The SMO training algorithm iteratively assigned weights to words in the written responses until the model was able to match the human scores within a certain margin of error. The models were then validated with 10-fold cross-validation, using 90% of the data to generate a model and the remaining 10% of the data to validate it, and then repeating this procedure for a total of 10 times such that each 10% of the data was used for validation exactly once and model generation 9 times. The authors averaged these models to get the final models used by the program, assessing whether they met quality benchmarks (90% accuracy and kappa coefficients ≥ 0.8) defined by the creators, and adjusting the training until the models did.

EvoGrader uses these validated models to score new responses from web users. Users must upload data in a specific format, which the portal verifies. If the data is formatted correctly, EvoGrader then evaluates each response using the existing validated models, and provides both machine scored data in a downloadable .csv format and a variety of web visualizations of the data. (Fig. 1)

Methods

Student data

We administered pre-instruction and post-instruction tests consisting of two questions (see Box 1) about evolution to students in an Introductory Cell and Molecular Biology course in the fall semester of 2014. Both questions asked students about how evolutionary processes occur. Question 1 asks about an evolutionary gain of antibiotic resistance in a population, while question 2 asks about the evolutionary loss of toxicity in a mushroom population. Completed pre- and post-test responses were obtained from 34 students for question 1 and from 36 students for question 2.
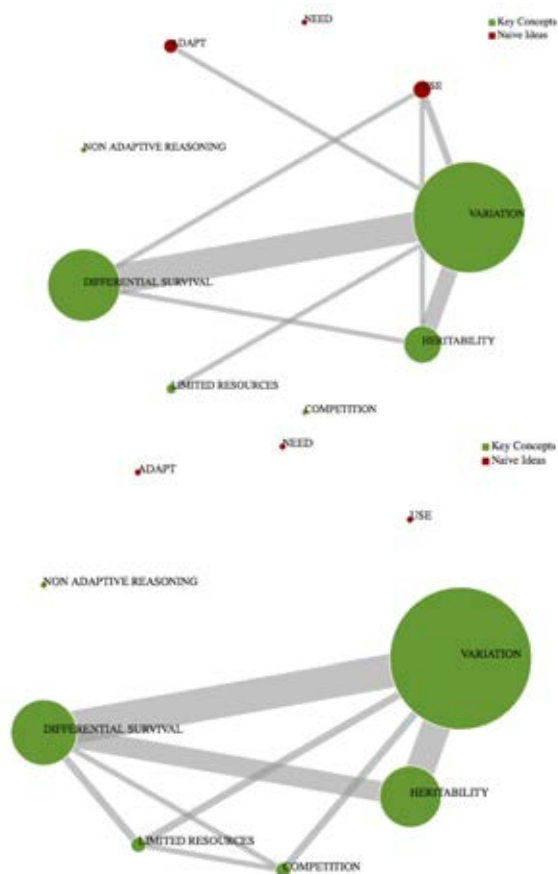
Figure 1: Concept maps produced by EvoGrader for the pre-instruction (upper panel) and post-instruction (lower panel) analysis of Question 1 (see Box 1). Sizes of the circles indicate percentage of responses scored as containing that concept; widths of the lines connecting concepts shows frequency of co-occurrence of those concepts.

Box 1

We evaluated student responses to two prompts:

Question 1: Explain how a microbial population evolves resistance to the effects of an antibiotic.

Question 2: A species of mushroom contains a chemical that is toxic to mammals. How would biologists explain the initial occurrence and increase in frequency of a number of individuals in the population that no longer produce this toxin?

We scored each response for whether it contained each of the following concepts:

Key Concepts:

- *Variation*: The presence and causes (mutation/recombination/sex) of differences among individuals in a population.

- *Heritability*: Traits that have a genetic basis and are able to be passed on from parent to offspring.

- *Competition*: A situation in which two or more individuals struggle to get resources which are not available to everyone.

- *Limited Resources*: Required resources for survival (food, mates, water, etc) which are not available in unlimited amounts.

- *Differential Survival*: Differential survival and/or reproduction of individuals.

- *Non-adaptive Ideas*: Genetic drift and related non-adaptive factors contributing to evolutionary change.

Naive Ideas:

- *Adapt*: Organisms/populations adjust or acclimate to their environment.

- *Need*: Organisms gain traits or advantage in response to a need or a goal to accomplish something.

- *Use/Disuse*: Traits are lost or gained due to use or disuse of traits.

Further, human evaluators determined whether or not a response answered the question asked; if the response did not, no credit was given for Key Concepts. For example, consider this student response:

> *Similar to above, some kind of mutation for the poison and those plants were not eaten so they were able to reproduce and pass thoses* [sic] *genes on to future generations. The population of poisonous mushrooms would soon outnumber non-poisonous ones since poisonous mushrooms are less likely to be eaten. Over time, animals would learn to stay away from teh* [sic] *mushroom simply be* [sic] *appearance, so the toxin would no longer be needed.*

Although this answer demonstrates adaptive reasoning about the origin of toxic mushrooms, the question was about the loss of toxin in this population, not the origin of the toxin. Only the last sentence addresses the loss of the toxin, and it does not demonstrate any of the Key Concepts.

Data

Data files containing all student responses, scoring, and data analysis may be found at `https://github.com/mjwiser/ALife2016`

Scoring responses

We used EvoGrader to score student responses on two open-ended questions about natural selection for six Key Concepts and three Naive Ideas (see Box 1). Two human graders (MJW and LSM) scored student responses for these same criteria. We resolved any disagreement among the humans by discussion, resulting in a consensus human score.

Statistical analysis

We measured inter-rater reliability (IRR) between the Evo-Grader scores and the consensus human scores for each question, as outlined in (Hallgren, 2012). Because we were interested in the IRR of specific questions, we combined both pre-and post-instruction responses into a combined data set. We computed IRR both for each question as a whole, and separately for the key concepts and the naive ideas within each question. We chose to not compute IRR for each individual concept, or separately for pre- and post-instruction questions, because of the lower statistical power from examining each set separately, and the increase in multiple comparisons this would necessitate. We also compared the EvoGrader and human consensus scores by way of 2-tailed paired t-tests to test for differences in the number of key concepts or naive ideas scored. We conducted all statistical testing in R version 3.2.3 (R Core Team, 2013).

Results and Discussion

The Inter-Rater Reliability (IRR) of EvoGrader and the consensus human scoring of these questions is good, with values of 0.63 for the antibiotic resistance question and 0.55 for the mushroom question (Fig. 2). This means that more than half of the total variance in scoring across these 9 concepts is shared among the raters. Landis and Koch (1977) suggest that IRR values from Cohens kappa in the range of 0.6 to 0.8 indicate substantial agreement among coders, and values between 0.4 and 0.6 indicate moderate agreement (Landis and Koch, 1977). By these criteria, when all of the concepts are analyzed together, the IRR for the antibiotic question is strong, and the IRR for the mushroom question is moderate.

We further examined IRR separately for Key Concepts and Naive Ideas (Fig. 3), to examine whether there was a systematic difference between the two concept types. In the antibiotic resistance question, the IRR is notably higher for the Key Concepts than the Naive Ideas (0.63 v 0.17). In fact, the 95% confidence interval for the Naive Ideas IRR overlaps 0, meaning that the IRR is not statistically significantly different from ratings being assigned at random. Conversely, IRR in the mushroom question is consistent across the Key Concepts and Naive Ideas (0.51 and 0.55, respectively), showing no meaningful difference across concept type.

What can account for these differences in IRR? One thing to take note of is that when there is very low variation in a given raters scoring across responses, there is very little statistical power to detect shared variance across raters. As a thought experiment, imagine that two different raters assign scores of Yes to 10% of responses, and No to 90%. Even if the two raters both assigned their scores randomly, the two raters would be expected to agree 82% of the time. IRR analyses take into account the expected frequency of scoring agreement, but a low variance across responses for a given rater will negatively affect the statistical power of IRR analyses. This is reflected in the wide confidence intervals for the Naive Ideas in particular. For one, there are fewer potential Naive Ideas scored (since there are at most three Naive Ideas per student response, while at most six Key Concepts per student response). This skew in responses had a larger impact on the Naive Ideas in the antibiotic resistance question than elsewhere; EvoGrader only scored the entire class as expressing five total Naive Ideas in the antibiotic question; the consensus human score was 90. This is part of a general trend: for both questions, the human consensus score differed from the EvoGrader score, and by a statistically significant margin even when correcting for multiple comparisons (see Table 1; all adjusted p-values <0.05). For both questions, the human consensus score detected more Naive Ideas than EvoGrader did. However, the humans detected more Key Concepts than EvoGrader did for the antibiotic question (question 1), but fewer in the mushroom question (question 2).

Several factors may serve to lower the IRR from ideal levels. One obvious cause is mentioned in Box 1: some student responses demonstrate reasoning about natural selection, but do not answer the question asked. In these cases, the humans did not credit the student with any of the Key Concepts that did not address the question asked. Evo-Grader, on the other hand, did not have this screening mechanism. Further, we analyzed both pre- and post-instruction responses jointly, and we expect the number of Naive Ideas expressed to decrease through instruction while we expect the number of Key Concepts expressed to increase through instruction. Such instructional effects would be a positive outcome for students, but both may reduce variance in the post-instructional scoring, reducing the statistical power to detect shared variance.

What can account for the difference in results between the two questions? There are two potentially salient contextual differences between the questions. One, the first question is a gain of a trait, while the second is a loss of a trait. Two, the two questions use different taxonomic groups as their examples. Both of these differences have been shown in the literature to be important to student reasoning (Nehm and
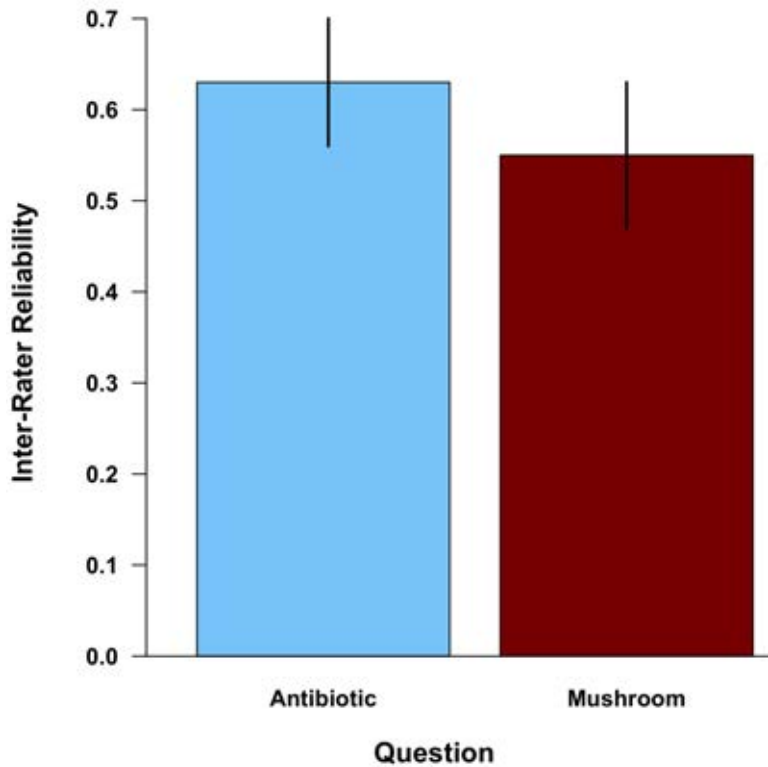
Figure 2: Inter-rater Reliability for Questions 1 and 2. Key Concepts and Naive Ideas are pooled within each question. Plotted values are Cohen's kappa. Error bars shown are 95% confidence intervals.

Comparison	t	df	p	adj. p
Antibiotic KC	5.779	67	$2.14 * 10^{-7}$	$8.58 * 10^{-7}$
Antibiotic NI	2.604	67	0.0113	0.0453
Mushroom KC	-2.806	71	0.00647	0.0259
Mushroom NI	3.384	71	0.00117	0.00466

Table 1: 2-tailed paired t-tests comparing EvoGrader and human consensus scoring of Key Concepts (KC) and Naive Ideas (NI). Negative values indicate more of these concepts detected by EvoGrader; positive values indicate more of these concepts detected by humans. A Bonferroni correction was used to generate the adj. p values.

Ha, 2011). In a future study, we will be able to disentangle these factors through a multifactorial design that considers multiple taxonomic groups and asks both a gain of trait and a loss of trait question within each.

Conclusions

EvoGrader is a useful tool for assessing student reasoning about natural selection. Even on questions not included in the training, it provides a reasonable level of reliability in scoring student responses on open-ended questions of a similar style to the ACORNS assessment. However, it is not
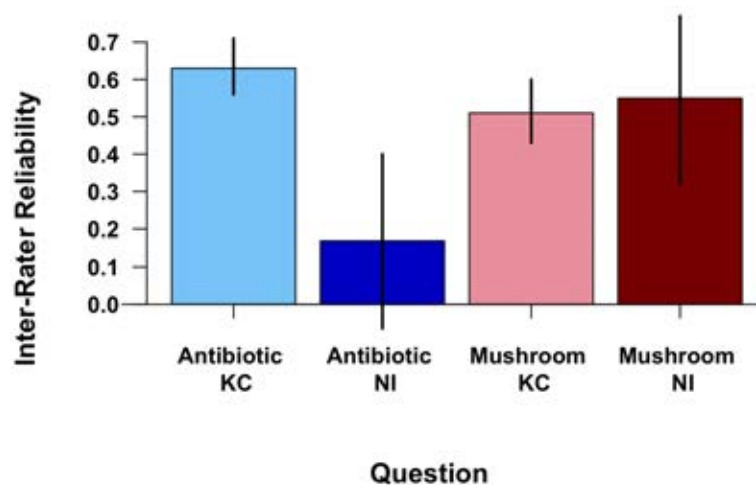
Figure 3: Inter-rater Reliability for Questions 1 and 2, broken down between Key Concepts (KC) and Naive Ideas (NI). Plotted values are Cohen's kappa. Error bars shown are 95% confidence intervals.

foolproof. In our study, EvoGrader credited students as displaying more Key Concepts, and fewer Naive Ideas, than our human raters did. In particular, EvoGrader may inaccurately credit student responses that do not address the specific question asked for evolutionary reasoning. For formative assessments, it can be a valuable tool to get a sense of student responses in a short period of time, but we caution against using EvoGrader to assign points to students, given its current limitations.

References

Anderson, D. L., Fisher, K. M., and Norman, G. J. (2002). Development and evaluation of the conceptual inventory of natural selection. *Journal of Research in Science Teaching*, 39(10):952–978.

Andrews, T. M., Price, R. M., Mead, L. S., McElhinny, T. L., Thanukos, A., Perez, K. E., Herreid, C. F., Terry, D. R., and Lemons, P. P. (2012). Biology Undergraduates' Misconceptions about Genetic Drift. *CBE-Life Sciences Education*, 11(3):248–259.

Bishop, B. A. and Anderson, C. W. (1990). Student conceptions of natural selection and its role in evolution. *Journal of Research in Science Teaching*, 27.

Brewer, C. A. and Smith, D. (2011). Vision and change in undergraduate biology education: a call to action. *American Association for the Advancement of Science, Washington, DC*.

Butler, A. C., Marsh, E. J., Slavinsky, J. P., and Baraniuk, R. G. (2014). Integrating Cognitive Science and Technology Improves Learning in a STEM Classroom. *Educational Psychology Review*, 26(2):331–340.

Hallgren, K. A. (2012). Computing Inter-Rater Reliability for Observational Data: An Overview and Tutorial. *Tutorials in quantitative methods for psychology*, 8(1):23–34.

Hiatt, A., Davis, G. K., Trujillo, C., Terry, M., French, D. P., Price, R. M., and Perez, K. E. (2013). Getting to Evo-Devo: Concepts and Challenges for Students Learning Evolutionary Developmental Biology. *CBE-Life Sciences Education*, 12(3):494–508.

Kidziński, Ł., Giannakos, M., Sampson, D. G., and Dillenbourg, P. (2016). A Tutorial on Machine Learning in Educational Science. In Li, Y., Chang, M., Kravcik, M., Popescu, E., Huang, R., Kinshuk, and Chen, N.-S., editors, *State-of-the-Art and Future Directions of Smart Learning*, pages 453–459. Springer Singapore, Singapore.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33.

Mayfield, E. and Rosé, C. P. (2013). Open Source Machine Learning for Text. *Handbook of automated essay evaluation: Current applications and new directions.*

Moharreri, K., Ha, M., and Nehm, R. H. (2014). EvoGrader: an online formative assessment tool for automatically evaluating written evolutionary explanations. *Evolution: Education and Outreach*, 7(1):1–14.

Nehm, R. H., Beggrow, E. P., Opfer, J. E., and Ha, M. (2012a). Reasoning about natural selection: diagnosing contextual competency using the ACORNS instrument. *The American Biology Teacher*, 74.

Nehm, R. H. and Ha, M. (2011). Item feature effects in evolution assessment. *Journal of Research in Science Teaching*, 48.

Nehm, R. H., Ha, M., and Mayfield, E. (2012b). Transforming biology assessment with machine learning: automated scoring of written evolutionary explanations. *Journal of Science Education and Technology*, 21.

Nehm, R. H. and Schonfeld, I. S. (2008). Measuring knowledge of natural selection: a comparison of the CINS, an open-response instrument, and an oral interview. *Journal of Research in Science Teaching*, 45.

R Core Team (2013). R: A language and environment for statistical computing.

Rector, M. A., Nehm, R. H., and Pearl, D. (2012). Learning the Language of Evolution: Lexical Ambiguity and Word Meaning in Student Explanations. *Research in Science Education*, 43(3):1107–1133.

States, N. L. (1900). *Next generation science standards: For states, by states*. National Academies Press.

Yahya, A. A., Osman, A., Taleb, A., and Alattab, A. A. (2013). Analyzing the Cognitive Level of Classroom Questions Using Machine Learning Techniques. *The 9th International Conference on Cognitive Science*, 97:587–595.

How Complexity Pervades Specialization in Canonical Embodied Evolution

Pedro Trueba[1], Abraham Prieto[1], Francisco Bellas[1] and Richard J. Duro[1]

[1]Integrated Group for Engineering Research
Universidade da Coruña, Spain
francisco.bellas@udc.es

Abstract

Embodied Evolution (EE) is an evolutionary strategy based on natural evolution in which the individuals that make up the population are embodied and situated in an environment where they interact in a local, decentralized and asynchronous fashion. It has been successfully applied in collective problems showing its validity to perform on-line evolution both in simulated and real agents. A key feature of EE is that of emergent specialization, that is, this strategy is able to autonomously generate a distribution of individuals into species if that is advantageous in the scenario. This paper goes in the line of studying such feature in more depth, analyzing how the complexity of the task (fitness landscape) and the complexity of the individuals (control system) affect the emergence of specialization. The analysis is carried out using a canonical EE algorithm in a real problem consisting in a collective surveillance task with simulated Micro Aerial Vehicles.

Introduction

As research has advanced in the field of evolutionary optimization, new approaches and techniques have allowed researchers to address even more complex problems. In this sense, a remarkable challenge is that of solving real-world dynamic problems in the absence of centralized and updated information. This type of problem appears in tasks like routing, surveillance, resource assignment, etc.

Some of the most successful approaches in this line are based on multi-agent systems that exploit the coordination between the agents to provide a collective solution to the problem (Hanna, 2009) (Rinde, 2012). This way, each agent is assumed to have decentralized and out-of-date information and, with it, it must handle its small part of the problem in real-time. The effort of the evolutionary algorithm is on finding a global coordinated solution to the problem as an aggregation of the partial ones, which can be really complex in dynamic environments.

With the aim of dealing with a more flexible and general search process, some authors have included the behavioral specialization of the agents as a new dimension of the collective problem. That is, the agents can be heterogeneous in operation, so specialists can emerge from evolution if they are beneficial for the task. It could seem that this new dimension increases the complexity of the search space, but it has been shown that allowing such flexibility simpler agents can emerge, which leads to a general simplification of the problem. It must be pointed out here that in this type of heterogeneous collective evolution, the solution to the problem is provided by the concurrent execution of the whole population, and not by a replication of the best individual.

The most remarkable evolutionary strategies one can find in heterogeneous collective optimization are Cooperative Coevolution Evolutionary Algorithms (CCEA) (Wiegand, 2003) (Panait, 2010). In this type of algorithms, the control system of each agent, typically an Artificial Neural Network, is evolved in an independent population, although the fitness of each individual is obtained by their joint execution with their team. Thus, if the solution is made up of n components, a CCEA evolves n populations, each one containing the genotype that will define the response of each component. Specific types of CCEAs like SANE (Gomez, 1997), Multi-agent ESP (Yong, 1999), CONE (Nitschke, 2012) or Hyb-CCEA (Gomes, 2015) have been widely applied with success in collective optimization of problems. The main problem of CCEAs for solving real-world dynamic problems is that they must run off-line due to their high computational requirements. Although the evolutionary processes can be executed in parallel, they must be serialized for evaluation, which must be performed several times in order to provide enough combinations of genotypes to achieve a reliable evaluation.

An online version of CCEA is the evolutionary strategy known as Embodied Evolution (EE). It was created by Ficici and Watson in 1999, inspired by Artificial Life experiments, with the aim of speeding up online evolution (Ficici, 1999). The main difference with traditional CCEAs is that evolution is at the population level in EE, that is, each individual only carries its own genotype. Consequently, EE follows a natural evolution scheme in which the individuals that make up the population are embodied and situated in an environment where they interact in a local, decentralized and asynchronous fashion. This interaction is not driven by a preset synchronization mechanism as in traditional evolutionary algorithms, but by the result of the particular behaviors of the active individuals in the environment and their interactions with other active and passive elements within it (Schut, 2009). Evolution in EE is open-ended, leading to a paradigm that is intrinsically adaptive and highly suitable for real time learning in distributed dynamic problems.

In the last decade, EE interest has grown mainly in the field of multi-robot systems (Bredeche, 2012) (Elfwing, 2011) (Eiben, 2010). As a consequence, different algorithms have arisen, which share the same operational principles but differ in how they implement specific operators. During subsequent

years, some of those algorithms were successfully applied to different collective problems (Bredeche, 2012) (Elfwing, 2011) (Prieto, 2010) (Duro, 2011), which allowed the validation of the paradigm in practical terms, but also the lack of a formal characterization to be considered by researchers in the evolutionary computation field became clear. As a first step towards this standardization of EE, a canonical EE algorithm that isolates its operational principles from those of the particular implementations was presented and thoroughly studied in (Prieto, 2015).

The current paper follows the line of EE formal characterization. In this case we are interested in analyzing in depth one of the main features of EE: the emergence of specialists (Nitchske, 2008). As shown in (Prieto, 2015) and (Trueba, 2013), in EE the optimal number of species emerge as required by the problem. But, what features of the problem determine the emergence of specialists? Can we anticipate the species that will arise or understand the reason why a given organization has emerged? These two questions are very relevant when developing an optimization algorithm. It is obvious that the spatial or temporal separation between individuals promotes specialization because it avoids mating (Trueba, 2013), but there are several more features. As it is well-known in fields like Complex Systems (Mitchell, 2008) and Ecology (Epstein, 1996), one of the most relevant factors is the complexity of the environment (Bonabeau, 1997) (Burbeck, 2007) and the complexity of the individuals (Anderson, 2001) (Detrain, 2002) that make up the population.

Thus, the question we aim to address here is how complexity affects the emergence of specialists in EE. We have analyzed this problem from two perspectives: varying the complexity of the environment where the agents are situated and varying the complexity of the agent itself, that is, its optimization capability. To carry out this analysis, we have applied the canonical EE algorithm in a collective surveillance task with simulated Micro Aerial Vehicles (MAVs), which is a prototypic example of dynamic and decentralized problem that must be solved in real time.

Canonical Embodied Evolution Algorithm

The canonical EE algorithm is explained with detail in (Prieto, 2015), and here we will provide just a brief summary of its main parameters and operation. During its development we decided to simplify and extract the barebones specification of a generalized EE algorithm in terms of as few parameters as possible. This generalization has been achieved by substituting the activation of particular operators triggered by events produced in the real problem with probability distributions, which are not task dependent. A second objective in the definition of a canonical EE algorithm was to do away with the bonds the environment imposes on the structure and operation of this type of situated algorithm. The adaptation to the environment determines the type of tasks that are considered tractable. It also makes the algorithm and its behavior even more task dependent, and as a result, it becomes more complicated to extract general conclusions from experiments. Consequently, the circumstantial/spatial interactions have been replaced in the canonical algorithm by stochastic variables, which follow probability functions. The canonical EE pseudo-code is the following:

1 *While* **simulation active** *do*
2 *Random creation of population*
3 *For* **each interaction**
4   *For* **each individual**
5   ***Assign fitness*** *– (Scenario interaction)*
6   *If random < P_{mating}*
7   ***Look for fertile partners*** *($P_{selection}$)*
8   ***Select a partner for recombination*** *($P_{elegibility}$)*
9   ***Generate offspring and store it*** *(P_{ls})*
10   *If random < $P_{replacement}$*
11   ***New individual*** ← *current offspring genotype*
11   ***Reset individual***
13 *End*

The canonical EE performs three basic processes: evaluation (line 5 in the pseudo-code), mating (lines 6 to 9) and replacement (lines 10 to 12). The processes and parameters that define the algorithm response are:

- *Mating selection*: it has been modeled as an event that is triggered by a uniform probability function that depends on a single parameter, the probability of mating for every time step (P_{mating}). This probability can be calculated based on the maximum number of mating one individual can perform, which can be assimilated to a "maximum tournament window size" (S_{max}), and on the maximum lifetime of an individual (T_{max}), which is the same for every individual:

$$P_{mating} = \frac{S_{max}}{T_{max}}$$

- *Selection policy:* the probability of being eligible as a candidate for mating ($P_{elegibility}$) is defined through a function (ψ_e) that is based on three different criteria: the genotypic distance (φ_{cand}), a distance measure between certain status parameters (P_{cand}) which vary during evaluation time due to both the phenotype of each individual and environmental circumstances (for example, position in a geometric space) and the fitness value (γ_{cand}):

$$P_{eligibility} = \psi_e(\varphi_{cand}, \gamma_{cand}, [P_{cand}])$$

- *Genotypic recombination*: as in natural evolution, two main recombination operators can be distinguished: mutation and crossover. In order to characterize this genotypic recombination in a general and simple way, a new intrinsic parameter is defined: the probability of using a local search strategy (P_{ls}), that is, a mutation operator. It is a measure of the exploration and exploitation balance through the ratio between crossover and mutation frequency.

- *Replacement*: the replacement process is modeled here as triggered by a replacement probability ($P_{replacement}$) and it is defined based on a more intuitive and manageable parameter, which is the life expectancy (T_{exp}):

$$P_{replacement} = \frac{1}{T_{exp}}$$

The life expectancy is defined for each individual in each time step based on its current fitness (Q_i), which depends on its genotype and the genotypes of the others. Specifically, a piecewise function has been defined to assign the life expectancy (T_{exp}) for any fitness value using a linear model. This function has two pieces, one for individuals with low fitness ($Q_i < 2/3Q_{max}$) and another for individuals with high fitness ($Q_i > 2/3Q_{max}$):

$$T_{exp} = T_{mat} + \frac{Q_i C_m (T_{max} - T_{mat})}{2/3Q_{max}} \text{ if } Q_i \leq \frac{2Q_{max}}{3}$$

$$T_{exp} = T_{mat} + C_m(T_{max} - T_{mat}) +$$
$$\frac{(1 - C_m(T_{max} - T_{mat}))}{1/3Q_{max}}(Q_i - 2/3Q_{max}) \text{ if } Q_i > \frac{2Q_{max}}{3}$$
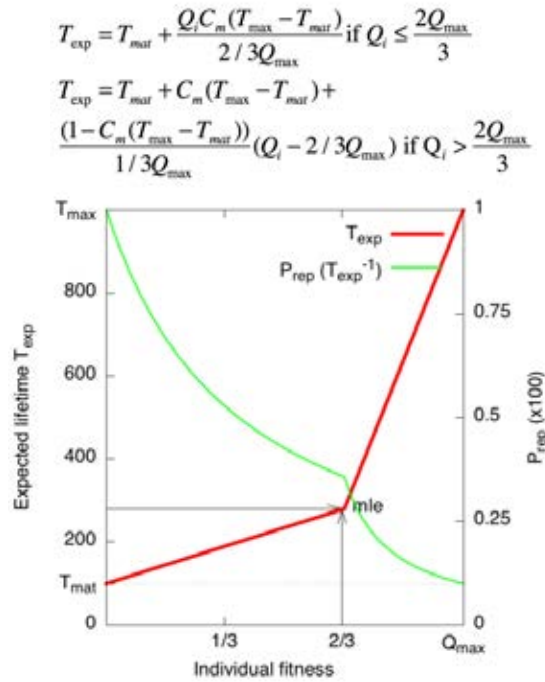
Fig. 1. Graphical representation of the function that parameterizes replacement

The use of this function to model replacement allows covering a broad range of different replacement policies, ranging from those with synchronous replacement ($T_{mat} = T_{max}$ and $C_m = 1$) to those with a strongly exploitative operation through elitism ($T_{mat} = 1$ and $C_m = 0$) by way of any intermediate combination. As it can be observed in figure 1, the relation between fitness and life expectancy has been modeled by means of four fixed parameters:

1. *Maturation time (T_{mat}):* it is defined here as a percentage of T_{max} and it sets the minimum T_{exp} for an individual and, as a consequence it captures the sum of two time periods, the minimum time required to reliably evaluate an individual (1 in our case) and the life-time for those with zero fitness. During the maturation time the individual cannot be replaced.
2. *Maximum current fitness (Q_{max}):* since the maximum fitness one individual can achieve for a specific scenario is not known beforehand, this value represents the maximum known fitness so far, and it is used to calculate a relative fitness $Q_r = Q_i/Q_{max}$.

3. *Mediocrity coefficient (C_m):* it establishes the expected lifetime for the individuals with a relative fitness value equal to 2/3 ($Q_r = Q_i/Q_{max} = 2/3$) of the maximum current fitness (it can be seen as the expected lifetime of the mediocre individuals). This parameter ranges from 0 (meaning their life expectancy will equate T_{mat}) to 1 (meaning it will equate T_{max}).
4. *The maximum lifetime (T_{max}):* it is the maximum number of time steps a chromosome can participate in the evolution. It is assigned to individuals with their fitness equal to the maximum current fitness.

As a summary, six intrinsic parameters and an eligibility function have been defined to encompass and generalize the operation of a general EE algorithm: T_{max}, S_{max}, P_{ls}, Q_{max}, T_{mat}, C_m and ψ_e. See (Prieto, 2015) for a sensitivity analysis of the canonical EE algorithm.

To apply the canonical EE algorithm to real problems, two operators have to be adapted to the constraints the scenario imposes, and therefore, their dependence on the task is unavoidable: the mating operator, which is constrained by the communication limitations of the scenario and individuals to exchange their genetic codes, and the evaluation operator, which relies on the actual behavior and on the state of the scenario.

Collective Surveillance Task

To analyze how complexity affects the emergence of species using the canonical EE, we have designed a simulated environment in which a fleet of Micro Aerial Vehicles (MAVs) has to collectively survey an indoor scenario where there is not centralized information available. To do it properly, the MAVs need to locate themselves to keep track of their trajectories and to share this information with other robots. The determination of their positions will be performed using their IMU, artificial landmarks that can be sensed using the onboard camera, and the position of other MAVs in sight. The control of each of the MAVs is provided by an Artificial Neural Network (ANN), and the parameters of this ANN will be adjusted using the canonical EE algorithm. Thus, EE is in charge of organizing the MAVs in the scenario in order to increase the accuracy of the fleet location, and consequently, the speed at which a new point of interest is reached.

Experiment description

The experimental setup has been defined in simulation, based on a real indoor gathering task performed by MAVs as a previous step to translate the algorithm or directly the controllers to a fleet of real MAVs. The specific MAV that has been modeled is the Parrot ARDrone 2.0, a very popular general-purpose commercial quadcopter. Indoor navigation is performed then by simulated ARDrones that, as in the case of the real ones, are provided with an IMU and a camera to perform autonomous positioning. The most important aspect of the simulation is the model of the response of the location sensors when a certain maneuver is carried out. Firstly, the IMU will provide some velocity signals for each degree of freedom, which has to be integrated to produce the estimated motion. The estimation of the velocity is modeled as subject

to a normal distribution centered on the real input velocity with its corresponding variance matrix as is frequently assumed on real navigation.

The model implemented for the artificial markers represents the use of the AprilTags created by The APRIL Robotics Laboratory at the University of Michigan (Olson, 2011) since it is also what we have used during our tests with the real ARDrones. Therefore, the location estimation provided by the markers is based on a real accuracy model which was produced in our laboratory using an ARDrone 2.0 and 40 cm long AprilTags, and that can be formulated as a function L_e which relates the variance of the estimation $Var(\vec{p}_{drone})$ with the relative distance ($\|\vec{p}_{drone} - \vec{p}_{tag}\|$) and orientation ($yaw_{drone:tag}$) between camera and tag:

$$Var(\vec{p}_{drone}) = \boldsymbol{L_e}(\|\vec{p}_{drone} - \vec{p}_{tag}\|, yaw_{drone:tag})$$

These tags (*permanent tags*) provide an absolute and potentially accurate position estimation, which does not degrade with time. In order to improve the performance of the navigation by improving the accuracy of the MAVs the same type of tags are attached to the body of the quadcopters (*mobile tags*), which will make up a hybrid location sensor. Therefore, the detection provided by a mobile tag still constitutes a direct location estimation but, unlike in the case of permanent tags, their accuracy is variable and it is modelled as proportional to the velocity of the MAV's. As a consequence, those MAVs which will serve as mobile tags will have to stay still to be able to provide location accuracy. The use of this type of mobile tags leads to accuracy becoming a resource that MAVs can get and share to be able to slow down its degradation, and therefore, to accomplish their main task more efficiently. It could also allow some of the MAVs not to require visits to static tags, which are frequently non optimally located, in order to improve their location accuracy, being 'nourished' by mobile tags.
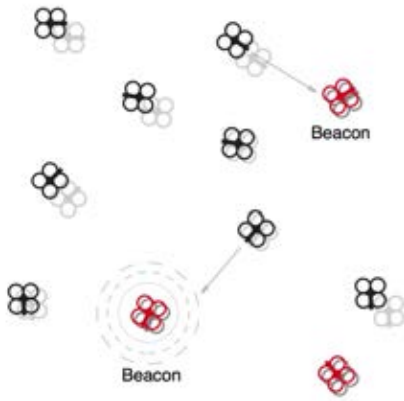
Fig. 2. Graphical representation of a portion of the scenario. The distance to each own shadow represents the current estimation error for the MAV. The red MAV represents a MAV acting as a mobile tag. The circles around the tags represent the different levels of accuracy provided by the tag.

Side of the arena (L)	768
Total area	L^2
Fixed tags detection range	L/4
Mobile tags detection range	L/16
Max velocity (V_{max})	L/50
Standard deviation for the velocity	$V_{max}/50$
Max accuracy provided by an April Tag	L/10

Table 1. Design parameters of the simulated scenario

The scenario was discretized to reduce computational effort as a 768 x 768 (square length units) non-toroidal square arena, which is provided with 4 fixed tags placed randomly. Figure 2 shows a schematic representation of a portion of the arena. Each MAV is associated to both a real and an estimated position. The former is shown with a solid color in the simulation and the later with a softened shadow of the MAV. The MAV has no idea of the real position, this is just an externally obtained value for display purposes. The further the distance between those positions, the higher the location estimation error and the lower the exploration level. Table 1 contains the specific parameters that define the scenario.

The final objective of the surveillance task is for the fleet of MAVs is to continuously cover the maximum possible area. In order to perform the search of unexplored areas, the MAVs can keep record of the areas they have already explored. This is stored in an *'exploration map'* carried by each MAV and that can also be shared with others when they meet. The exchange of this information allows the task to be solved cooperatively since it enables the distribution of the search among the group of MAVs. However, since the estimation of one's position has a varying accuracy, the updating of the exploration map has to take that into account. The exploration of a cell is modeled as an *exploration probability* (P_{ex}), which indicates the probability that a certain cell has of having been explored. This probability (P_{ex}) can be directly calculated as the ratio between the size of a cell (L_{cell}) and the location error range (E_{loc}) and is stored in the exploration map:

$$P_{ex} = \frac{L_{cell}^2}{\pi \cdot E_{loc}^2}$$

Subsequent MAVs must decide whether or not to *re-explore* that cell based on the guaranteed exploration probability P_{ex}. Therefore, the collaborative exploration map is the only information that a MAV gets from the scenario about the surveillance process. The individual fitness for each agent increases each time it covers and unexplored cell. The global fitness for the multi-agent system, which must be optimized by the canonical EE algorithm, is the sum of exploration probability for all the cells *j* in the scenario:

$$G = \sum_j P_{exj}$$

It must be highlighted that the MAVs are not transparent to each other and can collide with others and with the walls. When a MAV collides, it loses part of its location accuracy. To avoid collisions, they are provided with an obstacle sensor which mimics an infrared ring that detects near obstacles.

Individual encoding

The control architecture for each agent is a multilayer perceptron ANN with three layers: one input layer with three neurons, one hidden layer with a configurable number of neurons and one output layer with one neuron. The first input is the *current exploration capability* of the agent, which measures the exploration capability and it is based on its location error. The second input provides the *maximum attainable exploration* in the surrounding areas and the direction towards it. The third input provides the *distance to the closest obstacle* (up to the sensing range of the sensor) and the direction towards it. The size of the intermediate layer will be varied for different experimental configurations to analyze the complexity of the behavior of the agent, from 1 neuron in the simplest case to 10 neurons in the one with highest complexity. The output of the neural network modulates the behavior of the MAV between four pre-learned basic behaviors, namely: *move towards the most unexplored area detected*, *move towards the nearest tag*, *move against the closest obstacle* and *stay still to serve as a dynamic tag*. These behaviors are individually selected for each time step, the frequency with which each agent will display each behavior will determine the agent species.

Iterations	20.000
Population size	40
Maximum lifetime (T_{max})	1000
Maturity time (T_{mat})	25
Selection criteria (ψ_e)	Higher fitness (γ_{cand})
Tournament selection size (S_{max})	40
Local search probability (P_{ls})	1.0
Mediocrity coefficient (C_m)	1.0
Maximum current fitness (Q_{max})	Automatic
Chromosome length	{4,8,40} x [0,1]

Table 2. Parameters of the canonical EE algorithm

The implementation of the canonical EE algorithm used here follows the pseudocode presented in section 2, while the specific values for the parameters are those shown on Table 2. These values were selected according to the conclusions extracted from (Prieto, 2015).

Results

This section will show the results obtained for different configurations of the scenario in order test how different variations both in the complexity of the individuals and in the complexity of the environment affect the outcome of the evolution. The complexity of the individuals has been modified by changing the size of the ANN which controls the MAV. The complexity of the environment has been changed in two ways. First, by including more or less permanent tags, which will make mobile tags more necessary (or indispensable if there isn't any permanent tag) as the number of permanent tags decreases. Second, the environment is also adjusted by modifying the impact of the collisions on the degradation of the accuracy of the MAVs, which will make

their navigation more or less complex (from neglecting the obstacle sensor to rigorously avoiding obstacles).

To clarify the canonical EE response in this type of collective optimization problem, we will describe first the emergence of specialization in a representative run performed using the following experimental configuration: no permanent tags, an ANN with 4 weights (3x1x1) and no penalty for collisions. The top plot of figure 3 shows the evolution of the global fitness of the population during 20000 iterations while the bottom one displays the average number of species per iteration (N_s). This number of species was calculated using the metric defined by the Davies-Bouldin Index (DBI) (Davies, 1979) applied to a k-means clustering algorithm. Since that metric does not consider the existence of a single cluster, an auxiliary species of ten individuals is always included in the data to be clustered. The number of species (N_s) is obtained using the average of each possible number of clusters (from 1 to 10) weighted by the inverse of the DBI associated to them (all possible numbers of clusters), so that the strong discretization of the low number of species is avoided and the measurement is more accurate. To account for the inclusion of an auxiliary species the final result is obtained by subtracting one from the weighted average, that is:

$$N_s = \left(\sum_{i=1}^{10} i * DBI(i)^{-\gamma} \right) - 1 \qquad with\ \gamma = 2$$

The parameter γ provides a smoothing coefficient, the lower the coefficient the more discrete the result.
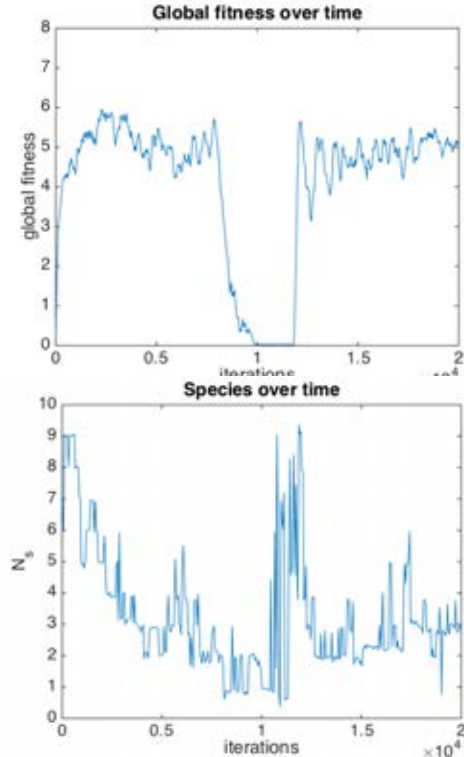
Fig. 3. Evolution of the global performance of the population (top) and average number of species (bottom) during 20000 evolution steps

As it can be observed in the top plot of figure 3, in this run, the population starts with low performance but it quickly achieves a successful global fitness level (less than 300 iterations). As demonstrated in several previous works (Prieto, 2010) (Duro, 2011), a remarkable strength of EE is the achievement of satisfactory results after few time steps. The number of species also converges to a value around 2, which can be observed in the bottom plot of figure 3.
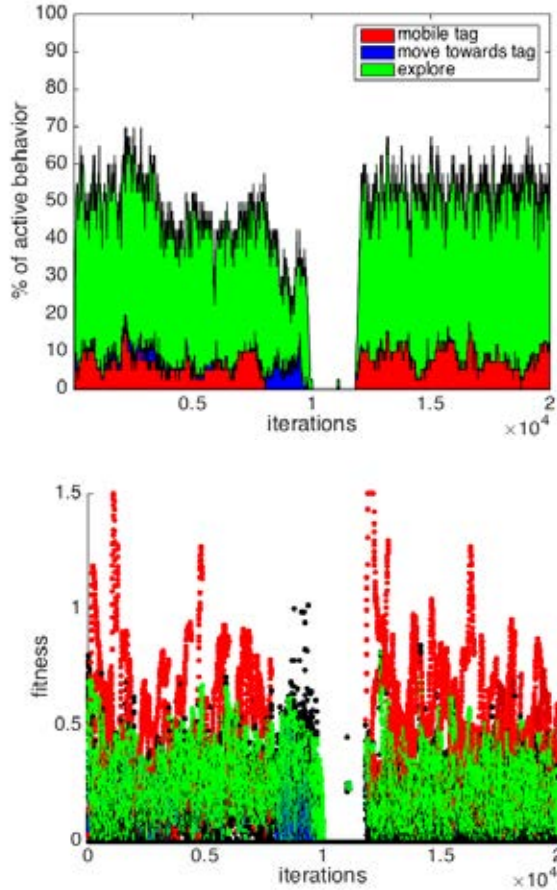
Fig. 4. Frequency of activation of each pre-learned behavior for the whole population (top) and individual fitness of each of the 40 individuals of the population (bottom)

The top plot of figure 4 shows the frequency of activation of each pre-learned behavior for the whole population, and the bottom one shows the individual fitness of each of the 40 individuals of the population. The coloring code is the same for all the plots in figure 4 and figure 5, which depicts individual parameters: red for those individuals with predominant *become mobile tag* behavior, green for *exploring the surrounding area*, blue for *moving towards the closer tag* and yellow for *avoiding obstacles* (although not present in this run since the penalty is set to zero). It can be seen in the top plot of figure 4 that up to iteration 8000, two main species have emerged: become *mobile tag* (red) and *exploring* the surrounding area (green). As shown in the bottom plot, being a mobile tag provides a higher individual fitness to the agents.

After around 8000 iterations, a period starts in which the population gets destabilized, which is something that happens often in a co-evolutionary process, and the global performance drops quickly (see figure 3 top). This run was indeed selected to illustrate both the dynamism of this type of evolution and the capabilities of the algorithm to recover from this period to achieve again a successful performance. As shown in the top plot of figure 4, in this unstable period, the number of species decreases to almost one since the *mobile tag* species (red) gets extinguished. Moreover, there is an increment in the *go to tag* species seeking accuracy (blue), but it is useless since there are no available tags any more. A few iterations before 10000 only *explorers* survived trying to use the little remaining accuracy to explore an environment which is becoming more and more uncovered. It takes the algorithm around 2000 more time steps of low quality individuals (all of them are condensed in a black line shown in the bottom part of the bottom plot of figure 4) to finally produce the two required species to successfully explore the environment and to rise the global fitness again to a successful level (become *mobile tag* and *exploring* the surrounding area).

Figure 5 contains a representation of the genes of the population along this evolution. To do so, each genotype is projected into two independent dimensions. Those individuals that live less than 100 time steps are depicted in black meaning that there was no time to consistently assign a behavior to them. As it can be observed in figure 5, during the instability gap (iterations 8000 to 12000), the genomes of the population expand along the genetic state space, producing a higher number of ephemeral species that, just before the recovery (iteration 12000), decrease quickly to converge again to a configuration of two main species.
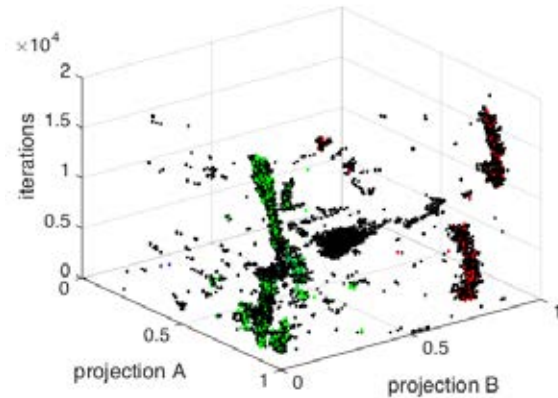
Fig. 5. Representation of the genes of the population along the evolution where each genotype has been projected into two independent dimensions.

Figure 6 displays 6 plots where the correlation between performance and number of species has been studied for a set of configurations that vary the environment and individual complexities. The size of the neural network was set to 3x1x1 (4 weights), 3x2x1 (8 weights) and 3x10x1 (40 weights) and it is displayed in the top, middle and bottom rows in figure 6. Moreover, the environment was set with and without permanent tags (left and right columns in figure 6), and with

and without obstacle penalties (red and blue lines in all the plots). The data displayed in this figure were obtained after 5 runs of 20000 iterations each.

The comparison shows several aspects of this interaction between complexity and the outcome of the co-evolutionary process. In the case of a scenario without permanent tags and without penalties for collisions (left column, blue lines), as expected, the simplest controller (4 weights) tends to produce two or three species to optimize its behavior (peak value of the blue curve shown in top-left plot). If the number of weights of the ANN is doubled (middle-left plot, blue line) then we can see that efficient solutions can be found with several configurations of species, with a slight tendency towards one species configurations, which indicates a greater robustness and versatility. However, if we set an ANN with up to 40 weights (bottom-left top, blue line), we observe a decrease in the overall performance and now the system tends to avoid only one species configurations. After studying different runs with this configuration (40 weights), it can be seen that it is harder for the algorithm to find a solution, and also to converge to only one species in such a high dimensional search space, and that some of the runs provide poor performance.
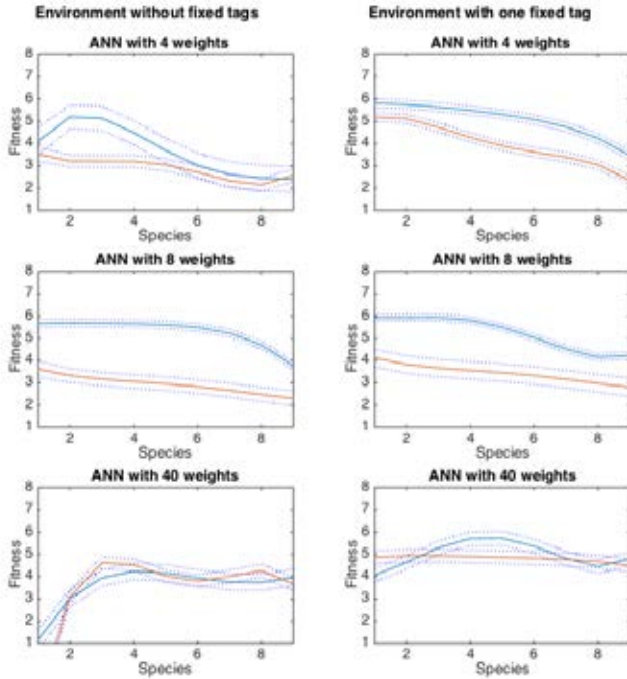
Fig. 6. Correlation between performance and number of species for a set of configurations that vary the environment and individual complexities. Top plots correspond to 3x1x1 (4 weights) ANN, middle ones to 3x2x1 (8 weights) and bottom ones to 3x10x1 (40 weights). Left plots do not have permanent tags while right ones do. Finally, the red lines correspond to executions with obstacle penalties while the blue lines correspond to executions without them.

When we activate the penalties for collisions (left plots, red lines), there is a higher complexity for navigation, and consequently the performance is much lower for 4 and 8

weights (top and middle plots). However, in the case of 40 weights it does not seem to affect greatly, which indicates that, although this configuration is harder to adjust, it is less affected by the complexity of the environment. On the other hand, when the environment becomes simpler because the permanent tags are included (right column plots), the evolution, in almost all the configurations, tends to create only one species which is able to perform successfully. Again, if the penalties are included (red lines), the performance decreases but the tendency regarding the species remains the same. Finally, the controller with 40 weights becomes, again, much less affected by the penalties and exhibits also again a tendency to avoid one species configurations.

Summarizing, for this experimental setup an intermediate complexity of the ANN (8 weights) has shown to be beneficial both in terms of performance and in stability regarding number of possible species. Using complex individuals or very simple ones led to a more unstable response of the algorithm. In the case of the more complex controller it is more complicated to evolve, but it is less affected by the complexity of the scenario. Finally, simpler scenarios tend to create one species configurations for most of the controllers.

In terms of the response of the algorithm, its capability to adapt the behavior and structure of the population to different individual and scenario configurations regarding their level of complexity has been shown. The algorithm is able to take advantage from the heterogeneity of the population when it is required to achieve a good global performance by making use of the specialization, as it has been constantly observed in natural evolution in real ecosystems (colonies of ants, termites or bees). It also tends to simplify the structure of the population when the scenario is simple in relation to the complexity of the individuals due to the evolutionary cost of adjusting a large genome. Interestingly, if the controller shows a high complexity the algorithm fails to produce an efficient population in some runs but when it doesn't it is more robust to changes in the complexity of the task.

Conclusions

This paper has studied the impact of complexity in the emergence of collective solutions to distributed problems by means of an implementation of Embodied Evolution for a collective surveillance task with simulated Micro Aerial Vehicles. The algorithm used, canonical Embodied Evolution, has already been tested for several applications and has shown its capability to decompose a task into several subtasks and to improve performance by generating different species among the individuals of the population. This work has presented a first attempt to deepen in the mechanisms that affect the creation of those species when the whole population pursues a common goal. The study has shown a strong impact on the outcome of the algorithm of variations in the complexity of the definition of the problem at different levels, namely, the complexity of the scenario and the complexity of the individuals. In terms of the response of the algorithm, the algorithm has found efficient groups of controllers for different individual and scenario setups regarding their level of complexity. As in the case of natural evolution, which

embodied evolution approaches mimic, the algorithm works with, and is able to take advantage of, heterogeneous populations when they are required to achieve a good global performance by making use of specialization. Our ongoing work, based on the results presented in this paper, is focused on including the number of parameters of the controllers as an individual evolvable parameter. The main goal will not be to obtain an optimal value for the size of the neural network of the whole population, since this can already be analyzed from the current results, but to study the dynamics produced when populations can be heterogeneous in both individual behavior and complexity.

References

Anderson C., McShea D.W. (2001), Individual versus social complexity, with particular reference to ant colonies, *Biological reviews of the Cambridge Philosophical Society. 76(2),* pages 211-37.

Bonabeau, E., Theraulaz, G., Deneubourg, J.L., Aron, S. Camazine, S. (1997) Self-organization in social insects *Trends in Ecology and Evolution*, Vol. 12, No. 5. pages 188-193

Bredeche, N., Montanier, J.M., Liu, W., Winfield, A. (2012) Environment-driven Distributed Evolutionary Adaptation in a Population of Autonomous Robotic Agents, *Mathematical and Computational Modelling of Dynamical Systems 18, 1*, pages 101-129

Burbeck, S. (2007), Complexity and the Evolution of Computing: Biological Principles for Managing Evolving Systems**,** *Technical Report* (http://www.evolutionofcomputing.org)

Corchado, J., Bajo, J., Kozlak, J., Pawlewski, P., Molina, J., Gaudou, B., Julian, V., Unland, R., Lopes, F., Hallenborg, K., Garcia, P. (2014) Highlights of Practical Applications of Heterogeneous Multi-Agent Systems, *the PAAMS Collection: PAAMS 2014 International Workshops,* Springer Publishing

Davies, D., Bouldin, D. (1979) A Cluster Separation Measure, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. PAMI-1 (2), pages 224–227.

Detrain, C., Deneubourg, J.L. (2002), Complexity of environment and parsimony of decision rules in insect societies, *Biological Bulletin 202(3),* pages 268-74.

Duro, R.J., Bellas, F., Prieto, A., Paz-López, A. (2011) Social Learning for Collaboration through ASiCo based Neuroevolution, *Journal of Intelligent and Fuzzy Systems 22*, pages 125-139.

Elfwing, S., Uchibe, E., Doya, K., Christensen, H. (2011) Darwinian embodied evolution of the learning ability for survival, *Adaptive Behavior 19, 2*, pages 101-120

Epstein, J., Axtell, R. (1996) Growing Artificial Societies, *The MIT Press*

Eiben, A.E., Haasdijk, E., Bredeche, N. (2010) Embodied, On-line, On-board Evolution for Autonomous Robotics, *Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution*, pages 361-382 Springer

Ficici, S., Watson, R, Pollack, J. (1999) Embodied Evolution: A Response to Challenges in Evolutionary Robotics. *Eighth European Workshop on Learning Robots*, pages 14-22

Gomes, J., Mariano, P., Christensen, A.L. (2015) Cooperative Coevolution of Partially Heterogeneous Multiagent Systems. *In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15)*, pages 297-305.

Gomez, F. and Miikkulainen, R. (1997) Incremental evolution of complex general behavior, *Adaptive Behavior, 5*, pages 317–342.

Hanna, L., and Cagan, J. (2009) Evolutionary Multi-Agent Systems: An Adaptive Approach To Optimization In Dynamic Environments, *ASME Journal of Mechanical Design*, Vol. 131, No. 1, pages 1-11

Nitschke, G.S., Schut, M.C., and Eiben, A.E. (2008) Emergent Specialization in Biologically Inspired Collective Behavior Systems. *Chapter in A. Yang and Y. Shan (eds.), Intelligent Complex Adaptive Systems,* pages 215-253, IGI publishing.

Nitschke, G.S., Schut, M.C., and Eiben, A.E. (2012) Evolving behavioral specialization in robot teams to solve a collective construction task, *Swarm and Evolutionary Computation, Volume 2*, pages 25-38

Mitchell, M. (2011) Complex Systems: a guided tour, *Oxford University Press*

Olson, E. (2011) AprilTag: A robust and flexible visual fiducial system. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA),* pages. 3400-3407

Panait, L. (2010) Theoretical convergence guarantees for cooperative coevolutionary algorithms. *Evolutionary Computation, 18(4)*, pages 581–615, MIT Press

Prieto, A., Becerra, J.A., Bellas, F., Duro, R.J. (2010) Open-ended Evolution as a means to Self-Organize Heterogeneous Multi-Robot Systems in Real Time, *Robotics and Autonomous Systems, vol. 58*, pages 1282-1291

Prieto, A., Trueba, P., Bellas, F., Duro, R.J. (2015) Towards the standardization of distributed Embodied Evolution, *Information Sciences, vol 312*, pages 55-77

Rinde R.S. van Lon, Tom Holvoet, Greet Vanden Berghe, Tom Wenseleers, and Juergen Branke. (2012) Evolutionary synthesis of multi-agent systems for dynamic dial-a-ride problems. In *Proceedings of the 14th annual conference companion on Genetic and evolutionary computation* (GECCO '12), pages 331-336, Terence Soule (Ed.)

Schut, M.C, Haasdijk, E., Prieto, A. (2009) Is situated evolution an alternative for classical evolution?, *Proceedings of the Eleventh conference on Congress on Evolutionary Computation*, pages 2971 – 2976

Trueba, P., Prieto, A., Bellas, F., Caamaño, P., Duro, R.J. (2013) Specialization analysis of embodied evolution for robotic collective tasks, *Robotics and Autonomous Systems, Volume 61, Issue 7*, pages 682-693

Wiegand, R. P. (2003) An Analysis of Cooperative Coevolutionary Algorithms. PhD thesis, George Mason University

Yong, C. H. and Miikkulainen, R. (2009) Coevolution of role-based cooperation in multiagent systems. *IEEE Transactions on Autonomous Mental Development, 1(3)*, pages 170–186

Distributed Age-Layered Novelty Search

Babak Hodjat[1], Hormoz Shahrzad[1], and Risto Miikkulainen[1,2]

[1]Sentient Technologies, Inc.
[2]The University of Texas at Austin
babak,hormoz,risto.miikkulainen@sentient.ai

Abstract

Novelty search is a powerful biologically motivated method for discovering successful behaviors especially in deceptive domains, like those in artificial life. This paper extends the biological motivation further by distributing novelty search to run in parallel in multiple islands, with periodic migration among them. In this manner, it is possible to scale novelty search to larger populations and more diverse runs, and also to harness available computing power better. A second extension is to improve novelty search's ability to solve practical problems by biasing the migration and elitism towards higher fitness. The resulting method, DANS, is shown to find better solutions much faster than pure single-population novelty search, making it a promising candidate for solving deceptive design problems in the real world.

Introduction

Novelty search is a new approach to population-based search motivated by the creativity and diversity of biological evolution (Stanley and Lehman, 2015; Lehman and Stanley, 2010a). Instead of optimizing a fitness objective, novelty search maximizes phenotypical diversity. Individuals are seeking novel niches to fill, developing emergent problem-solving abilities in the process. Novelty search is particularly powerful in domains that are deceptive, where it is necessary to discover low-fitness stepping stones first before actual solutions can be reached. Many tasks in artificial life are deceptive in this way, including behaviors that require developing learning, memory, and communication abilities (Lehman and Miikkulainen, 2014). Novelty search is thus a promising approach to constructing complex behavior in artificial life domains.

This paper aims to improve novelty search as a general problem-solving method in two ways. First, a distributed version of novelty search is developed. The idea is that search progresses in parallel in separate islands, and the best (i.e. most novel) individuals are periodically exchanged between them. Such a distribution is motivated by biological evolution, potentially leading to an implementation that can account for biological phenomena more accurately. It is also a well known diversity-maintenance technique in evolutionary algorithms in general (Whitley et al., 1999). However, the main motivation in this paper is to make novelty search computationally more powerful. Distribution makes it possible to scale novelty search to a much larger pool of individuals, and to take advantage of diversity between different novelty search runs. It also makes it possible to harness available parallel computing resources to serve novelty search. It therefore makes it possible to use novelty search to solve harder problems faster.

Second, a principled way of guiding novelty search towards high-fitness solutions is developed. Novelty is still the primary selection mechanism, but fitness is used to bias the process in two ways: (1) by migrating only the most fit individuals across the islands, and (2) by selecting the more fit of two similar individuals into the elitist pool. Such subtle biases do not prevent novelty search from creating and retaining novel individuals, but they make it more likely to create individuals that are solutions to the given problem.

Together these two extensions result in a powerful version of novelty search that can be used to solve difficult design problems in the real world. As a demonstration, in this paper they were implemented in an existing distributed evolution system called EC-Star (O'Reilly et al., 2013). This hub-and-spoke architecture manages a number of clients running separate evolutionary searches. A key feature of EC-Star is age-layering (Hodjat and Shahrzad, 2013): candidates are first evaluated in a small number of samples, and if they are promising, with more samples. Age-layering makes evolution more efficient, decreasing run times an order of magnitude or more (Shahrzad et al., 2016). It is also well-suited for evaluating novelty across many separate novelty searches.

The resulting method, Distributed Age-Layered Novelty Search, or DANS, is demonstrated on two challenging engineering design tasks: the 11-bit multiplexer, and the eight-input sorting network. The results show that DANS can find better solutions much faster than single-population novelty search. It is therefore a promising artificial life approach for solving deceptive problems in the real world.

Background and Related Work

The idea of divergent search, of which novelty search is an example, is first discussed, followed by existing work on combining novelty with fitness. The distributed evolution platform of EC-Star with age-layering, used to implement DANS, is then reviewed.

Objective vs. Divergent Search

In the traditional objective-based search, a population of candidate solutions are evolved to maximize a specific measure, or objective, called a fitness function. Individuals are selected for reproduction, and offspring are accepted into the population, if they score high in that function. The idea is that evolution thus gradually discovers better and better individuals, until it finds some that optimize the chosen objective.

Even though objective-based search is effective in many cases, there are two problems with it that are especially relevant in artificial life. First, it is not always clear how the objective function should be defined. The desired behavior may consist of many aspects that interact (such as speed, energy consumption, effectiveness, quality of the result), and some of them may be difficult to express formally (such as believability, creativity, elegance). Second, the domains are often deceptive, i.e. optimization requires creating individuals that do not perform well, but can serve as stepping stones in constructing those that do. This effect can be seen in many cognitive tasks that require memory, learning, or communication (Lehman and Miikkulainen, 2014), but it is also clear in the process of creating interesting images (Secretan et al., 2011).

Divergent search methods have recently emerged, mostly in the field of artificial life, as a potential solution to these issues. The idea is not to incrementally approach an optimum of a specified fitness function, but instead create as much diversity in the search as possible. This idea has been expressed in several forms, including empowerment (Salge et al., 2013), entropy maximization (Wissner-Gross and Freer, 2013), and behavioral diversity (Mouret and Doncieux, 2012). The particular formulation used in this paper is novelty search (Stanley and Lehman, 2015; Lehman and Stanley, 2010a), where individuals are rewarded based on how different they are from other individuals encountered so far. The novelty ρ for an individual x is defined as

$$\rho(x) = \frac{1}{k} \sum_{i=0}^{k} \text{dist}(x, \mu_i), \qquad (1)$$

where μ_i is the ith nearest neighbor of x according to the distance metric dist. Note that distance is measured in the phenotypic, i.e. behavioral, space, not in the genotypic space. The motivation comes from biology: behavioral niches that are novel will survive, regardless of what their genetic coding is.

Novelty search can be surprisingly effective in solving problems, especially those that are deceptive. For instance, evolving a robot to run through a simulated maze, using distance to the goal as the fitness, is easy if the maze is relatively simple. However, dead ends close to the goal make it deceptive, and objective-based search often gets stuck. In contrast, novelty search will create individuals that explore all the different parts of the maze, and eventually will find a way to the goal even though such solutions require traveling away from it occasionally (Lehman and Stanley, 2010a).

Novelty search also provides an interesting abstraction of biological evolution. There is no specific goal in biological evolution; instead individuals and species survive if they find a niche that they can exploit. Life therefore rapidly spreads through the available niches, leading to species that are highly adapted to their environment, and to tremendous diversity overall. Extinction events can serve to accelerate this process by selecting for highly evolvable individuals and species (Lehman and Miikkulainen, 2015).

One aspect of biological novelty search that is not captured by current methods is that such search takes place simultaneously and in parallel across the space of solutions. Individuals and species do not necessarily compete with everyone in that space, but with those that are local to them. In other words, biological novelty search is distributed. It may result in discovering similar individuals multiple times, but it may also result in discovering more diverse individuals. This distribution is the first design principle of DANS in this paper.

Combining Novelty and Fitness

The second design principle of DANS is incorporating fitness as a component into divergent search. Even though biology may not have a goal, engineering problem solving does. At the very least there needs to be a mechanism for detecting viable solutions produced by the divergent search, but there may be a benefit in guiding it as well. Diversity and novelty is necessary for discovering the stepping stones, but since we know what we ultimately want to achieve, it may be possible to guide the search towards promising areas, without diluting its power.

Several approaches for combining fitness and novelty have been proposed, and shown to be effective in solving practical problems (Gomes et al., 2015). Many of them combine a fitness objective with a novelty objective in some way, for instance as a weighted sum (Cuccu and Gomez, 2011), or as different objectives in a multi-objective search (Mouret and Doncieux, 2012). Another approach is to keep the two kinds of search separate, and make them interact through time. For instance, it is possible to first create a diverse pool of solutions using novelty search, presumably overcoming deception that way, and then find solutions through fitness-based search (Krcah and Toropila, 2010). A third approach is to run fitness-based search with a large number of objec-

tive functions that span the space of solutions, and use novelty search to encourage search to utilize all those functions (Cully et al., 2015; Mouret and Clune, 2015; Pugh et al., 2015). A fourth category of approaches is to run novelty search as the primary mechanism, and use fitness to select among the solutions. For instance, it is possible to add local competition through fitness to novelty search (Lehman and Stanley, 2011). Another version is to accept novel solutions only if they satisfy minimal performance criteria (Lehman and Stanley, 2010b; Gomes et al., 2013).

This paper advances the techniques for combining novelty and fitness in this fourth category, in two ways. First, novelty search is run in each of the parallel and distributed clients, but periodically their most novel solutions are harvested at the system level for those that are the most fit—they are then injected into the populations of the parallel searches to bias them towards high fitness. This approach is an extension that utilizes the distributed nature of DANS. Second, in selecting an individual to keep from the least novel pair, each client search prefers the fitter choice, creating a fitness bias within each search. The results show that the resulting fitness-biased novelty search is more powerful than the standard version.

Distributed Evolution Through EC-Star and Age Layering

Age-layered fitness calculation is an approach suitable for data problems in which evolved solutions need to be applied to many fitness samples in order to measure a candidates fitness confidently (Hodjat and Shahrzad, 2013). Age layering is an elitist approach: best candidates of each generation are retained to be run on more fitness cases to improve confidence in the candidates fitness. The number of fitness evaluations (i.e. samples shown) in this method depends on the relative fitness of a candidate solution compared to others at the current state of the search.

Note that this age-layering technique is distinctly different from similarly named Age-Layered Population Structure (ALPS) method (Hornby, 2006). ALPS partitions populations into layers according to generations, with the main goal of maintaining diversity. Age layering in this paper is more closely related to the Early Stopping method in evolutionary robotics, where a complex evaluation is terminated if it is guaranteed not to produce offspring even if evaluated fully (Nolfi and Floreano, 2000; Bongard and Hornby, 2010; Bongard, 2011).

EC-Star (O'Reilly et al., 2013) is a massively distributed evolutionary platform that uses age-varying fitness as the basis for distribution, and thus makes it possible to distribute large data problems through sampling, hashing, and feature reduction techniques. The available data is divided into smaller chunks, each contributing to the overall evaluation of the candidates.

In EC-Star, age is defined as the number of fitness samples upon which a candidate has been evaluated. EC-Star uses a hub-and-spoke architecture for distribution, where the main evolutionary process is moved to the processing clients (Figure 1). Each client, or Evolution Engine, has its own pool and independently runs through the evolutionary cycle. At each new generation, an Evolution Engine submits its fittest candidates to the server, or Evolution Coordinator, for consideration. The submission takes place typically after each candidate has been evaluated on fixed number of samples, called the maturity age.

The Evolution Coordinator maintains a list of the best candidates so far. EC-Star achieves scale through making copies of genes at the server, sending them to Evolution Engines for aging, and merging the aged results reported back by the Evolution Engines. This process also allows the spreading of the fitter genetic material. EC-Star is massively distributable by running each Evolution Engine on a processing node (e.g. CPU) possibly with limited bandwidth and occasional availability (Hodjat et al., 2014). Typical runs utilize hundreds of thousands of processing units spanning across thousands of geographically dispersed sites.

In the Evolution Coordinator, only candidates of the same age-range are compared with one another (i.e. they are age-layered). Each age-range has a fixed quota, and a "shadow" of a candidate that has aged out of an age-layer is retained as a placeholder for filtering incoming candidates. In this manner, unreliable estimates do not dominate the evaluation process.

EC-Star with Age Layering is well suited for implementing DANS, as will be described next.

Distributed Age-Layered Novelty Search

In the DANS approach, the Evolution Engines (clients) are configured to use novelty search for their parent selection, while the Evolution Coordinator (servers) continues to make use of fitness to decide which individuals to allow in the server pool. Evolution Engines still receive individuals from Evolution Coordinators for aging, and these individuals are added to the local elitist pool, participating as parents in creating the next generation. Each Evolution Engine, however, solely operates on the basis of novelty rather than fitness, selecting the most novel individuals in the local pool as parents. The algorithm running in each Evolution Engine is summarized in Table 1.

For each sample in the data set, a hash representation of each individuals' behavior is logged. Each individual receives a sample from the data set as its input and generates an output that defines the action to be executed. For example, in the 11-multiplexer problem, where there are eight actions, each referring to one of the data bits in the multiplexer, the behavior logged is the address of the data bit that the individual outputs for the given input.

After maturity age, individuals judged to be the most novel are selected for the elitist pool. Instead of a global
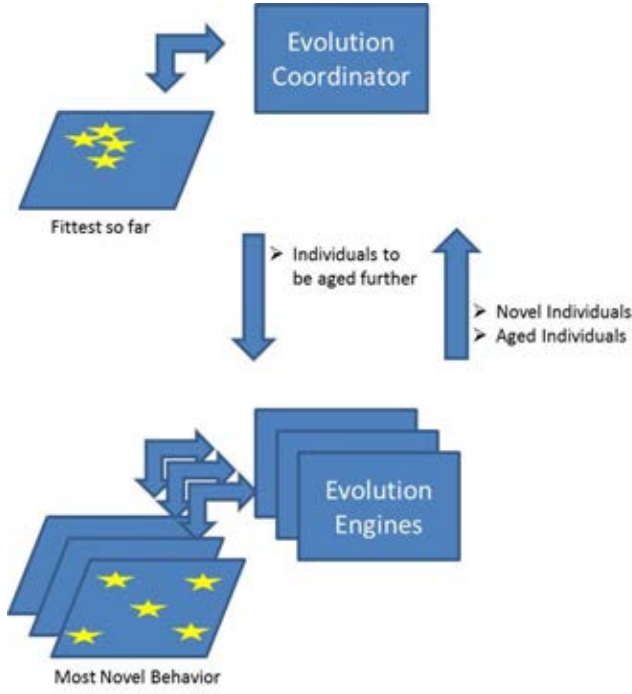
Figure 1: Implementation of Distributed Age-Layered Novelty Search (DANS) on the EC-Star System. DANS consists of a number of Evolution Engines running novelty search and a single Evolution Coordinator in a hub-and-spoke arrangement. Each candidate in each Evolution Engine is evaluated with a fixed number of samples (called the maturity age); each Engine then sends their most novel individuals to the Evolution Coordinator. The Coordinator maintains a list of these highly novel individuals ordered by fitness, and sends the most fit ones back to other Evolution Engines for further evaluation (i.e. aging) and evolution. In this manner, multiple novelty searches execute in parallel, exploring the space with more diversity, benefiting from each other's discoveries, biased towards areas with higher fitness.

archive, novelty is measured in the current population so that every individual's behavior log is based on the same set of examples. Using this log as Cartesian coordinates, Euclidean distances between individuals are calculated, and one of the individuals in the pair of individuals nearest one another is eliminated. This process is repeated until the quota for parents in the elitist pool is met. The resulting set of parents is then used to create the next generation.

Two different versions are implemented in forming the elitist pool. The first one is based purely on novelty: in the pair of most similar individuals, the one that's nearer to at least one of the other individuals in the pool is removed. The second one is based partly on fitness: in that pair, the one with the lower fitness is removed, thus subtly biasing the system to favor genes with better fitness. These two versions, called Pure Novelty and Hybrid, will be compared in the experiments that follow.

1. Receive a batch of individuals from the Evolution Coordinator.
2. If pool has capacity, fill it with randomly generated individuals.
3. Test each individual in the pool on a maturity-age number of fitness samples (each individual is run on the same sample as the others), and construct the representation of its behavior.
4. The individuals received from Coordinator have now been evaluated with more samples than before: report the results back to the Coordinator.
5. Calculate the minimum of pair-wise distances of all individuals in the pool and discard one of that pair based on distance from all other genes (the pure-novelty version), or fitness (the hybrid novelty/fitness version). Do this until only the elitist percentage of individuals remain.
6. Report the most novel individuals (i.e. the elitists from prior step) to the Coordinator.
7. Refill the pool by applying crossover and mutation operators on the elitist genes from Step 5.
8. Go to 1.

Table 1: The Evolution Engine Algorithm, i.e. the sequence of steps in advancing evolution for one generation.

Experiments

DANS was tested on two practical design optimization problems: the 11-Multiplexer and the eight-input sorting network. Each problem is described first, with its experimental setup, and then results.

The 11-Multiplexer Domain

Multiplexer functions have long been used to evaluate machine learning methods because they are difficult to learn but easy to check. In general, the input to the multiplexer function consists of u address bits A_v and 2^u data bits D_v, i.e. it is a string of length $u + 2^u$ of the form $A_{u-1}...A_1A_0D_{2^u-1}...D_1D_0$. The value of the multiplexer function is the value (0 or 1) of the particular data bit that is singled out by the u address bits. For example, for the 11-Multiplexer, where $u = 3$, if the three address bits $A_2A_1A_0$ are 110, then the multiplexer singles out data bit number 6 (i.e. D_6) to be its output (Figure 2).

A Boolean function with $u + 2^u$ arguments has 2^{u+2^u} rows in its truth table. Thus, the sample space for the Boolean multiplexer is of size 2^{u+2^u}. When $u = 3$, the search space is of size $2^{2^{11}} = 2^{2048} \approx 10^{616}$. However, since evolution can also generate redundant expressions that are all logically equal, the real size of the search space can be much larger, depending on the representation.

Following prior work on the 11-Multiplexer problem (Shahrzad and Hodjat, 2015), a rule-based representation was used where each candidate specifies a set of rules of
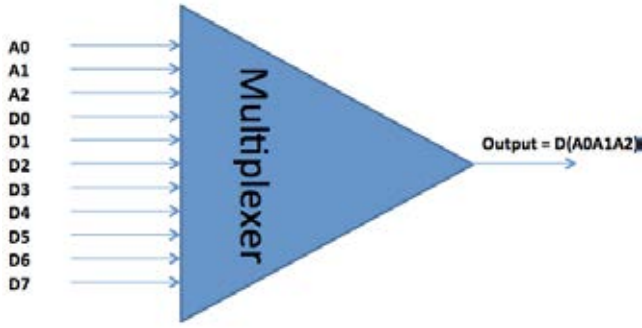
Figure 2: The 11-Multiplexer. The three address bits (top) specify one of the eight data bits (bottom) whose value will then be output. Multiplexers are a good test domain for design optimization because the search space is very large, but it is easy to check whether a design is valid.

the type

$$< \text{rule} > ::= < \text{conditions} > \rightarrow < \text{action} > .$$

The conditions specify values on the bit string and the action identifies the index of the bit whose value is then output. For instance, the following rule outputs the value of data bit 6 when the first three bits are 110:

$$< A_0 = 0 \ \& \ A_1 = 1 \ \& \ !A_2 = 0 > \rightarrow D_6.$$

These rules are evolved through the usual genetic operators in genetic programming (Berlanga et al., 2010).

In the 11-multiplexer experiments, each evolution engine has a pool size of 4000, an elitist percentage of 5%, and a maturity age of 128. That is, in each generation, each of the 4000 candidates is evaluated once with 128 randomly chosen multiplexer input samples, for a total of 512,000 evaluations per generation. At the top age-layer each candidate has thus seen 2048 samples. Crossover combines subsets of rules of each parent; mutations modify components of each rule. Fitness is defined as the number of samples an individual processes correctly, outputting the value of the data bit specified by the address bits in the 11-bit input sample. The novelty measure is the data bit address outputted by the individual for each sample. Each evolution is run until a valid multiplexer is found, i.e. one that outputs the correct bit for every possible combination of address bits.

Experiments were run comparing non-distributed runs to distributed runs with eight evolution engines per run. Two versions of distributed runs were compared: those with pure-novelty elitism, and those with hybrid novelty/fitness elitism. The non-distributed runs were implemented as hybrid runs on a single evolution engine, using the same parameters as the distributed version. Each experiments was repeated ten times, and the results averaged.
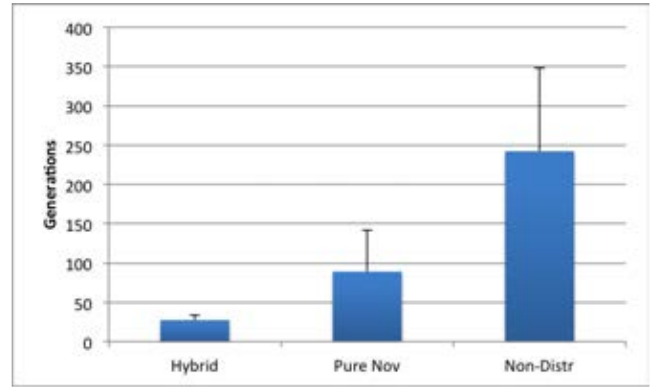
Figure 3: Performance of Hybrid and Pure-Novelty DANS vs. Non-Distributed Novelty Search on the 11-Multiplexer Problem. The plot shows the average and standard deviation of number of generations to find a valid solution. DANS significantly outperforms non-distributed novelty search, and the hybrid version of DANS the pure novelty version. The speedup is approximately linear in Evolution Engines, suggesting that DANS is an effective way to parallelize novelty search.

11-Multiplexer Results

The results are summarized in the bar graph shown in Figure 3. The main conclusion is that DANS significantly outperforms the non-distributed runs; within DANS, the hybrid elitism outperforms pure novelty. The hybrid version found a valid solution in 27.5 generations on average, pure novelty in 89.1 generations, and non-distributed evolution in 242.2 generations. Thus, distribution across the eight Evolution Engines makes evolution more reliable and speeds it up significantly, i.e. approximately linearly in the number of Evolution Engines.

The Sorting Network Domain

The second experimental domain is minimization of eight-input sorting networks. A sorting network of n inputs is a fixed layout of comparison-exchange operations (comparators) that sorts all inputs of size n (Figure 4; Knuth 1998). Since the same layout can sort any input, it represents an oblivious or data-independent sorting algorithm, that is, the layout of comparisons does not depend on the input data. The resulting fixed communication pattern makes sorting networks desirable in parallel implementations of sorting, such as those in graphics processing units, multi-processor computers, and switching networks (Kipfer et al., 2004; Baddar, 2009; Valsalam and Miikkulainen, 2013).

Beyond validity, the main goal in designing sorting networks is to minimize the number of layers, because it determines how many steps are required in a parallel implementation. A tertiary goal is to minimize the total number of comparators in the networks. Designing such minimal sort-
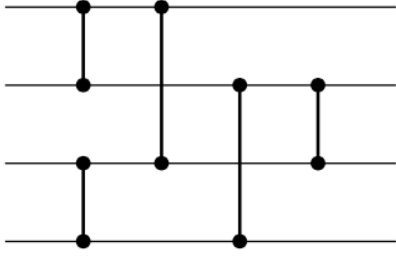
Figure 4: A Four-Input Sorting Network. This network takes as its input (left) four numbers, and produces output (right) where those number are sorted (small to large top to bottom). Each comparator (connection between the lines) swaps the numbers on its two lines if they are not in order, otherwise it does nothing. This network has four layers and five comparators, and is the minimal four-input sorting network. Minimal networks are generally not known for input sizes larger than eight, and designing them is a challenging optimization problem.

ing networks is a challenging optimization problem that has been the subject of active research since the 1950s (Knuth, 1998). Although the space of possible networks is infinite, it is relatively easy to test whether a particular network is correct: If it sorts all combinations of zeros and ones correctly, it will sort all inputs correctly (Knuth, 1998). Sorting networks are therefore a good domain to test the power of evolutionary algorithms; indeed many of the recent advances in sorting network design are due to evolutionary methods (Valsalam and Miikkulainen, 2013). The eight-input case is a good test case because the optimal network is known: it has six layers and 19 comparators (Knuth, 1998).

The sorting network representation for DANS is built on the rule-set representation of the 11-multiplexer. Each rule represents a layer of comparators; each condition within each rule identifies the input lines of the comparator; the action is not used. In this manner, it is possible to evolve sorting networks using the same methodology as for evolving rule sets. As a matter of fact, all Evolution Engine settings are the same as for the 11-Multiplexer experiments. In particular, the maturity age is 128 samples and the individuals in the top age layer have been tested with 2048 random samples.

The fitness of the network is primarily based on its ability to sort correctly, secondarily by the number of layers, and tertiarily by the number of comparators:

$$F = aS - (2^n L + C), \qquad (2)$$

where a is a proportionality constant ($10000 * 2^{16}$ in these experiments), S is the number of samples the network sorts correctly, n is the number of lines (8 in these experiments), L is the number of layers, and C is the number of comparators in the network. Because all three of these goals need to be optimized simultaneously, sorting networks represent

a more challenging and open-ended, as well as more deceptive, domain than the 11-multiplexer.

In order to measure the novelty in sorting behavior, note that there is no action to rely on, but instead the behavior needs to be constructed from the structure of the network itself. To this end, each input line is represented with a successive prime number, i.e. 1, 2, 3, 5, 7, 11, 13, and 17. The sorting network is run on the sample input, and for each pair of lines that it exchanges, the corresponding prime numbers are multiplied. The product of these values constitutes a hash for the phenotypical behavior on that sample. For example, if the sorting network has the structure

Layer 1: sort(line0, line3) and sort(line1, line2)
Layer 2: sort(line0, line4),

and the sample is 11010100 (with lines ordered 7..0), the network will rearrange it to 11000011. The phenotypical hash is then

(2 * 3) * (1 * 7) = 42.

A vector of these hash values for a number of samples (i.e. the maturity age) represents the behavior of the network, and the Euclidean distance between these vectors is used to measure novelty.

The sorting network experiments were all run until 1000 generations (which takes about two hours of total CPU time on an Intel i7 2.60GHz machine). The two versions of DANS and the non-distributed version were then compared in three dimensions (1) how fast they found a valid sorting network, (2) how many layers and (3) how many comparators did the best network found have. The results were averaged over ten runs.

Sorting Network Results

The DANS approach found valid sorting networks in 14.4 (hybrid) and 25 (pure) generations on average, compared to the non-distributed approach which took 98.1 generations on average (Figure 5. Similarly, DANS found solutions with significantly fewer layers than the non-distributed version (32.1), with the hybrid version significantly fewer than the pure novelty version (7.4 vs. 15.7; Figure 6). It was also most economical in the number of comparators: Whereas the non-distributed version used 56.2 comparators on average, the pure novelty version used 32.2 and the hybrid version only 21.7 (Figure 7).

Interestingly, two of the ten hybrid runs actually found optimal sorting networks, with six layers and 19 comparators, within the 1000 generations. These results suggest that the hybrid version of DANS could be used to discover new minimal networks, given sufficient computing effort.

Discussion and Future Work

The DANS approach can be seen as a highly robust artificial life system in which islands of evolution are searching for
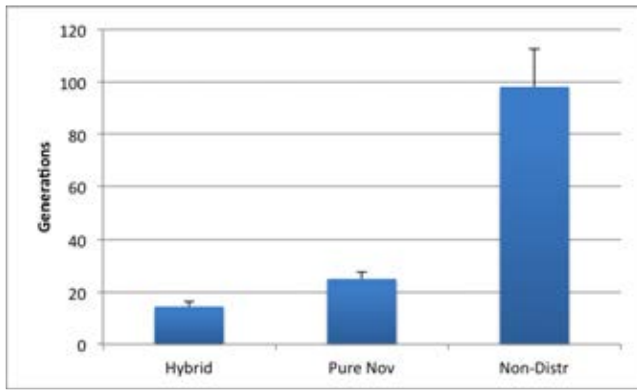
Figure 5: Number of Generations Hybrid and Pure-Novelty DANS and Non-Distributed Novelty Search Need to Discover a Valid Eight-Input Sorting Network. DANS significantly outperforms the non-distributed version, and hybrid version of DANS the pure novelty version.

behavioral niches to fill in the search space. Occasionally their most novel solutions migrate to a coordinator that aims to solve a particular problem, and therefore injects guidance into the islands in terms of the most fit of those novel individuals.

As a practical method for problem solving, DANS finds better solutions significantly faster than a similar non-distributed search: In the test problems in this paper, the speedup is approximately linear in the number of Evolution Engines. This result is remarkable because the search problem cannot be simply divided into subproblems that could be solved independently in parallel. Instead, the result is likely due to the larger total population of individuals and the increased diversity across multiple islands. The distribution makes it possible to combine fitness with novelty search effectively, by incorporating it into the migration between the islands, as well as in the selection of elitist individuals. DANS thus makes it possible to apply novelty search effectively to practical design problems.

Computationally, the system is scalable, and the coordinators can be federated (Hodjat et al., 2014). The system is also robust because it can tolerate temporarily losing its ability to coordinate (e.g, due to communication problems, or server outages, etc.), and it can even reconstruct its list of candidate solutions, should the data be lost at the coordinator. The system can also tolerate the loss of evolution engines. The approach can therefore be used to tackle big data problems that require massive amounts of computing to solve, such as minimizing large sorting networks or VLSI design in general, optimizing large-scale logistics and scheduling problems, protein folding and other biomedical optimization problems, and in general problems where each processing node can only have access to a subset of the data through sampling.
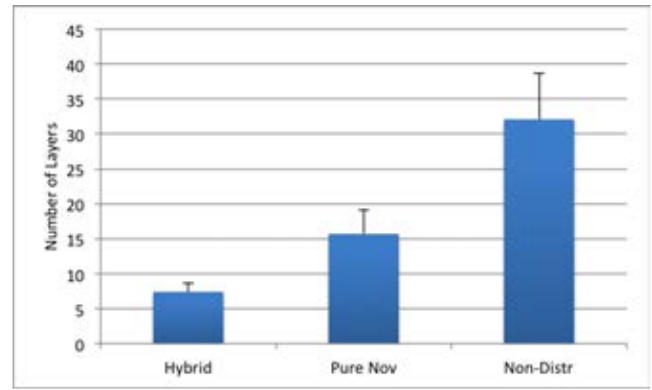
Figure 6: Number of Layers Discovered by Hybrid and Pure-Novelty DANS vs. Non-Distributed Novelty Search in 1000 Generations on the Eight-Input Sorting Network Problem. After validity, minimizing the number of layers is the main design goal; DANS significantly outperforms the non-distributed version, and hybrid version of DANS the pure novelty version.

DANS can also be useful in dynamic problems where the fundamental attributes of the problem can change through time: the evolution engines operate based on the behavior of the solutions, striving to achieve maximal coverage of the search space, versus concentrating on subspaces defined by the peculiarities of the fitness landscape. DANS is thus a mechanism that converts a powerful principle in artificial life into a practical tool for solving challenging engineering design problems.

Conclusion

This paper presents DANS, a parallel distributed design for the novelty search algorithm, and a principled way of combining fitness with novelty. These extensions result in a system that can discover better solutions much faster than standard novelty search. It thereby shows how fundamental ideas in artificial life can be useful in problem solving in the real world. DANS should be most useful in finding optimal solutions to big-data problems such as those in engineering design.

References

Baddar, S. W. A. (2009). *Finding Better Sorting Networks*. PhD thesis, Kent State University.

Berlanga, F. J., Rivera, A., del Jesús, M. J., and Herrera, F. (2010). Gp-coach: Genetic programming-based learning of compact and accurate fuzzy rule-based classification systems for high-dimensional problems. *Information Sciences*, 180(8):1183–1200.

Bongard, J. C. (2011). Innocent until proven guilty: Reducing robot shaping from polynomial to linear time. *IEEE Transactions on Evolutionary Computation*, 15:571–585.

Bongard, J. C. and Hornby, G. S. (2010). Guarding against premature convergence while accelerating evolutionary search. In
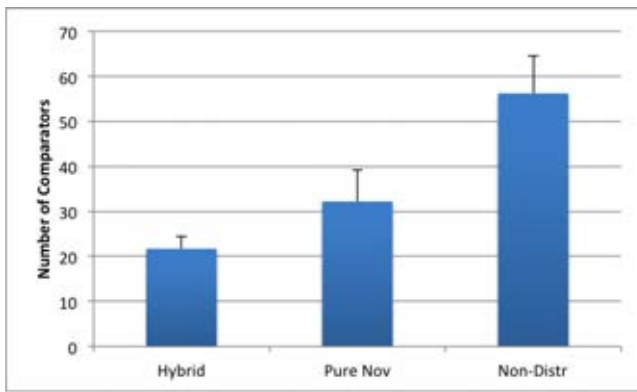
Figure 7: Number of Comparators Discovered by Hybrid and Pure-Novelty DANS vs. Non-Distributed Novelty Search in 1000 Generations on the Eight-Input Sorting Network Problem. Number of comparators is the third, and least important, component of fitness; again DANS significantly outperforms the non-distributed version, and hybrid version of DANS the pure novelty version.

Proceedings of the Genetic and Evolutionary Computation Conference.

Cuccu, G. and Gomez, F. (2011). When novelty is not enough. In *Proceedings of the 2011 International Conference on Applications of Evolutionary Computation - Volume Part I*, pages 234–243, Berlin, Heidelberg. Springer-Verlag.

Cully, A., Clune, J., Tarapore, D., and Mouret, J. B. (2015). Robots that can adapt like animals. *Nature*, 521:503–507.

Gomes, J., Mariano, P., and Christensen, A. L. (2015). Devising effective novelty search algorithms: A comprehensive empirical study. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 943–950, New York, NY, USA. ACM.

Gomes, J., Urbano, P., and Christensen, A. L. (2013). Evolution of swarm robotics systems with novelty search. *Swarm Intelligence*, 7:115–144.

Hodjat, B., Hemberg, E., Shahrzad, H., and OReilly, U.-M. (2014). Maintenance of a long running distributed genetic programming system for solving problems requiring big data. In *Genetic Programming Theory and Practice XI*, pages 65–83. Springer, New York.

Hodjat, B. and Shahrzad, H. (2013). Introducing an age-varying fitness estimation function. In Riolo, R., Vladislavleva, E., Ritchie, M. D., and Moore, J. H., editors, *Genetic Programming Theory and Practice X*, pages 59–71. Springer, New York.

Hornby, G. S. (2006). ALPS: The age-layered population structure for reducing the problem of premature convergence. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 815–822.

Kipfer, P., Segal, M., and Westermann, R. (2004). Uberflow: A gpu-based particle engine. In *HWWS '04: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 115–122, New York, NY, USA. ACM.

Knuth, D. E. (1998). *Art of Computer Programming: Sorting and Searching*, volume 3. Addison-Wesley Professional, 2 edition.

Krcah, P. and Toropila, D. (2010). Combination of novelty search and fitness-based search applied to robot body-brain co-evolution. In *Proceedings of the 13th Czech-Japan Seminar on Data Analysis and Decision Making in Service Science*.

Lehman, J. and Miikkulainen, R. (2014). Overcoming deception in evolution of cognitive behaviors. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2014)*, Vancouver, BC, Canada.

Lehman, J. and Miikkulainen, R. (2015). Extinction events can accelerate evolution. *PLoS ONE*, 10(8):e0132886.

Lehman, J. and Stanley, K. O. (2010a). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 2011:189–223.

Lehman, J. and Stanley, K. O. (2010b). Revising the evolutionary computation abstraction: Minimal criteria novelty search. In *Proceedings of the Genetic and Evolutionary Computation Conference*.

Lehman, J. and Stanley, K. O. (2011). Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th Annual Genetic and Evolutionary Computation Conference (GECCO 2011)*, Dublin, Ireland.

Mouret, J. B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv*, 1504.04909v1.

Mouret, J.-B. and Doncieux, S. (2012). Encouraging behavioral diversity in evolutionary robotics: An empirical study. *Evolutionary Computation*, 20:91–133.

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics*. MIT Press, Cambridge.

O'Reilly, U.-M., Wagy, M., and Hodjat, B. (2013). EC-Star: A massive-scale, hub and spoke, distributed genetic programming system. In Riolo, R., Vladislavleva, E., Ritchie, M. D., and Moore, J. H., editors, *Genetic Programming Theory and Practice X*, pages 73–85. Springer, New York.

Pugh, J. K., Soros, L. B., Szerlip, P. A., and Stanley, K. O. (2015). Confronting the challenge of quality diversity. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, New York, NY, USA. ACM.

Salge, C., Glackin, C., and Polani, D. (2013). Empowerment - an introduction. *CoRR*, abs/1310.1863.

Secretan, J., Beato, N., D'Ambrosio, D. B., Rodriguez, A., Campbell, A., Folsom-Kovarik, J. T., and Stanley, K. O. (2011). Picbreeder: A case study in collaborative evolutionary exploration of design space. *Evolutionary Computation*, 19:345–371.

Shahrzad, H. and Hodjat, B. (2015). Tackling the Boolean multiplexer function using a highly distributed genetic programming system. In Riolo, R., Worzel, W. P., and Kotanchek, M., editors, *Genetic Programming Theory and Practice XII*, pages 167–179. Springer, New York.

Shahrzad, H., Hodjat, B., and Miikkulainen, R. (2016). Estimating the advantage of age-layering in evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2016)*, Denver, CO.

Stanley, K. O. and Lehman, J. (2015). *Why Greatness Cannot Be Planned: The Myth of the Objective*. Springer, Berlin.

Valsalam, V. K. and Miikkulainen, R. (2013). Using symmetry and evolutionary search to minimize sorting networks. *Journal of Machine Learning Research*, 14(Feb):303–331.

Whitley, D., Rana, S., and Heckendorn, R. B. (1999). The island model genetic algorithm: On separability, population size and convergence. *Journal of Computing and Information Technology*, 7:33–48.

Wissner-Gross, A. D. and Freer, C. E. (2013). Causal entropic forces. *Physical Review Letters*, 110:168702.

Robotics

Spiral autowaves as minimal, distributed gait controllers for soft-bodied animats

Michał Joachimczak, Rishemjit Kaur, Reiji Suzuki, Takaya Arita

Graduate School of Information Science, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan
mjoach@alife.cs.is.nagoya-u.ac.jp

Abstract

Inspired by the self-organization of growing embryos and co-ordinated movement of multicellular assemblies such as the slime mold *Dictyostelium*, where each cell is controlled by the same controller (a DNA-encoded gene regulatory network), we evolve distributed gait control mechanisms for soft-bodied animats. The animats are made of compressible material, with each body region capable of independent actuation, controlled by a cell at its center. Each animat consists of hundreds of cells uniformly distributed throughout the body, each sharing the same artificial gene regulatory network and aware of the state of their local neighborhood. We found that one of the most common actuation patterns that emerged relied on cells synchronizing their oscillations in order to produce a rotating, spiral wave spanning throughout the body. We found this type of mechanism to emerge for a wide range of animat morphologies as well as in very different types of initial conditions. We investigate how the evolved controllers produce the pattern through local feedbacks and evaluate spiral stability when imperfect, noisy cells are used.

Introduction

Taking inspiration from distributed control mechanisms observed in nature, such as self-organization of a growing multicellular embryos and movement of multicellular assemblies of certain amoeba known as slime molds (e.g., *Dictyostelium*), we investigated the possibility of evolving distributed controllers for prespecified morphologies of soft-bodied robots that would produce gaits in a truly decentralized manner. By dividing animat bodies into hundreds of cells capable of communicating with their neighbors, we were expecting to observe the evolution of some form of autowaves organizing the gaits. Autowaves are a special type of nonlinear waves that are known to occur in active media and the main difference between autowaves and classical waves is that propagation of the former occurs at the expense of energy stored in the medium. The energy is used to trigger process into adjacent regions (Roska et al., 1995; Manganaro et al., 1999). Autowaves occur in many biological phenomena, in particular they are essential to multicellular development, but are also central to processes such as propagation in nerve fibers or heart excitation.

The unexpected result of our evolutionary experiments was the predominant type of control mechanism that emerged. It was based on producing a very specific type of autowave: a rotating spiral known as a spiral autowave. Spiral autowaves are frequently observed in excitable media
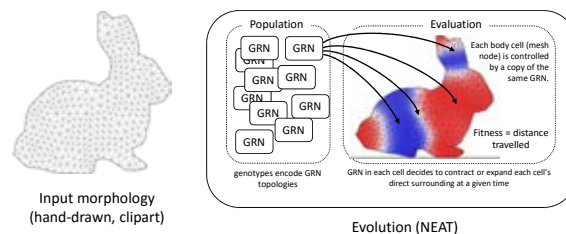
Figure 1: Overview of the evolutionary approach used to the evolve distributed controllers for simulated soft-animats.

(Ma et al., 2010), and have been observed to emerge in media as different as chemical solution of Belousov-Zabotinski reaction, cardiac tissue or neurons of neocortex, though they usually emerge in a chaotic, unpredictable form. In this work, however, we were able to observe how evolution creates controllers that self-organize into spiral autowaves anchored at a specific location of animat's body and produce cyclic, sustainable gaits.

Methods

We have employed the same approach to simulate soft-bodied animat locomotion as in our previous studies (see full description in Joachimczak et al., 2015), that is animats are two dimensional and are represented as a set of point masses (corresponding to cells) connected with springs. Unlike our earlier work, where we investigated co-evolution of bodies and brains, here we focused solely on the design of distributed controllers only. Hence, we assumed that morphology of an animat is specified at the beginning of an evolutionary run and does not change (other than the elastic changes during locomotion). Animat shapes were specified either as a drawing or a clip-art and then algorithmically triangulated to produce a mesh with a desired number of nodes. Locomotion was possible owing to the local, elastic changes to the body controlled by each cell.

During the evaluation, each cell of an animat is controlled by a copy of the same evolved artificial gene regulatory network (GRN) encoded in the genome (Fig. 1), with gene expression levels changing in a continuous manner. Despite the same controller, cells can differ in their behaviors, due to differences in environmental (input) signals ultimately producing different internal states of cells. The GRN topology was evolved using the NEAT (Stanley and Miikkulainen, 2002) algorithm, a state of the art technique for evolving network topologies. Fitness function promoted
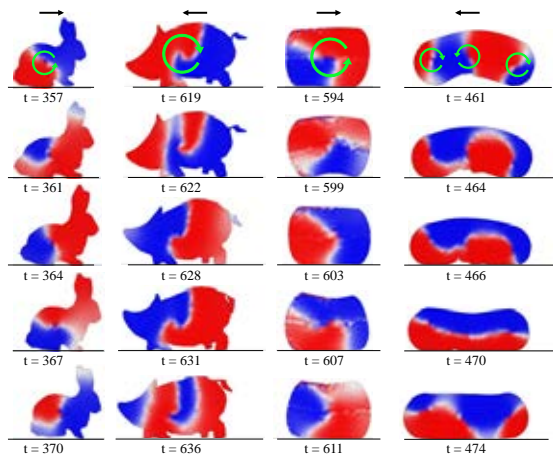
Figure 2: Examples of spiral autowave driven actuation evolved for four different morphologies. Color shows current local actuation signal (red - expansion, blue - contraction). Circular arrow indicates the direction of spiral rotation. Videos of each animat's gait available at: https://goo.gl/j10UnZ (additional links in the description of the video)

distance achieved by simulated animat.

We evolved GRNs with only a single output determining the current level of contraction or expansion of a body region around a given cell. The main input represented averaged output state of the cell's neighbors. To ensure that oscillatory activity cannot start and sustain itself without receiving signals from neighbors, we did not use any bias inputs. We then experimented with different methods of seeding the initial activity by stimulating a few cells or providing maternal gradients. We would remove the seeding signals after a short period of time, so that the activity within the body had to sustain itself through the propagating autowaves. Finally, to identify the evolved mechanisms of emerging autowave patterns, we compared experiments in which recurrent connections in GRNs are allowed or disabled.

Results

We found that evolution, tasked with a problem of evolving distributed, local communication-driven controllers for soft animats repeatedly converged on a very simple and creative solution that relies on producing a rotating spiral autowave anchored in the center of the body or even multiple synchronized spirals in case of elongated individuals (Fig. 2). We also found that this simple control mechanism evolves for a wide range of tested animat morphologies and emerges both if a highly localized (two cells) or a global (gradient) seed stimuli are used to initialize the waves of cellular activity. This suggests that the spiral patterns stem from initial heterogeneity in cellular activity, though a single cell stimulus was not sufficient in our experiments. In each case, what starts as seemingly chaotic waves propagating through the body, in a few hundreds of GRN updates forms a rotating spiral that sustains itself, often indefinitely. As the rotating arm of a spiral sweeps through the bottom of animat's body, the bottommost cells are raised, producing a gait that works
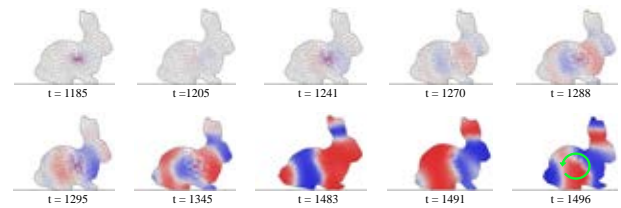
Figure 3: A rotating spiral autowave emerges from cellular activity seeded in two cells at the center of the body. Video available at: https://goo.gl/0klqno

both for morphologies that have a flat bottom as well as for morphologies supported by appendages.

Finally, we compared scenario in which cells can rely on a feedback of their own state with that of cells relying on the state of their neighbors and found that while the spiral autowaves emerged in each of the cases, different types of designs had very different robustness to noise. In particular, if we assumed that cells' internal clocks are imperfect, only the experiments in which cells communicate with neighbors were able to produce sustainable spirals.

Conclusions

While the spiral autowaves are a common phenomenon in many physical and biological systems, we see their unexpected emergence in the context of evolving distributed gait controllers for soft-animats as an example of how artificial evolution can surprise us and suggest entirely new type of design, one that would be otherwise unlikely to be proposed by a human designer. Further study will reveal how robust is this type of design and how well it can apply to actual, 3-D soft-robots.

References

Joachimczak, M., Suzuki, R., and Arita, T. (2015). From tadpole to frog: artificial metamorphosis as a method of evolving self-reconfiguring robots. In *Proc. of the 13th European Conference on the Synthesis and Simulation of Living Systems (ECAL 2015)*, pages 51–58. The MIT Press.

Ma, J., Wang, C.-N., Jin, W.-Y., and Wu, Y. (2010). Transition from spiral wave to target wave and other coherent structures in the networks of Hodgkin-Huxley neurons. *Applied Mathematics and Computation*, 217(8):3844–3852.

Manganaro, G., Arena, P., and Fortuna, L. (1999). *Cellular Neural Networks Chaos, Complexity and VLSI Processing*. Springer Berlin Heidelberg.

Roska, T., Chua, L. O., Wolf, D., Kozek, T., Tetzlaff, R., and Puffer, F. (1995). Simulating nonlinear waves and partial differential equations via CNN. I. Basic techniques. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(10):807–815.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evol. Comput.*, 10(2):99–127.

Self-organized control of an tendon driven arm by differential extrinsic plasticity

Georg Martius[1], Rafael Hostettler[2], Alois Knoll[2], and Ralf Der[3]

[1]IST Austria, Am Campus 1, 3400 Klosterneuburg, Austria
[2]Institut für Informatik VI, TU München, Boltzmannstr. 3, 85748 Garching bei München, Germany
[3]Max Planck Institute for Mathematics in the Science, Inselstr. 22, 04103 Leipzig, Germany
gmartius@ist.ac.at

The self-determined cognitive development of high-complexity autonomous robots is a challenging task for both the creation of robot-human ecosystems and the creation of artificial life systems with real, human-like robots. Anthropomimetic robots are a prominent example of this challenge. Different from classical robots, anthropomimetic robots are built following the morphology of the human body. Such robots are more soft than classical systems making them safer to interact with and thus favorable for service robots in human environments. Moreover, because of their human like morphology, they can be used for better understanding human behavior generation and development.

World wide, several of such muscle-tendon driven (MTD) systems have already been built. While mechatronically at an advanced level, the control of both MTD and soft robotic systems in general is still in its infancy. A generic example is given by pertinent EU projects ranging from CRONOS, to ECCEROBOT to MYOROBOTICS. While excellent work has been done in building these robots their control faces many problems. Learning of control policies becomes essential and is investigated mainly in the reinforcement learning setting.

Without a very compact parametrization learning a new behavior takes a very long time in high-dimensional systems. This situation clearly calls for new controller paradigms which optimally exploit the physical properties of such soft systems as indicated by embodied AI. This paper presents an approach that includes the world—i. e. body plus environment—more actively and more systematically in the control process than other embodied control approaches. By inverting the roles of the controller and the controlled, the world becomes not only "its own best model" (Rodney Brook's idea) but is leveraged to "its own best controller" (Der and Martius, 2016). This idea can be implemented by a neural network with a novel synaptic plasticity rule (Der and Martius, 2015, 2016), as shown in Fig. 1.

The novelty of the controller can be demonstrated best by applying it to MTD systems, for instance the Myo-robotics arm, reported here, with its ball and socket shoulder joint and 9 muscles in total. Different from classical robots with
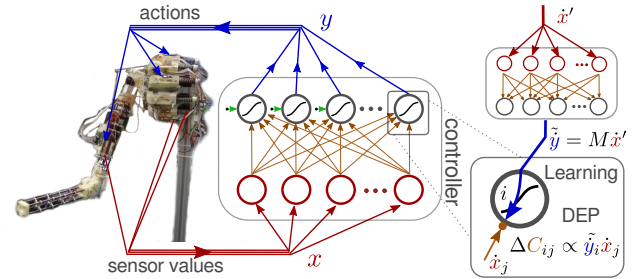
Figure 1: **Neural controller network connected to the Myo-robotic arm.** The inset on the right illustrates the synaptic plasticity, called differential extrinsic plasticity, which is driven by a modified differential Hebbian law, multiplying the time derivatives of the incoming sensor values $\dot{x}$ with the virtual motor values $\tilde{y}$, which are generated by the inverse model (M, one-to-one mapping in the case of the arm) from next input's derivative $\dot{x}'$.

revolute joints, the motor positions do not directly translate into joint angles and into poses. Due to the elasticity of the muscles, there are infinitely many combinations of motor positions for a single arm pose. Apart from that, the most challenging task is to avoid the dislocation of the shoulder which cannot happen with revolute joints. Astonishingly, although structurally extremely simple, the new control paradigm does not have problems with these particularities. For instance the tendons are kept tight automatically, such that no dislocation appears. When embedding our controller, see Fig. 1, into the sensorimotor loop, a meta-system—consisting of the mechanical system, the controller with its sensor driven synaptic dynamics, and the energy supply (battery)—is created displaying a rich behavioral spectrum like limit cycle attractors, long lived transients, and fixed point flows generating pseudo-random sequences of poses. The concrete behavior is not given explicitly, but specific behaviors develop by themselves in a dynamical interplay between controller dynamics and world dynamics. This open physical system is like a reservoir of meta-stable behavior patterns waiting to be excited. Ex-

citation can be achieved either by manual interaction (see below) or by the self-amplification of latent modes spontaneously arising in physical subsystems, see Video 1 (see `playfulmachines.com/MyoArm-2`). By way of example, consider Video 3 where a weight (a bottle) was suspended from the tip of the arm with a string, forming as a physical subsystem. In the beginning, minimal motor activities are seen to spontaneously excite minor pendulum motions. These oscillations directly exert physical forces on the arm which propagate via the springs into the sensor values and eventually into the synaptic dynamics which governs the behavior. This may lead to the amplification of latent pendulum modes until a stable circular movement of the pendulum is achieved. These findings elucidate how a physical subsystem (the pendulum) may pilot—by its internal dynamics—the meta-system into a resonant state, i.e. a whole-system mode with defined frequency.

Actually, this is the essence of the method which explains the emergence of specific modes—specific for the actual physical setting—of the system. For instance, when attaching a bottle half-filled with water to the tip of the arm in either horizontal or vertical orientation, stable shaking modes are arising, as demonstrated by both Video 6 and Video 7 showing modes specific for each physical setting (horizontal or vertical bottle). Again, we see how the meta-system may become resonant with the internal dynamics of a subsystem, if the latter provides perceivable correlations over space and time. This is the case for instance when the water is hitting either the walls or top and bottom of the bottle. These impacts cause a reaction of the springs and hence of the sensor values, which may increase correlations in the synaptic dynamics resulting in enhances motions of the arm in coherence with these signals.

By this compliance mechanism, the "brain" may also discover (dynamical) affordances of the physical world it is interacting with. In further experiments, the robot is connected to a revolvable bar or wheel with weights for giving it the some moment of inertia. In Video 8 the robotic arm finds a behavior rotating the wheel from an initial push by the user. When positioning the wheel in parallel to the arm, the modes were emerging even more readily as seen by Video 9. Moreover, the system can immediately be switched between the forward and backward rotation mode. This is possible because the time-scale of the synaptic plasticity is so fast, in the order of one second, that the new dynamics is quickly propagating into the controller via the plasticity rule. By changing a time-scale meta-parameter the frequency of rotation can be adjusted, see Video 10. The spontaneous emergence of the wheel rotation behavior can be argued to be a cognitive act if we consider—in the sense of (radical) embodied cognitive science—that cognition is to be described in terms of agent-environment dynamics and not in terms of computation and representation.

In another experimental situation, the robot is equipped with a brush and forced by manual guidance to wipe a table. Video 11 demonstrates how, by the combination of the limiting table plane and the manual force, the robot is driven into a two-dimensional wiping mode. This is seen to slowly wander through different wiping modes by the dynamics of the meta-system. Again, the manual interaction with the arm by little forces is always possible as seen later in the video. This is due to the tight closed loop control and the property of the synaptic dynamics to be compliant to external perturbances, see Der and Martius (2015). Most importantly, emerging motion patterns can be identified and stored by the user simply by taking snapshots of the synaptic weights. Video 12 shows the recall of previously acquired wiping modes. The transition between different modes is achieved by hard switching of the fixed controller weights, nevertheless smooth transients are observed.

In summary, we have treated a soft, high-complexity robotic system which a novel goal free exploratory control algorithm. It reverts the role of the controller and the controlled and makes a set of non-trivial and highly coordinated behaviors emerge solely from the interaction of synaptic dynamics, neural transmission and the mechatronic system. It provides a systematic approach for behavioral self-organization avoiding the reality gap as demonstrated by our applications to both simulated and real robots. In this way it can help to lift Artificial Life ecologies to a new level of complexity approaching physical reality of, say, human-robot ecologies. The new controller may speed up evolution enormously Der and Martius (2015) as the emergence of a new trait needs only a mutation in morphology with adequate behaviors coming for free. Also the controller is fully deterministic revealing that behavioral proliferation can be the result of spontaneous symmetry breaking. Seen as a practical approach to generate complex, force-sensitive interactions with the environment this controller could also augment the repertoire of classical controllers. Additionally, it may shed light on how biological musculoskeletal systems generate the complex trajectories they use to interact with the environment with an unrivalled flexibility—not as a heavily controlled process but as an emergent phenomenon.

References

Der, R. and Martius, G. (2015). Novel plasticity rule can explain the development of sensorimotor intelligence. *PNAS*, 112(45):E6224–E6232. arXiv: abs/1505.00835.

Der, R. and Martius, G. (2016). The world as its own best controller: a case study with anthropomimetic robots. arXiv: abs/1602.02990.

Neuroevolution of Feedback Control for Object Manipulation by 3D Agents

Adam Stanton and Alastair Channon

School of Computing and Mathematics, Keele University, ST5 5BG, UK
{a.stanton, a.d.channon}@keele.ac.uk

Abstract

It has been shown that manipulation of objects by 3D virtual creatures can play an important role in the evolution of complex, embodied sensorimotor behaviours. In this work we examine the capacity of virtual creatures that use evolutionary and control architectures already shown to be capable of sensor-differential gradient-following locomotion (tropotaxis) to adapt to solve a physical problem involving the manipulation of 3D objects in their environments. Specifically, the creatures' task is to guide a physically-modelled cube through their environments in order to achieve maximum covered distance of the object. Agents were evolved in the manipulation environment from random initial genotypes and from genotypes previously optimised for performance in a different task. Performance was evaluated both before and after evolutionary adaptation. We show that the architecture achieves embodied feedback control in the block movement task. We observed some overlap between the earlier and later environments but also that success in the first environment does not preclude or entail success in the second. We found that species evolving from scratch do no better or worse than those optimised for a different environment, and that sensory feedback is necessary for correct approach and control behaviours in agents, although close control is less dependent on sensory input than distance approach.

Introduction

The evolution of virtual creatures in physically simulated three-dimensional worlds was first demonstrated in 1994 in the work of Karl Sims, who first evolved articulated agents to swim, walk, jump or follow a light source (Sims, 1994b) and then evolved such agents to compete to gain control of an object (Sims, 1994a). The diverse range of strategies and counter-strategies evolved through the latter task demonstrated both co-evolution's ability to generate increasingly complex behaviours and that object manipulation can play an important role in the evolution of sensorimotor intelligence (beyond mere locomotion and taxis) in simulation, as in nature.

The 3D River Crossing (3D RC) task, first presented in Stanton and Channon (2015), provides an ideal base from where the evolution of sensorimotor intelligence and related issues of physical embodiment can be explored. In that work we adapted the *shunting model* of Grossberg (1988) and

Yang and Meng (2000), used for the first time in an a-life context in Robinson et al. (2007), to build an evolutionary environment able to evolve control architectures of 3D virtual creatures that exhibit both reactive and deliberative behaviours. However, the problem-solving aspect of the 3D RC task in that work was abstracted from the physicality of the agent's morphology. Although each agent's joint motors were driven by some of the outputs from its neurocontroller, other neural outputs only notionally represented manipulation of physical objects in the agent's world.

An important extension of the earlier work into richer interactions is thus to introduce aspects of the deliberative problem to the agents' physical world, requiring an intricate manipulation of simulated objects to solve the challenge. In this work, we take a first step toward that goal by investigating whether the neural architecture outlined in that work can successfully constitute the control system for a simple manipulation task: displacement of a physically-modelled block in the agent's world, requiring feedback control, hereafter called the *block displacement* (BD) task.

Our general approach is to consider populations of agents in a new environment that provides the physical block challenge. The agents' neural control systems are sensitised to the location of the block by direct interaction with the shunting model, simplifying the adaptive problem. We investigate evolution on the BD task from both random (unevolved) populations and from populations of creatures previously evolved in the 3D RC environment. Hereafter we refer to random populations as *unevolved* populations and populations evolved only in the 3D RC environment as *naive* populations.

Hypotheses

The objective of this work is the evolution of agents able to successfully complete the BD task, as observed through 3D visualisation. In addition, we developed and tested the following hypotheses in order to further understand the interactions between the various components of the system and explore the limitations of the 3D RC architecture:

H1. The hybrid architecture is sufficient to achieve feedback

control that allows agents to successfully manipulate and guide an external object;

H2. There is some overlap between the earlier 3D RC task and the BD task due to the requirement for speedy and accurate movement in both environments;

H3. Species evolved in the 3D RC task show increased performance after evolution in the BD environment (i.e., it is possible to optimise this behaviour further), and that success in the 3D RC environment does not preclude success in the BD environment.

H4. Some 3D RC species are on evolutionary trajectories more suitable for the BD task than others.

The remainder of the paper presents an overview of the method used to generate the agents, and the results of the evolutionary and ablative experiments designed to test the above hypotheses. We then present conclusions and a discussion that relates the design of the base system to the observed results.

Methods

In this section, we describe how the overall objective of implementing a system capable of using an evolutionary algorithm to produce agents able to manipulate objects in a 3D, realistic physics world was achieved. The solution is split into three parts. The first part is the design of the evolutionary problem that the agent species must evolve to solve; the second part documents the abstractions made in the agent's morphology and control architecture that are under the control of the evolutionary algorithm and the third part describes the evolutionary algorithm itself. Finally we describe the data collection scheme we use to collect outputs from the experiments.

The Physical 3D RC Problem

The general problem used in this work is an adaptation of the 3D RC task described in earlier work (Stanton and Channon, 2015), following the same key ideas described and used to various ends in Robinson et al. (2007) and Borg and Channon (2011). The innovation in this work is the addition of a requirement for agents to physically manipulate objects in the environment; in our previous work only the body of the agent is physically simulated and all environmental interaction is through a two-dimensional, grid-world abstraction. In the original RC task, 2-dimensional agents are able to move between discrete cells in a 20x20 grid world containing hazards (*traps* and *water*) and resources (*stones* and *resource*. Stones can be carried by the agent and placed into water, enabling bridges to be built. Success in this environment is determined by agents' ability to avoid hazards and reach the single *resource* by learning an appropriate action policy given the current state; this includes capturing an element of deliberative planning in order to build bridges in worlds containing an otherwise impassable stretch of water.

This task was extended in our 2015 work to three dimensions (the 3D RC task), making the problem significantly harder. Agents are embodied in a four-legged fixed-morphology physical form that is simulated using a Newtonian rigid-body mechanics system, meaning that physical control (principally, the locomotive and orienting behaviours required for moving between grid cells) must be part of any solution. The agent's position in 3D is projected and quantised to the 2D RC world and any output from the control architecture translates directly into motor control in the 3D environment.

This work introduces the Physical 3D RC (P3D RC) task where the physical problem is extended beyond the agents' control of their bodies, to the wider environment. Solutions to the P3D RC task involve manipulation: in addition to the agents' bodies, a cube representing a *stone* in the world is also physically simulated. Any solution must use physical motor control to manipulate the cube into a configuration that allows the agent to access the resource objective.

As a step toward the P3D RC task, we first investigate simpler problems where agents must simply move blocks around in the world, without the requirement to solve the deliberative component of the RC challenge. This paper addresses the first of these challenges, where the problem is to move the environmental block as far as possible.

As in our earlier work and summarised here, agents have a symmetrical quadruped body plan comprising a torso (dimension $1.0 \times 1.0 \times 0.2$), four upper limbs and four lower limbs (dimensions $0.5 \times 0.2 \times 0.2$ each). Upper limbs are attached to the torso at each lower corner with a 2-axis constraint, limiting the range of motion relative to the torso. Knees connect upper libs to lower limbs, constraining their relative motion to a hinge. Four small sensors are also modelled in the physical environment as fixed appendages to the agent's torso; this is for convenience of updating sensor values based on their position and the sensors have no effect on the physical operation of the agent. The physical simulator used was Open Dynamics Engine (ODE) version 0.13.1, using a fixed timestep of 0.01s, friction pyramid approximation for contact response ($\mu = 10.0$) between agent and the ground plane, universal error reduction (ERP) of 0.2 and force-mixing (CFM) of 5×10^{-5}. In addition to the agents, a $1 \times 1 \times 1$ block is simulated at the centre of the environment ($\rho = 0.1$). On initialisation, agents are randomly positioned on a circle with radius 5 units from this point.

Agent Control

Given the above problem, a strategy to solve it necessarily requires a control architecture that receives sensor data from the environment and produces appropriate motor stimulation to guide the agent through the challenges of the world. We use a bespoke, hybrid neural network (HNN) to this end. The HNN comprises feed-forward networks for saliency calculation from sensor data, a locally-

connected, topologically-organised *shunting* neural network (Yang and Meng, 2000) for modelling the agent's world, a feed-forward bridge between this model and the motor control parts of the architecture and a series of recurrent leaky integrator networks in the style of Beer's Continuous-Time RNNs (Beer and Gallagher, 1992) that actually produce motor output from the control system. These components we label the *decision network* (DN), the *shunting model* (SM), the *physical network* (PN) and the *pattern generators* (PG). Together, these components are able to successfully solve the 3D RC task, as demonstrated in Stanton and Channon (2015).

Since the details of this hybrid architecture are elucidated in previous work, we present a only a summary of the architecture below, along with notes on aspects that have been modified for the present work. See figure 1 for a detailed exposition in graphical form.

In our 2015 work, the DN and SM follow the ideas presented in Robinson et al. (2007) closely. Together and properly configured, they provide a neural-like encoding of a fixed action policy relating current state (position, local objects and carrying state) to action (preferred movement direction, and a pick-up or put-down action). In the first part of the present work, the focus is on physical performance rather than the species' capacities to learn an appropriate state–action policy. As such, we hard-code ι-values (saliency values) for objects in the agents' worlds rather than learn appropriate weights in the DN.

The PN controls the agent's behaviour in the world. The state transition landscape produced by the agent's SM is sampled at four points physically located on the agent's body and these values are used as input to this network. Thus, information about desirable state transitions (in this model, directions to move) is available to the PN and can be used by agents to discriminate important features of the preferred state configuration relative to the agent's configuration. The agent's configuration can then be updated to climb the gradient in the state space.

Actual control of the agent's body to achieve this reconfiguration is mediated by the PG network. The network is an array of five three-neuron oscillator circuits, comprising simple leaky-integrator neurons governed by a set of coupled differential equations, modelled after those of Reil and Husbands (2002). The PG network receives input from the PN that perturbs the oscillating cycles which in turn affects the agent's behaviour in the world. The oscillator circuits are a given abstraction in the agents' design, generated by a pre-evolutionary phase that is documented in previous work and summarised in the next section.

Last, outputs from this network are used as target angles for the various joints in the agent's body; actual torques are applied according to a proportional-derivative (PD) equation based on the difference between the current and desired angles at the joint.
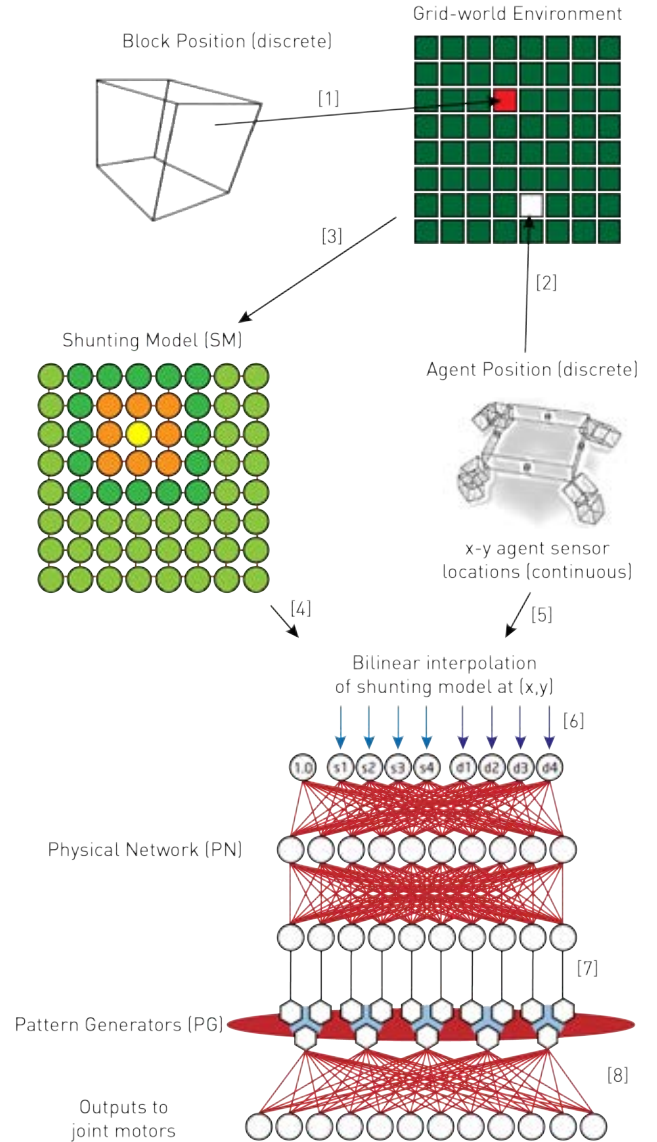
Figure 1: Neural architecture. The agent's 3D world, containing the agent and the block, is discretised into a 2D grid (1 and 2; cell-width is 1 unit in the physical model); grid locations are given ι-values where salient objects exist and this is used to generate the diffusive shunting model (3). Agents sample the landscape (4) at four different continuous positions given by their four sensors (5) by interpolation of values around the sensor location (6). These values pass through a feed-forward network and affect the dynamical trajectories of pattern generators (7) that ultimately output values to effectors via weighted links to joint motors (8). Links shown in red are subject to evolutionary optimisation, both in the pre-evolutionary phase and in the later block task. This includes the red region around the five preset pattern generators whose interneuron weights are also variable: within a single generator preset weights are adapted; across generators weights are initialised at zero but can also move from this value.

Evolutionary Algorithm

Pre-evolution As noted above, populations exploring the block task have been pre-evolved in other environments and also contain specific neural circuits that were produced in an additional, separate environment. These circuits were produced in isolation: three-neuron motifs were evaluated for their capacity to stably generate a 1Hz sinusoidal oscillation in the presence of an input signal and to be quiescent otherwise using an objective function based on the Fourier transform of their output over a 10-second window. The major pre-evolutionary phase involved the simulation of 20 species of agent in the original 3D RC environment. These species progressed through the documented incremental evolutionary phases of food collection, sprinting and hazard avoidance; the evolutionary process was halted before the deliberative part of the incremental challenge. (Specifically, agent populations were allowed 250k 3-individual tournaments; it was found that all 20 species had progressed to the deliberative component by this point.) The 20 species, all capable of tropotactic locomotion, were then installed in the block environment. All evaluation was carried out using a bespoke distributed evaluation system across approximately 200 CPU cores, achieving approximately 100 evaluations per minute.

Evolutionary Parameters In all cases the evolutionary algorithm is a three-individual tournament selection-based optimisation process, operating on a population of 150 genomes. Individuals' neuro-controllers are represented as an array of floating-point values. On reproduction, single-point crossover occurs between the two winning individuals in the tournament, and Gaussian mutation is applied to alleles of the resulting child genome with probability $1/l$, where $\mu = 0$ and $\sigma = 1$.

Objective Functions For the pre-evolution of oscillator circuits, the objective function was the number of non-1Hz frequencies in the frequency domain of a ten-second sample of the output neuron's signal in the input-high state, and the total number of frequencies in the input-low state. During the pre-evolution of gradient-ascending virtual creatures, the objective was as defined in (Stanton and Channon, 2015); agents of high fitness completed many of the incremental stages of the 3D RC task. For the evolved block-pushing task, the objective is to maximise the distance covered by the block in the discrete grid-world.

Data collection

To examine H1, we collected observations of agent behaviour, including extracting trajectory data from the highest-scoring individual from the BD task under various sensory ablation conditions. Deafferentation of control inputs was achieved by systematically disabling sensors, and the agent's progress in a controlled version of the BD task was recorded across a two minute time interval. For each ablation, we examine *approach* and *control*. In both cases the block is positioned at (20,20); for *approach* the agents start

Figure 2: Visualisation of a single agent in the block-displacement world. Agent is displaying a low, heavy gait suitable for block pushing.

far from the block at (5,5), and for *control* they start very close at (18,20). The trajectories followed by agents and block in the two scenarios illuminate the dependence of the gaits on sensory feedback. To examine H2, we used mean evolutionary performance data from the final 1000 tournaments of the 3D RC pre-evolution phase in comparison to the mean score of the same species in the BD task, evaluated for 2 minute and 10 minute periods (simulation time). To examine H3, we measure the naive BD score before evolution takes place of each individual in each population, in 10 randomly initialised trials. Each trial evaluates the individual for 10 minutes in the BD task. After the evolutionary phase, we repeat the process. We also collected evaluation data for each individual in each of 20 populations over 10 trials of 10 minutes each, after 100k tournament evolution when individuals begin with random genotypes. To examine H4, we use the evaluation fitnesses for naive and evolved species.

Results

H1: The hybrid architecture is sufficient to achieve feedback control that allows agents to successfully manipulate and guide an external object Visualisations of agent behaviour can be seen at `https://youtu.be/gZaUvXcdMK8`, and figure 2 provides a static view of an agent. The zoopraxiscopic figures (in the style of Eadweard Muybridge) show a time-series of snapshots that illustrate how agents approach the block from a distance (figure 3a), and manipulate the block in their world (figure 3b). The sensory ablation data are presented in figure 8. Figure 8a shows the planar trajectory followed by agents approaching the block from a distant point under various deafferentation conditions; figure 8b shows the response of agents to the same sensory culling in a closer, control scenario.

H2: There is some overlap between the 3D RC task and the BD task due to the requirement for speedy and accurate movement in both environments. A non-parametric correlation analysis was undertaken between the species' relative ranks for mean fitness during the final 1000 tournaments of the 250k-tournament 3D RC pre-evolutionary runs and the mean score on the BD task. Figure 4 presents this correlation graphically for both two minute and ten minute evaluation times. In the 10m trial we found a statis-
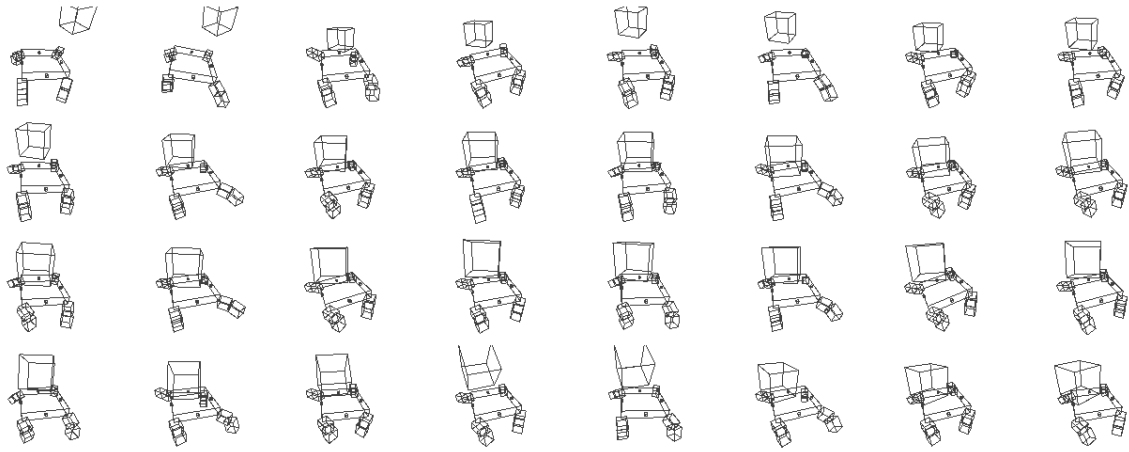
(a) Approach gait. The agent is moving toward the block from a distance. All limbs are contributing to the movement.
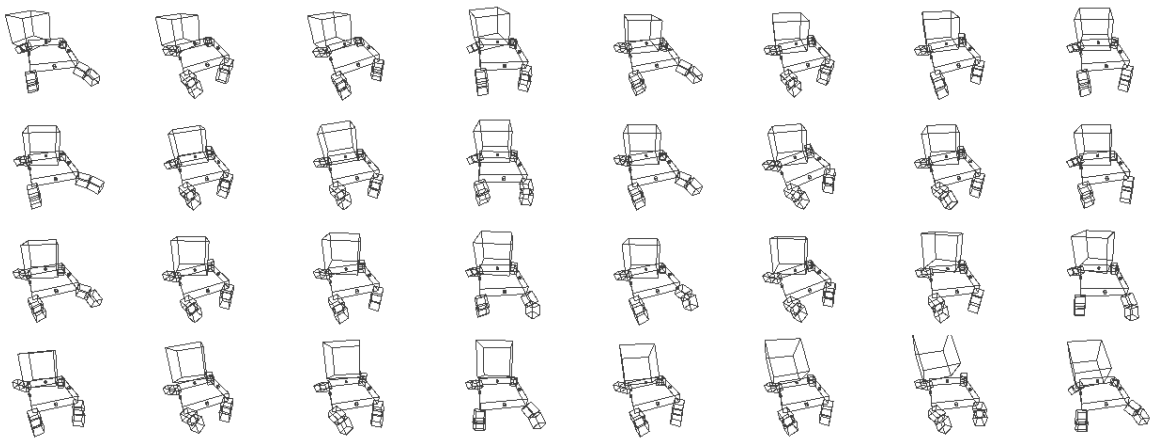
(b) Control gait. The agent is pushing forward with its 'back' limbs, maintaining the block between its forelimbs.

Figure 3: Zoopraxiscopic diagrams that show the gaits of the best evolved agent (run 11, individual 105). Presentation is in natural reading order, left-to-right, top-to-bottom. The viewpoint is fixed but tracks the agent as it moves through the world.

tically significant although weak correlation ($\rho = 0.38$; H_0 $p < 0.05$). The correlation between 3D RC and BD performance in the 2m BD trial is much stronger ($\rho = 0.51$; H_0 $p < 0.05$).

H3: Species evolved in the 3D RC task show increased performance after evolution in the BD environment. There is a clear improvement in all cases over the 25k tournament evolutionary run: the mean fitness over all naive populations was 37.29, compared to 124.16 in the evolved set (H_0 $p < 10^{10}$). Figure 5a shows progress of runs beginning from random genotypes over evolutionary time (100k tournaments in 1k tournament averages). Figure 5b shows the same view of populations beginning from naive genotypes, over 25k tournaments. Both treatments show a levelling off of fitness and there is no significant difference between the evaluation performance of the two starting conditions (figure 6).

H4: Some 3D RC species are on evolutionary trajectories more suitable for the BD task than others. We found a correlation between naive score and evolved scores across the 20 species ($\rho = 0.59$; H_0 $p < 0.01$) but no correlation between the naive scores and the magnitude of the change in fitness ($\rho = 0.26$; H_0 $p > 0.1$). Figure 7 demonstrates these relationships: δ-fitness is uncorrelated with naive fitness.

Conclusions and Discussion

We have shown that feedback motor control in evolved agents is possible with the given architecture, and that the architecture is flexible enough to support and adapt to a variety of evolutionary scenarios presented sequentially. This demonstrates that the platform has the potential to support environments that require even more sensorimotor control and is a reasonable starting point from where physical complexities can be added into the 3D RC task, eventually ap-
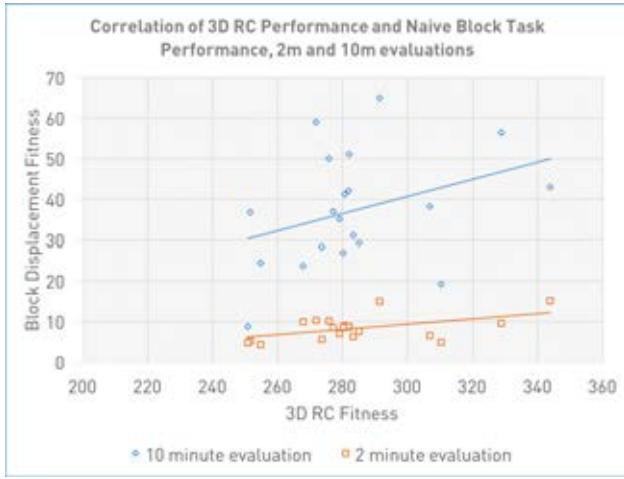
Figure 4: Across-species correlation comparing 3D RC performance and BD performance. Outcomes across the two tasks are more correlated when evaluation time is shorter ($\rho = 0.51$), indicating that movement speed is a factor in success in the block task and shared between the two problems. However, a strong gait is required to push the block and this is not selected for in the 3D RC task, hence the lesser correlation in the 10m task ($\rho = 0.38$).

(a) Fitness on the BD task (moving average over a 1000-tournament moving window) for evolution from a random (un-evolved) population.

(b) Fitness on the BD task (moving average over a 1000-tournament moving window) for evolution from a naive (evolved in 3D RC) population.

Figure 5: Progress of runs over evolutionary time; note that the x-axis differs due the different starting conditions and number of tournaments for each treatment.

proaching a full physical model of the problem. Observations of the agents' behaviour gained through 3D visualisation have revealed a rich variety of evolved strategies for solving the problem. Different classes of gait for approaching and manipulating the block appear due to the genetic heritage of species, and it is clear that low, heavy gaits work best for pushing the object in the BD task. From the deafferentation studies it can be seen that these gaits are not self-generating, blind gaits that simply aim the agent to the target location but are more complex aggregates of sensory data that depend on the agents position relative to the block in order to successfully achieve increased displacement.

When we consider the two evolutionary scenarios, 3D RC and BD, we found some overlap between the two problems. A strong correlation was observed between performance in the BD challenge before evolution in a two-minute evaluation, and performance at the end of the 3D RC task, indicating that some components of both challenges contribute similarly to relative agent fitness. This is likely to be the speed and directness of movement in the world which has a greater effect in a smaller evaluation period. As the evaluation period grows larger, this correlation decreases indicating that the block-pushing dimension of fitness in this scenario is not well captured in the 3D RC task and ultimately is the most important component. (It was also observed by measuring the time taken by agents to reach the block that most naive species sacrifice movement speed for block pushing capability during evolution, and this aspect should be investigated more thoroughly to determine whether this is an
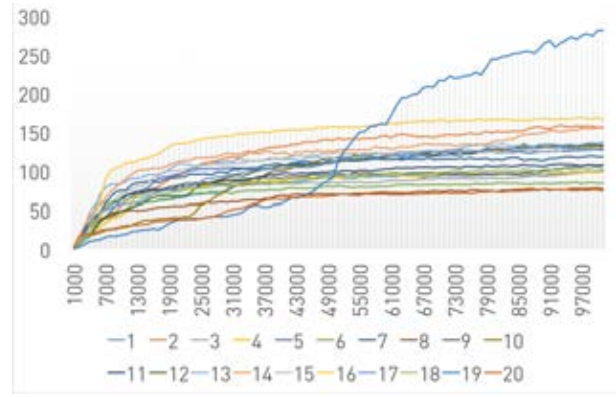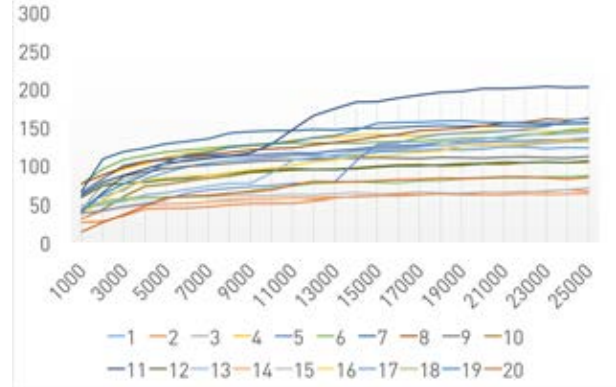
artefact or a consistent trend.) We showed that performance from either starting point (3D RC or unevolved genotypes) is comparable, demonstrating that an incremental approach incorporating both types of environment is possible in principle. We noted one extremely high-fitness run in the *random* category; upon visual inspection this species is a classic degenerate solution whose strategy is to rapidly vibrate the block to achieve high fitness. It is possible that more complex environments (such as 3D RC) prevent this kind of trivial solution by requiring a richer agent–environment interface. Our results comparing BD fitness before and after evolution demonstrate that whilst naive performance is an indicator of final performance, it is not an indicator of how much any particular species will improve. There is a risk that incrementally presenting new environments to only the most successful species could exclude good general solutions, a problem potentially mitigated by heterogeneous presentation of multiple environments.
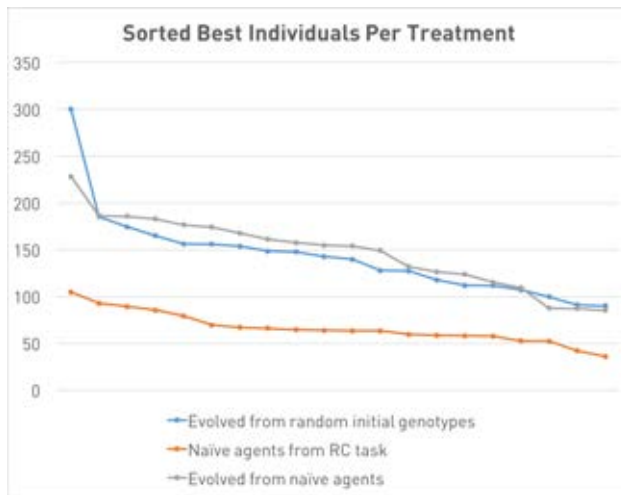
Figure 6: Comparison of the best individuals from the naive population, and from populations evolved from the random (unevolved) and naive-evolved populations.

Figure 7: Correlation of evolved fitness with naive fitness ($\rho = 0.59$), and delta fitness with naive fitness ($\rho = 0.26$).

Further work Ongoing work is toward the P3D RC task: a physically-embodied deliberative river crossing problem. The next step is to consider not just displacement but also positioning of the block using the shunting landscape. This is likely to demand significant revision of the underlying control architecture to incorporate reasoning about relative positioning. Additionally, the question of whether specific types of solutions in the 3D RC world have specific performance profiles in the BD world could be addressed by examining in detail whether some species always slow down and some always speed up. Additionally, it is possible that evolved morphology could significantly contribute to physical manipulation behaviours.

Acknowledgements

References

Beer, R. D. and Gallagher, J. C. (1992). Evolving dynamical neural networks for adaptive behavior. In *Adaptive Behavior 1*, pages 91–122.

Borg, J. and Channon, A. (2011). Discovering and maintaining behaviours inaccessible to incremental genetic evolution through transcription errors and cultural transmission. In *Advances in Artificial Life, ECAL 2011: Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems*, pages 101–108.

Grossberg, S. (1988). Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural networks*, 1(1):17–61.

Reil, T. and Husbands, P. (2002). Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation*, 6(2):159–168.

Robinson, E., Ellis, T., and Channon, A. (2007). Neuroevolution of agents capable of reactive and deliberative behaviours in novel and dynamic environments. In *Advances in Artificial Life*, pages 345–354. Springer.

Sims, K. (1994a). Evolving 3d morphology and behavior by competition. In *Artificial Life IV*, pages 28–39. MIT Press.

Sims, K. (1994b). Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, SIGGRAPH '94, pages 15–22, New York, NY, USA. ACM.

Stanton, A. and Channon, A. (2015). Incremental neuroevolution of reactive and deliberative 3d agents. In *Advances in Artificial Life, ECAL 2015: Proceedings of the Thirteenth European Conference on the Synthesis and Simulation of Living Systems*, pages 341–348. MIT Press.

Tange, O. (2011). Gnu parallel - the command-line power tool. *;login: The USENIX Magazine*, 36(1):42–47.

Yang, S. X. and Meng, M. (2000). An efficient neural network approach to dynamic robot motion planning. *Neural Networks*, 13(2):143 – 148.
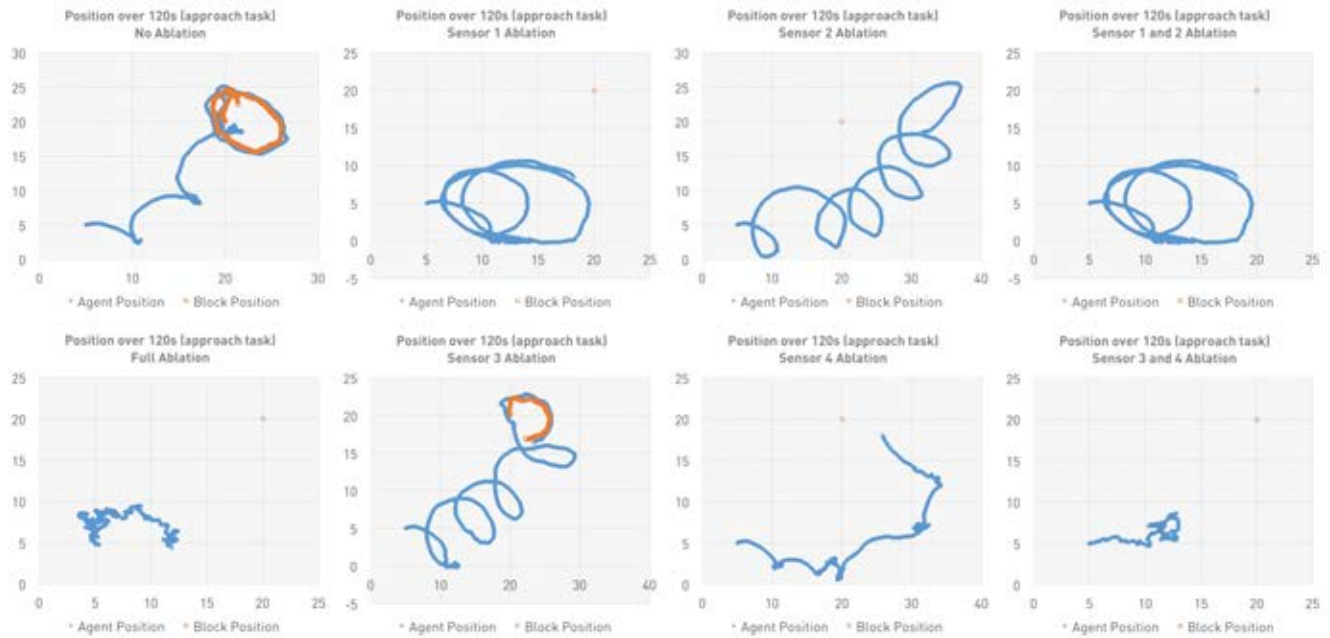
(a) Approach task. The agent begins at (5,5) and attempts to reach the block. The unaltered agent's trajectory is shown in the top left; this agent tends to overshoot its target and then correct by rotating, as the two loops in the path record. All sensors have some effect on this behaviour although sensor 1 is by far the most pronounced difference in a single cut. In complete deafferentation (bottom right) the agent moves randomly.
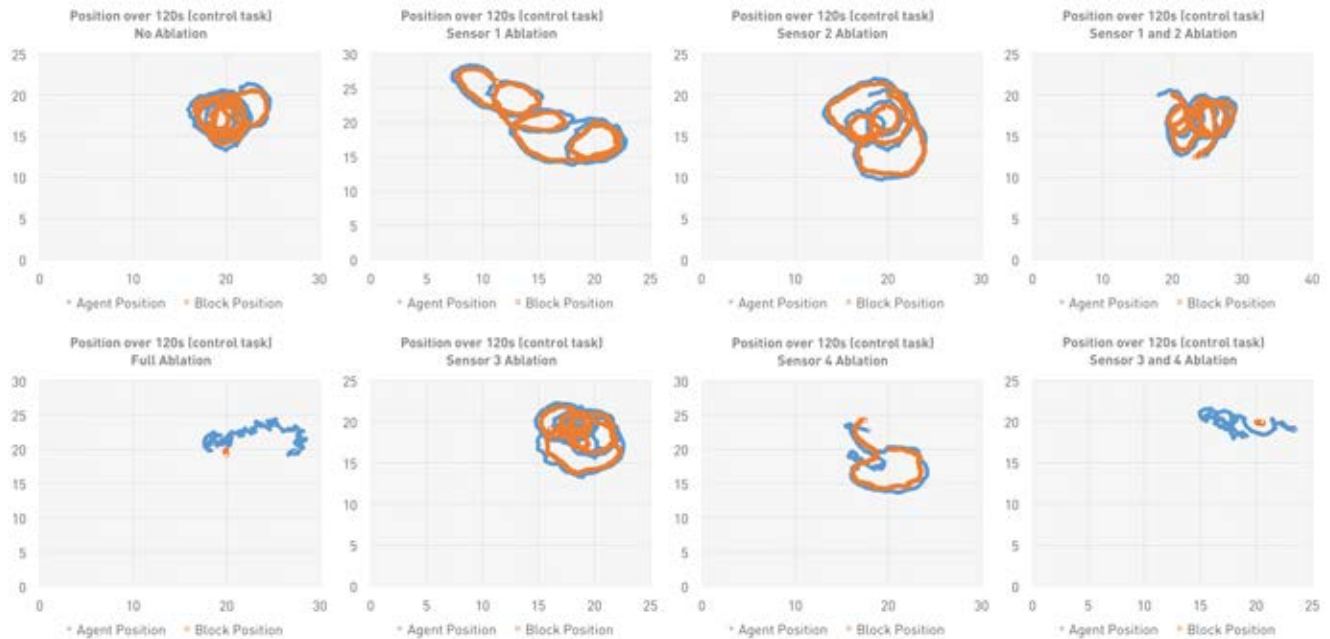
(b) Control task. The agent begins at (18,20), adjacent to the block. The unaltered agent pushes the block in a tight circle to maximise fitness (top left). Sensor ablations do not have a catastrophic effect as in the approach task; all single cuts still maintain block movement although the trajectory is less efficient, as does the dual cut of sensors 1 and 2. Only by cutting sensors 3 and 4 or complete deafferentation did we observe failure to displace the block at all.

Figure 8: Agent-block trajectories of best agent from best overall trained population under various sensor ablation treatments. The figure demonstrates how a combination of sensory inputs is necessary for reliable gait generation for distance approach and to control the block. In all cases the block initially rests at (20,20).

Evolving Specialisation in a Population of Heterogeneous Robots: the Challenge of Bootstrapping and Maintaining Genotypic Polymorphism

Arthur Bernard[1], Jean-Baptiste André[2] and Nicolas Bredeche[1]

[1]Sorbonne Universités, UPMC Univ Paris 06, CNRS, Institute of Intelligent Systems and Robotics (ISIR), F-75005 Paris, France
[2]Institut des Sciences de l'Evolution, Université de Montpellier, CNRS,
IRD, EPHE, CC065, Pl. E. Bataillon, 34095 Montpellier, France

Abstract

In this article, we are intested in the evolution of speciali-sation among a single population of heterogeneous robotic agents in a cooperative foraging task. In particular, we want to compare (1) the emergence and (2) fixation of genotypic polymorphism under two different selection methods: elitist and fitness-proportionate. We show that, while the emergence of specialists is easy under an elitist selection, this method cannot maintain heterogeneous behaviours throughout the whole simulation. In comparison a fitness-proportionate al-gorithm proves to be inefficient in evolving any cooperative strategy but ensures the conservation of heterogeneity when it is present in the population. We then reveal through addi-tional experiments two key factors for the evolution of het-erogenous behaviours in our task: (1) protection of genotypic diversity and (2) efficient selection of partners. We finally demonstrate this assertion and, while our main problem re-mains unsolved, we provide directions on how it could be successfully approached.

Introduction

Task specialisation is a defining characteristic in achieving efficient coordination and is thus considered to be crucial in the evolution of complex cooperative behaviours (Sza-thamàry and Maynard Smith, 1995). The problem of evolv-ing cooperation has been largely studied in evolutionary robotics as it raises interesting persepectives for the design of collective robotics (Trianni et al., 2007; Hauert et al., 2010; Doncieux et al., 2015). As a consequence, the man-ner in which robotic agents could evolve specialisation (or division of labour) for a cooperative task represents a com-pelling challenge in evolutionary robotics. As such, a large body of litterature has already been dedicated to this subject. However, most research focus on the particular case of ho-mogeneous groups of individuals (Waibel et al., 2009) as is classic in evolutionary robotics. This means that the indi-viduals are forced to rely on phenotypical plasticity (Waibel et al., 2006; Ferrante et al., 2015; Eskridge et al., 2015) and/or environmental cues (Waibel et al., 2006; Goldsby et al., 2010) in order to achieve specialisation.

In this paper, we focus on a slightly different problem: the evolution of a polymorphic population where division of labour is encoded at the genotypic level. More precisely, we want to study the evolution of a population containing two (or more) different types of genotypes. Each of these types of genotype should be able to encode for a differ-ent role without requiring the addition of mechanisms for lifetime specialisation. Thus it poses the problem of both *evolving* and *maintaining* genotypic polymorphism in a sin-gle population. Here we want to investigate the conditions under which specialised behaviours for a cooperative task can evolve in a single population of heterogeneous individ-uals. In particular, we are interested in the influence of the selection process in achieving division of labour.

We design a 2-robots cooperative foraging task where both a solitary and a cooperative strategies can evolve but where cooperation is highly rewarded. The genotype of each robotic agent is separately chosen in the population and the individuals therefore form an heterogeneous group. This task is greatly favored by the evolution of efficient coordina-tion strategies. In particular, our previous work on a similar task (Bernard et al., 2015) showed that two types of coop-erative strategy could evolve: one where both individuals adopt homogeneous behaviours (generalists) and the other one where they adopt a leader/follower strategy (specialists). Moreover, it was shown that the latter could only emerge be-tween heterogeneous individuals. As it is also the more effi-cient behaviour, we study the conditions for its emergence. The evolutionary dynamics of two popular selection meth-ods are studied: (1) an elitist $(\mu + \lambda)$ evolution strategy and (2) fitness-proportionate selection. Fitness-proportionate in particular is interesting with regards to genotypic polymor-phism as it is known to allow the evolution of frequency-dependent selection (Altenberg, 1991).

In the next Section, we introduce the experimental setup. Then we present the two types of cooperative strategies that can evolve. Next, we investigate whether any of the se-lection methods could evolve heterogeneous behaviours. In particular, we study for both schemes the evolutionary out-comes depending on whether the population is initially con-stituted of random individuals or seeded with pre-evolved efficient specialists. Then we present the results of com-

putational analyses in order to reveal and understand more deeply the mechanisms at play. In a final experiment, we reveal key mechanisms which could be investigated to solve this problem. Finally we discuss our findings and shed light on interesting perspectives for future work.

Methods

We evaluate two robotic agents in a 800 by 800 units square arena devoid of any obstacles except for the foraging targets. At the beginning of a simulation, 18 targets are randomly positioned in the environment. While the agents may move freely in the arena, the targets' positions are fixed. For a target to be collected, any agent needs to stay in contact with it for a specified amount of time (800 simulation steps). The target is removed after this duration and put back at another random position so that the number of targets is kept the same throughout a simulation. We consider that cooperative foraging happens if both individuals are in contact of the target when it is removed. When an agent collects a target, it is rewarded **50** if this target has been foraged in a solitary manner or **250** if both agents have cooperated to collect it.

Each agent is circular-shaped with a diameter of 20 units and possesses a collection of different sensory inputs. The first type of inputs is a 90 degrees front camera and is composed of 12 rays, each one indicating the type and distance to the nearest object (either another agent or a target). The other type of inputs are 12 proximity sensors evenly distributed around the agent's body. With a range of twice the agent's diameter, each proximity sensor outputs the proximity of the nearest obstacle in its range.

Both agents begin the simulation next to each other at the same end of the arena and can move according to the outputs of their neural network. This neural network is a fully connected multi-layer perceptron with one hidden layer. The inputs of the neural network are comprised of all the sensory information of the agent, i.e. 36 input neurons for the camera (3 inputs for each ray) and 12 for the proximity sensors. A final input neuron whose value is always 1 is used as a bias neuron. This amounts the total number of input neurons to 49. The hidden layer is constituted of 8 neurons while the 2 neurons of the output layer return the speed of the agent's wheels. A sigmoid is used as the activation function of each neuron. Finally, the topology of the network is kept constant during the experiments.

The population of individuals is evolved thanks to a classical evolutionary algorithm. The genotype of each individual is constituted of a collection of the 410 real-valued connection weights of the neural network. At each generation of the algorithm, every individual is evaluated by being successively paired with another individual randomly chosen in the population 5 times. Each pair interacts in the setting presented before during 20000 simulation steps which we call a *trial*. We perform 5 trials for each pair of individuals in order to decrease the impact of the targets' random positions

on the individuals' performance. The fitness score of an individual is computed as the average reward per trial.

The population for the next generation is created according to two different selection schemes :

- $(\mu + \lambda)$ **elitist selection:** the population of the next generation is constituted of the μ best individuals from this generation and λ offsprings sampled from the best individuals.

- **Fitness-proportionate:** offsprings are randomly sampled from the current generation to constitute the population of the next generation. The probability to sample a particular parent is proportional to this parent's fitness score.

Regardless of the selection method used, every offspring is a mutated clone of its parent and no recombination is used in our algorithm. The probability for each gene to mutate is 5×10^{-3} and mutations are sampled according to a gaussian operator with a standard deviation of 2×10^{-2}. Finally, experiments were conducted with the robotic 2D simulator of SFERESv2 (Mouret and Doncieux, 2010), a framework for evolutionary computation. You can find the source code for the experiments available for download at http://pages.isir.upmc.fr/~bredeche/Experiments/ALIFE2016-specialisation.tgz.

Behaviours of Specialists in a Cooperative Foraging Task

We showed in a previous article (Bernard et al., 2015) that two cooperative strategies could evolve in this particular task: *turning* (between two *turners*) and *leader/follower* (between a *leader* and a *follower*). Both of these strategies achieve cooperative foraging but with varied efficiency.

In the turning strategy, both individuals turn around one another so that they can keep the other individual in their line of sight and stay close to it (see Figure 1(a)). At the same time, the two individuals try to get closer to a target. This way, as soon as one of the two individuals is in contact with a target, the other individual can join it so the target may be collected cooperatively. Consequently, both individuals adopt a similar behaviour in this strategy and can be described as generalists.

In the leader/follower strategy, the individuals specialise in two roles: a leader and a follower. The leader always gets on the target first and checks rarely for its partner. In comparison, the follower tries to keep its leader in view during the entirety of the simulation so that it can get on the same target (see Figure 1(b)). Consequently, we observe the expression of two clearly heterogeneous behaviours which implies that both individuals are specialists. More importantly we also showed that, given our agents' capabilities, each phenotype needed to be encoded by a different genotype for specialisation to happen.
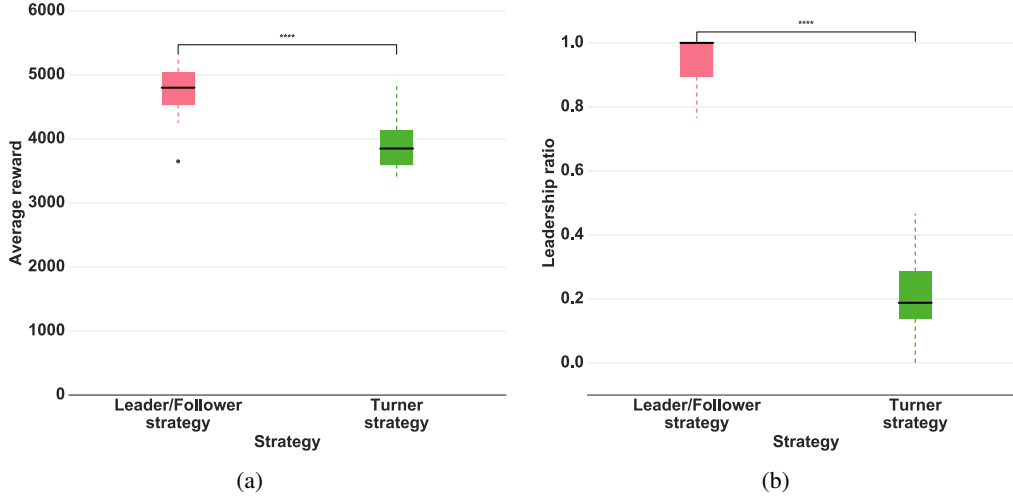
Figure 2: **Average reward and leadership proportion with a leader/follower or turning strategy** Boxplots of *(a)* the average reward and *(b)* the leadership proportion over 20 independent trials for the leader/follower and turning strategies. The leadership ratio of an individual represents the propensity for one individual among the pair to arrive first more often than its partner on a target collected in a cooperative fashion. The position of each target at the beginning of each trial was randomized.
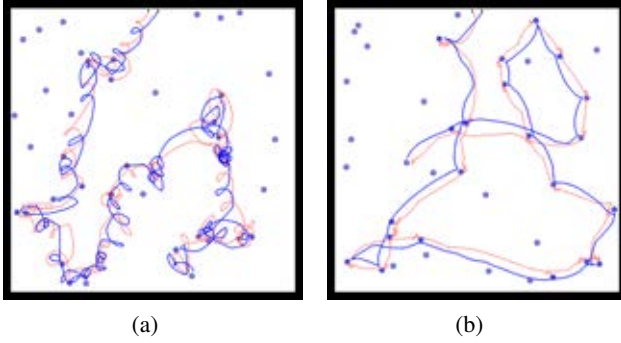
Figure 1: **Snapshots of the simulation after an entire trial in the foraging task.** The path of each robotic agent from their initial positions (black dots) is represented in red and blue. The blue discs represent the 18 targets in the environment. When a target is foraged by the two agents, a red cross (resp. blue) is drawn on the target if the red agent (resp. blue) arrived on it first. Each snapshot corresponds to a trial where agents adopted a different strategy: *(a)* turning or *(b)* leader/follower.

Figure 2(a) shows the efficiency of each strategy, defined as the average reward obtained by the two individuals during a simulation over 20 independent trials (with randomized targets' positions for each trial). We can see that, as expected, the leader/follower strategy achieves a significantly higher efficiency (Mann-Whitney U-test on the average reward over 20 trials, p-value < 0.0001). This difference in efficiency is directly correlated to a highly significant difference in the proportion of leadership as shown in Figure 2(b)

(Mann-Whitney U-test on the leadership proportion over 20 trials, p-value < 0.0001). We compute this proportion by looking at the propensity for one of the two individuals to arrive first more often on a target foraged cooperatively (i.e. the emergence of a leader).

Evolving Heterogeneous Behaviours with an Elitist Selection

Bootstrapping leader/follower strategies

In this first experiment, we are interested in the emergence of a leader/follower strategy when starting with a population of random individuals under an $(\mu + \lambda)$ elitist selection. In order to investigate the influence of population size, we tested three different sizes N: 20, 40 and 100. For each population size, we conducted 11 independent runs, each one lasting 90000 evaluations. For each population size N, we defined μ (i.e. the number of parents) and λ (i.e. the number of offsprings) as $\frac{N}{2}$. For example, when population size was 100, 50 individuals were kept from the previous generation and used to create 50 mutated offsprings.

Table 1 shows the repartition of the best individuals' strategies at the last generation of evolution for each population size. We consider a behaviour to be cooperative when more than 50% of the total number of targets collected are foraged cooperatively. First, we observe that in every replicate individuals always end up evolving a cooperative strategy. We also see that evolving a leader/follower strategy is difficult as specialists evolve in only 1 run (out of 33) and when the population size is 100. These results suggest that it is nearly impossible to evolve such heterogeneous behaviours with this setting.

Pop. size	# L/F Strat.	# Turning Strat.	# NC Strat.	Total
20	0	11	0	**11**
40	0	11	0	**11**
100	1	10	0	**11**

Table 1: **Strategies evolved by the best individuals under elitist selection with an initially random population.** Repartition of the different strategies adopted by the best individuals at the last evaluation in each of the replicates for different population sizes N. We indicate in each cell the number of simulations where a particular strategy evolved. Populations were evolved under an $(\mu + \lambda)$ elitist selection, with $\mu = \frac{N}{2}$ and $\lambda = \frac{N}{2}$. Individuals' genotype values were intially random. In the table "L/F" stands for leader/follower and "NC" for "Non-Cooperative".

However, when looking at the whole evolutionary history we can reveal additional information about the evolution of specialists. We show in Figure 3 the proportion of evolutionary time when the best individual of each run adopted a leader/follower strategy. This value is computed as the ratio of the number of generations when the leadership ratio was high enough (over a threshold value of 0.6) out of the total number of generations. We observe that even if the best individuals end up adopting a generalist strategy, this was not the case during the entirety of the evolution. In particular, there is a significant increase (Mann-Whitney, p-value < 0.05) in the number of generations where the best individual showed a leader/follower strategy when population size was 100 compared to a population size of 20. Therefore this implies that it is possible to evolve specialists but their stability in the population over time is nearly impossible to achieve.

Maintaining heterogeneity in a population seeded with specialists

In order to investigate the lack of stability of genotypic polymorphism under elitist selection, we design another experiment. We separately evolve a population of efficient *leader individuals* and *follower individuals* beforehand. We then replace the worst individuals w.r.t. fitness score in the population of leaders by a certain amount of followers. Our goal is to study if artificially constructing such population could result in the invasion and fixation of a stable leader/follower strategy.

The number of followers initially inserted in the population was varied according to two different settings: (1) we add only one follower or (2) we add an amount of followers equal to half of the population. Experiments were replicated 11 times during 90000 evaluations with population size of 40 and 100.

We show (Table 1) no significant differences in comparison to simulations with a population constituted of initially
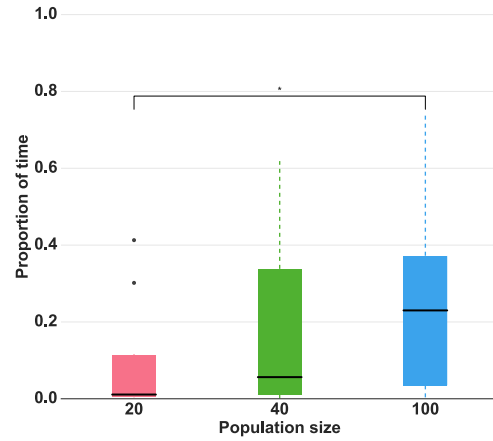
Figure 3: **Proportion of time with a leader/follower strategy.** Boxplots of the number of generations where the best individual in each replicate adopted a leader/follower strategy out of the total number of generations. We consider that the best individual adopted a leader/follower strategy when its leadership ratio was over a threshold value of 0.6.

random individuals w.r.t. the number of simulations where a leader/follower strategy evolved. These results suggest that even when purposely adding specialists, their stability in the population is still very hard to achieve. This implies that whether the behaviours are evolved from random genotypes or bootstrapped with efficient individuals is not as important as maintaining heterogeneity in the population. In particular, in only one replicate among the 3 runs where a leader/strategy was eventually adopted (out of 44) did the specialists initially added were maintained. In the 2 other runs we observe multiple emergences and disappearances of specialists throughout evolution.

Evolution Under a Fitness-Proportionate Selection

In this next experiment we want to investigate the evolution of heterogeneous behaviours when using a fitness-proportionate selection. As fitness-proportionate is known to allow frequency-dependent selection, we hypothesize that it may facilitate the evolution of specialists.

Bootstrapping leader/follower strategies

Similarly to the elitist selection, we replicated our experiments in 11 independent runs during 90000 evaluations. Likewise, population sizes were 20, 40 and 100.

We show in Table 3 that results are highly different when using such selection scheme. In particular, the fitness-proportionate selection performed poorly w.r.t. evolving cooperative strategies. For each population size, no cooperative strategy evolved at all in the vast majority of replicates. However in one particular run we do observe the emergence

Pop. size	Followers added	# L/F Strat.	# Turning Strat.	# NC Strat.	Total
40	1	0	11	0	**11**
40	20	0	11	0	**11**
100	1	1	10	0	**11**
100	50	2	9	0	**11**

Table 2: **Strategies evolved by the best individuals under elitist selection when adding followers.** Repartition of the different strategies adopted by the best individuals at last evaluation in each of the replicates for different population sizes N. We indicate in each cell the number of simulations where a particular strategy evolved. Populations were evolved under a $(\mu + \lambda)$ elitist selection, with $\mu = \frac{N}{2}$ and $\lambda = \frac{N}{2}$. The population was initially seeded with a population of leaders in which we added a specific amount of followers. In the table "L/F" stands for leader/follower and "NC" for "Non-Cooperative".

Pop. size	# L/F Strat.	# Turning Strat.	# NC Strat.	Total
20	0	1	10	**11**
40	0	1	10	**11**
100	1	2	8	**11**

Table 3: **Strategies evolved by the best individuals under fitness-proportionate selection with an initially random population.** Repartition of the different strategies adopted by the best individuals at the last evaluation in each of the replicates for different population sizes. We indicate in each cell the number of simulations where a particular strategy evolved. Populations were evolved under a fitness-proportionate selection. Individuals' genotype values were initially random. In the table "L/F" stands for leader/follower and "NC" for "Non-Cooperative".

and fixation of specialists. This is similar to what was observed under elitist selection w.r.t. evolving specialists.

Yet a closer look at the dynamics of evolution under a fitness-proportionate selection yields interesting results. In particular, there is not much variation in the strategy adopted by the best individuals throughout evolution. This is consistent with the fact that the bootstrap of a cooperative strategy was not observed in most of the replicates: fitness-proportionate is not efficient in evolving any cooperative behaviour. In consequence, there is not much variation in the proportion of individuals adopting a leader/follower strategy during evolution. As a matter of fact, we observe that in the only replicate where there was genotypic polymorphism at the end of the simulation, specialists were already present at the random initialisation of the population and did not evolve through mutation. This is very different with the elitist selection where we observe multiple emergences of specialists

(even briefly) during evolution in many different runs.

Maintaining heterogeneity in a population seeded with specialists

Pop. size	Followers added	# L/F Strat.	# Turning Strat.	# NC Strat.	Total
40	1	7	0	4	**11**
40	20	8	0	3	**11**
100	1	10	0	1	**11**
100	50	10	0	1	**11**

Table 4: **Strategies evolved by the best individuals under fitness-proportionate selection when adding followers.** Repartition of the different strategies adopted by the best individuals at the last evaluation in each of the replicates for different population sizes N. We indicate in each cell the number of simulations where a particular strategy evolved. Populations were evolved under a fitness-proportionate selection. The population was initially seeded with a population of leaders in which we added a specific amount of followers. In the table "L/F" stands for leader/follower and "NC" for "Non-Cooperative".

As expected from previous results, fitness-proportionate performs well in terms of stability of heterogeneous behaviours. We show in Table 4 that in the majority of replicates the best individuals adopt a leader/follower strategy at the end of the simulations. This is particularly true when population size is high enough (100). A major difference with the elitist selection is that in all replicates where a leader/follower strategy was observed at the end of the run, the specialists were maintained from the start throughout evolutionary time. These results suggest that, although not efficient at bootstrapping cooperative behaviours, fitness-proportionate performs well w.r.t. the stability of genotypic heterogeneity. Furthermore, we can hypothesize that this selection scheme is good at maintaining heterogeneity specifically because it largely fails (under our choice of parameters) at bootstrapping any cooperative strategy.

Computational Analyses of Population Dynamics

In this present section, our goal is to understand more deeply the dynamics at play which allow the invasion of suboptimal generalists even when efficient specialists are present. To that end we run computational analyses based on the expected fitness of each of the three phenotypes. Table 5 shows the average payoff of pair-wise simulations between each type of phenotypes. We consider the payoffs for both phenotypes in each pair to be identical as no significant differences were observed between their payoffs.

Several observations can be made directly from these results. First, we can confirm that the leader/follower strategy

Phenotype	*Leader*	*Follower*	*Turner*
Leader	1265	5000	3480
Follower	5000	100	2750
Turner	3480	2750	2755

Table 5: **Payoff matrix for pair-wise simulations of each phenotype.** Average payoffs of each phenotype against every phenotype in a pair-wise simulation. Each pair was evaluated 10 times in order to decrease the stochastic effects of the initial conditions (i.e. random positions of the targets).

displayed by a (*leader*, *follower*) pair is clearly the best strategy. However each one of these two phenotypes performs very poorly against itself with the worst payoff obtained by a pair constituted of two *followers*. Secondly, *turner* individuals perform also very well against *leaders*. Last, there is no significant differences w.r.t. payoffs when a *turner* is paired with a *follower* or another *turner*. These last two points hint at a shared lineage between *followers* and *turners*.

Indeed analyses of the genotypes' histories in our previous experiments reveal that *turner* individuals in fact descend from *follower* individuals. This means that they act as *followers* when interacting with *leaders* but are not as efficient. However they are a lot more efficient than *followers* when paired with individuals of the same phenotype (or *followers*).

From this payoff matrix, we run computational analyses to model the gradient of phenotypes' repartition in an infinite population. The fitness W of a particular phenotype i is computed as follows:

$$W_i = \sum_{j=1}^{M} P(ij) * F(j)$$

with j the phenotype it is paired with, M the number of different phenotypes (3), $P(ij)$ the payoff of phenotype i against j and $F(j)$ the proportion of phenotype j in the population. From this fitness, we can deduce the variation of phenotypes repartition by updating the proportion F of each phenotype i:

$$F_i = F_i * \frac{W_i}{\sum_{j=1}^{M} W_j}$$

We show in Figure 4(a) a vector field of this gradient. We can see that there actually exists an equilibrium between the three phenotypes (marked by the a dot at the crossing between the the dotted lines). This implies that even though the *turner* strategy is not the more efficient one, it is still expected that this phenotype can invade and coexist with the two other phenotypes.

We can hypothesize that we could not observe this equilibrium in our robotic simulations because of the stochastic

effects arising from selection in a finite population. In order to study this hypothesis we ran additional computational simulations based on the same payoff matrix. The initial population is entirely composed of *leaders* and the selection method is an elitist ($\frac{N}{2} + \frac{N}{2}$) evolution strategy where N is the population size. Every 10 generations, each offspring has a probability of $1 * 10^{-2}$ to mutate into any of the two other phenotypes.

Figure 4(b) shows the final repartition of phenotypes after 1500 generations of evolution for $N = 20$, $N = 100$ and $N = 1000$ in 11 independent replicates. We can see that when increasing population size we also increase the probability that an equilibrium where the three phenotypes exist is reached. We actually observe that the repartition of phenotypes at the last generation of evolution gets closer to the predicted equilibrium as population size increases. This implies that when population size increases, the probability to lose particular phenotypes decreases. In other words, the effect that the stochasticity of fitness evaluation has on the sampling of the genotypes for the next generation is mitigated: population size is essential to the maintenance of specialists.

General Properties for Evolving Heterogeneous Behaviours

From the previous Section, we can hypothesize two key properties for the successful evolution of genotypic polymorphism. First, we showed that population size needed to be large enough in order to decrease the probability that heterogeneity could be lost during the evolutionary time. Even under an elitist selection where the best individuals are immediately selected, the stochastic nature of fitness evaluation entails that there is no guarantee that both types get selected. This means that a performance biased selection may lead to the composition of the new population not accurately representing the genotypic diversity of the previous one. Therefore, there needs to be a mechanism for the preservation of genotypic diversity. Second, we previously saw that one key reason for the invasion of *turner* individuals is that, while *followers* perform badly against themselves, this is not the case for the formers. This means that the manner in which robots are paired is essential for achieving specialisation.

In order to test these hypotheses we design a last experiment where we diverge from the initial problem and now coevolve two separate populations. In this coevolution algorithm, each individual of one population is always evaluated against an individual of the other population (5 times as in previous experiments). Then, each population separately undergoes selection under an elitist $(10 + 10)$ selection method to create the population of the next generation (which means that each population size is 20). We conducted 11 independent replicates which lasted 90000 evaluations each. The populations were initially constituted of random individuals.

We show (Table 6) that when using coevolution, we al-

(a)

(b)
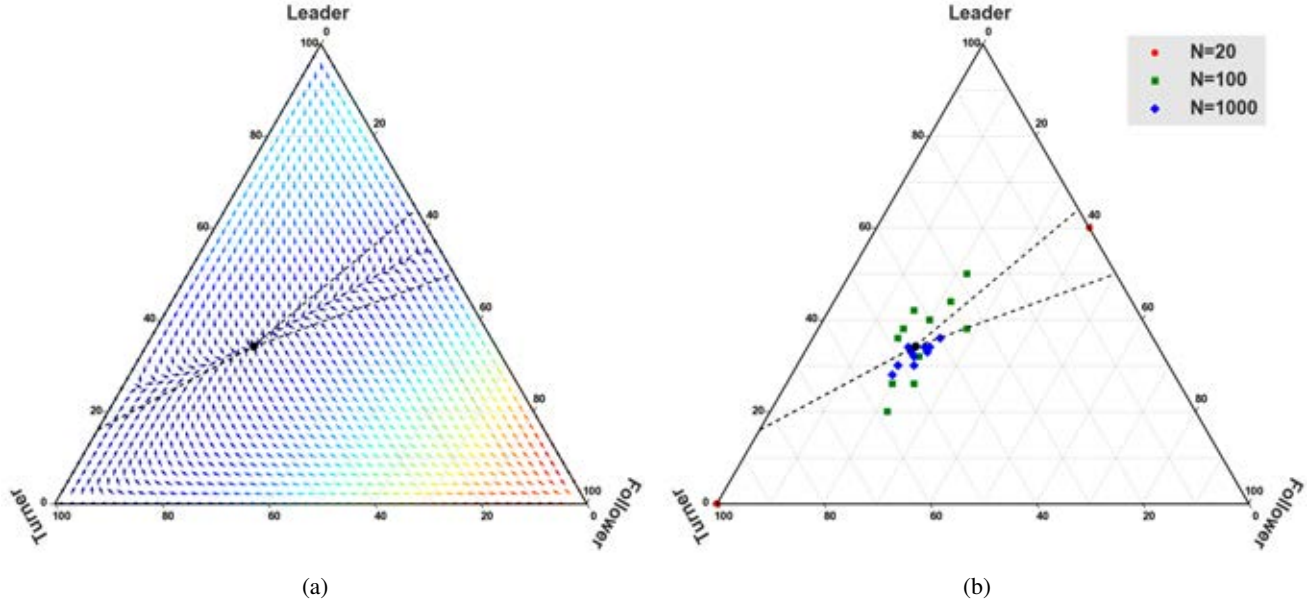
Figure 4: **Vector field of the gradient of phenotypes' proportions and proportions of phenotypes at the last generation of evolution.** *(a)* Vector field of the gradient of phenotypes' proportions in an infinite population. The strength of variation is indicated by the color of the arrow. *(b)* Repartition of phenotypes at the last generation of evolution for all three population sizes. Evolution lasted 1500 generations and results were replicated across 11 independent simulations. The initial population was entirely composed of *leaders*.

# L/F Strat.	# Turning Strat.	# NC Strat.	Total
11	0	0	**11**

Table 6: **Strategies evolved by the best individuals when coevolving two populations.** Repartition of the different strategies adopted bt the best individuals at the last evaluation in each of the 11 replicates. We indicate in each cell the number of simulations where a particular strategy evolved. Two populations were coevolved under elitist selection and the individuals' genotype values were initially random. In the table "L/F" stands for leader/follower and "NC" for "Non-Cooperative".

ways evolve specialists in every replicates. Moreover, this algorithm is highly stable as the heterogeneous behaviours that emerged were never lost during evolution in every replicates. This means that coevolution is highly efficient both for the bootstrap of a leader/follower strategy and its maintenance throughout evolution. Regarding our hypothesized properties, we can check that the coevolution algorithm respects both of them. Firstly, as populations are separately coevolved, we make sure that performance-based selection does not accidentally lead to the disappearance of specialists. Thus we ensure that the populations' genotypic diversity is protected. Secondly, we create a very specific pairing between individuals. Indeed individuals inside the same

population are never partnered with one another. This means that *followers* are always paired with *leaders*. As *turners* thus possesses no fitness benefit over the other phenotypes, their invasion is prevented. The question is open as to how to endow an algorithm working on a single population with such properties.

Discussion and Conclusions

In this paper, we investigated the evolution of specialisation through a leader/follower strategy in a cooperative foraging task. Our goal was to reveal the difficulties that arise when trying to evolve genotypic polymorphism in a single population. To that end, we mainly studied the dynamics of evolution with two different selection methods: an $(\mu + \lambda)$ elitist evolution strategy and fitness-proportionate selection.

We first showed that the long term evolution of a leader/follower strategy was nearly impossible with an elitist selection. However bootstrapping specialists was not a problem as we observed that they frequently emerged during evolution. The major obstacle was rather to maintain heterogeneity over evolutionary time. Indeed, even when adding efficient followers to a population of leaders to force the adoption of a leader/follower strategy, specialists couldn't be maintained. In comparison, the properties shown by the fitness-proportionate algorithm were quite the opposite. While it was almost not capable of evolving a leader/follower strategy (nor any other cooperative strategy),

the fitness-proportionate selection demonstrated high stability. It was therefore capable of maintaining specialists when present. We thus revealed two critical properties for evolving heterogeneous behaviours in a single population: *bootstrapping* these behaviours and *maintaining* them throughout evolution.

We then ran computational analyses and showed that while a pair of turners is indeed less efficient w.r.t. payoff than a pair of leader and follower, it is a lot more efficient than a pair of leaders or a pair of followers. As a result, these individuals can easily invade part of the population. Moreoever, we also showed that the maintenance of specialists was very sensible to population size. Performance-based selection can indeed affect heterogeneity in the composition of the next generation's population. Finally, a coevolution algorithm, which we showed to be always successful in evolving heterogeneous behaviours, solved both of these two problems with (1) *specific partners selection* as pairs were constituted of individuals from different populations and (2) *protection* of the behaviours evolved by applying selection separately on the two populations. While this algorithm is not concerned with genotypic polymorphism in a single population, it is useful to yield effective mechanisms which could be studied to solve our problem.

This raises several interesting perspectives on how to solve this problem. First, niche protection could prevent the disappearance of the efficient but unstable leader/follower strategy. As a matter of fact, coevolution is akin to a particular type of niches protection with 2 niches. However, we intend to investigate how we could implement such mechanism without specifying the explicit number nor the organization of the niches. Rewarding diversity (Lehman and Stanley, 2008) is also known as an effective way to protect novel behaviours and could be another promising direction. In particular, a multiobjective algorithm on performance and diversity (Doncieux and Mouret, 2014), by rewarding genotypic and phenotypic diversity, may protect evolved specialists.

Secondly, we showed that because partners were chosen randomly among the population, it created the opportunity for a "parasitic" strategy to invade. An interesting direction for future works could be to investigate restrictions in the choice of partners. For example it would be compelling to investigate how the individuals could evolve to select their partner based on genotypic or phenotypic information.

Acknowledgements

References

Altenberg, L. (1991). Chaos from linear frequency-dependent selection. *The American Naturalist*, 138(1):51–68.

Bernard, A., André, J.-B., and Bredeche, N. (2015). Evolution of Cooperation in Evolutionary Robotics : the Tradeoff between Evolvability and Efficiency. In *Proceedings of the European Conference on Artificial Life 2015*, pages 495–502.

Doncieux, S., Bredeche, N., Mouret, J.-B., and Eiben, A. E. G. (2015). Evolutionary Robotics: What, Why, and Where to. *Frontiers in Robotics and AI*, 2(4).

Doncieux, S. and Mouret, J.-B. (2014). Beyond black-box optimization: a review of selective pressures for evolutionary robotics. *Evolutionary Intelligence*, pages 1–18.

Eskridge, B. E., Valle, E., and Schlupp, I. (2015). Emergence of Leadership within a Homogeneous Group. *PloS one*, 10(7).

Ferrante, E., Turgut, A. E., Duéñez Guzmán, E., Dorigo, M., and Wenseleers, T. (2015). Evolution of Self-Organized Task Specialization in Robot Swarms. *PLoS computational biology*, 11(8).

Goldsby, H. J., Knoester, D. B., and Ofria, C. (2010). Evolution of Division of Labor in Genetically Homogenous Groups Categories and Subject Descriptors. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation - GECCO '10*, pages 135–142. ACM.

Hauert, S., Mitri, S., Keller, L., and Floreano, D. (2010). Evolving Cooperation : From Biology to Engineering. In Vargas, P., Di parolo, E., Harvey, I., and Husbands, P., editors, *The Horizons of Evolutionary Robotics*, pages 203–217. MIT Press, Cambridge.

Lehman, J. and Stanley, K. O. (2008). Exploiting Open-Endedness to Solve Problems Through the Search for Novelty. *Artificial Life XI*, (Alife Xi):329–336.

Mouret, J.-B. and Doncieux, S. (2010). SFERESv2: Evolvin' in the Multi-Core World. In *Proceedings of Congress on Evolutionary Computation (CEC)*, number 2, pages 4079–4086. Ieee.

Szathamàry, E. and Maynard Smith, J. (1995). The major evolutionary transitions. *Nature*, 374(6519):227–232.

Trianni, V., Ampatzis, C., Christensen, A. L., Tuci, E., Dorigo, M., and Nolfi, S. (2007). From Solitary to Collective Behaviours : Decision Making and Cooperation. In *Advances in Artificial Life*, pages 575–584.

Waibel, M., Floreano, D., Magnenat, S., and Keller, L. (2006). Division of labour and colony efficiency in social insects: effects of interactions between genetic architecture, colony kin structure and rate of perturbations. *Proceedings. Biological sciences / The Royal Society*, 273(1595):1815–23.

Waibel, M., Keller, L., Floreano, D., and Member, S. (2009). Genetic Team Composition and Level of Selection in the Evolution of Cooperation. *IEEE Transactions on Evolutionary Computation*, 13(3):648–660.

Functional Modularity Enables the Realization of Smooth and Effective Behavior Integration

Jonata Tyska Carvalho[1,2], Stefano Nolfi[1]

[1]Institute of Cognitive Sciences and Technologies, National Research Council, Via S. Martino della Battaglia, 44, Roma, Italia
[2]Center for Computational Sciences (C3), Federal University of Rio Grande (FURG), Av. Italia, km 8, Rio Grande, Brasil
jonatatyska@gmail.com & stefano.nolfi@istc.cnr.it

Abstract

In this paper we show how evolving robots can develop behaviors displaying a modular organization characterized by semi-discrete and semi-dissociable sub-behavioral units playing different functions. In our experiments, the development of differentiated behaviors is not realized through the subdivision of the control system into modules and/or through the utilization of differentiated training processes. Instead, it simply originates as a consequence of the adaptive advantage provided by the possibility to display and use functionally specialized behaviors. These are selected by evolution not only with respect to their capability to perform a given sub-function but also with respect to the capability to support smooth and effective transition with other behaviors. This is achieved by having different co-adapted behaviors and by evaluating the variation affecting the behaviors on the basis of the impact they have on the overall performance of the robots. Moreover this process enables the development of the ability to carry on preparatory actions that are necessary for the effective execution of the following behaviors. We refer to this type of modularity as functional modularity, since unlike structural modularity, it is not based on behavioral modules that are separated by clear boundaries and/or that are programmed or trained independently.

Introduction

The acquisition of new behavioral skills and the ability to progressively expand the behavioral repertoire represents one key aspect of natural intelligence and a fundamental capability of robots that operate in dynamic and uncertain environments. One way to achieve this objective consists in using a structural modular approach in which different layers or modules of the robots controller are responsible for the production of different corresponding behaviors and in which the behavioral repertoire of the robots can be expanded by adding new layers or modules in an incremental fashion. Indeed, the discovery and utilization of control architecture of this type (Brooks, 1986; Arkin, 1998) enabled the achievement of tremendous progress in robotics.

In structural modular architectures of this type each module has the following characteristics: it is responsible for the production of a specific behavior, it is separated from the other modules by clear boundaries, and it is programmed

or trained independently. These characteristics present advantages but also drawbacks that can outnumber the advantages, especially when there is a significant interdependence between the different behaviors. On the one hand, the fact that modules are separated by clear boundaries enables the utilization of a divide and conquer strategy which enables to divide the overall design problem into a set of partially independent simpler problems. Moreover, the separation and independence among modules potentially provide a straightforward solution for the realization of a progressive expansion of the behavior repertoire that can be realized through the progressive addition of new modules. On the other hand, it also inevitably leads to solutions in which the importance of the interdependence between the different behaviors is neglected. Furthermore, the rigid separation between the modules prevents the exploitation of solutions that require the introduction of minor modifications on previous developed modules/behaviors that might be crucial for the possibility to re-use previous capabilities for realization of new additional skills.

In that respect it is important to point out that the behavior of natural organisms typically displays a modular organization characterized by somewhat semi-discrete and semi-dissociable subunits, or sub-behaviors, playing different functions or sub-functions (West-Eberhard, 2003). These sub-behaviors are not completely separated, dissociable, and independent. The modular organization of behavior in natural organisms therefore is characterized by both discreteness and the possible presence of boundaries between sub-behaviors and by connectedness and integration among them (West-Eberhard, 2003). Moreover, it is important to consider that the effective execution of a behavior performing a given function often requires the execution of preparatory actions. For example, the effective execution of a grasping behavior requires the execution of preparatory actions that modify appropriately the posture of the hand already during the execution of the reaching behavior that precedes the grasping activity (von Hofsten and Ronnqvist, 1988).

In this paper we show how evolving robots can develop behaviors displaying a modular organization characterized

by semi-discrete and semi-dissociable sub-behavioral units playing different functions. In our experiments the development of differentiated behaviors is not realized through the subdivision of the control system into modules and/or through the utilization of differentiated training processes. Instead, it simply originates as a consequence of the adaptive advantage provided by the possibility to display and use functionally specialized behaviors. These are selected by evolution not only with respect to their capability to perform a given sub-function but also with respect to the capability to support smooth and effective transition with other behaviors. This is achieved by having different co-adapted behaviors and by evaluating the variation affecting the behaviors on the basis of the impact they have on the overall performance of the robots. Moreover this process enables the development of the ability to carry on preparatory actions that are necessary for the effective execution of the following behaviors. We refer to this type of modularity as functional modularity since, as in the case of structural modularity, it is characterized by the presence of differentiated behaviors achieving specialized functions but, differently from structural modularity, is not based on behavioral modules which are separated by clear boundaries and that are programmed or trained independently.

In the context of structural modular approaches, the possibility to realize smooth transitions between behaviors also depends on the arbitration mechanism utilized. In the case of competitive arbitration mechanisms, in which the transition between behaviors is achieved by suddenly shifting the control of the robot actuators from one module to another, the transitions tend to be abrupt. Instead, in cooperative arbitration mechanisms, in which multiple modules can concurrently control the robot actuator and in which the arbitration is realized by gradually changing the relative weight of the different modules (Arkin, 1998), the transitions between behaviors tend to be smoother. However, the type of behavior produced during the transition phase on the basis of the latter approach, is necessarily constituted by a weighted average of the behaviors that is produced on the basis of the single modules. This type of average behavior is not necessarily effective. Moreover, in a transition between two behaviors, this method does not provide a way to realize preparatory actions, i.e. actions that do not belong neither to the first nor to the second behavior but that represent a pre-requisite for the appropriate execution of the second behavior.

Our work is related to previous evolutionary robotics studies that have addressed the evolution of multiple behaviors (Izquierdo and Bhrmann, 2008; Seth, 2011; Schrum and Miikkulainen, 2012; Petrosino et al., 2013; Williams and Beer, 2013). In these experiments, however, the synthesis and the exhibition of multiple behaviors represented the only possible viable solution since the evolving robots were required to carry on mutually exclusive tasks (e.g. eating or avoid eating a specific food type (Seth, 2011; Petrosino et al., 2013)

or moving on the basis of a wheeled or legged actuators (Williams and Beer, 2013). In another related work Rahim et al. (2014) evolved neural network controllers that received as input the output produced by a set of pre-programmed modular controllers. Thus to the best of our knowledge, no previous studies focused on whether behavior differentiation and functional modularity can be observed on robots evolved for the ability to perform a single task.

The Method

To study this issue we decided to consider a cleaning experimental scenario in which a wheeled robot needs to vacuum clean the floor of an unknown in-door environment. We choose this problem since it represents the first (and still the most significant) successful application domain of autonomous robot solutions (Roomba, the first autonomous vacuum-cleaning robot developed by iRobots® under the supervision of Rodney Brooks and commercialized from 2002 has been sold in more than 10 million units to date, see IRobot (2013). Rather than designing the controller by hand, we studied whether effective controllers can be developed from scratch through an evolutionary method in which the evolving robots are selected on the basis of the percentage of successfully cleaned surface, i.e. on the basis of a scalar value that rates their overall ability to perform the task.

It is important to point out that we chose this domain also because it involves the execution of a task with a single goal (cleaning the environment) that does not necessarily require modular solutions. This enables us to study whether and how functionally modular robots evolve, whether and why behavior differentiation and functional modularity provide an advantage with respect to non-modular solutions, and eventually which are the characteristics and functions of the evolved sub-behaviors. In fact, domains involving multiple conflicting goals, such as those used in the literature addressing the study of action selection cited above, necessarily require the development of solutions characterized by multiple behaviors and implicitly constrain the number and type of required sub-behaviors.

The investigation of the cleaning problem also permits to compare our evolved solutions with those developed by companies that sell cleaning robots. In that respect, the fact that the behavioral policies displayed by different versions of the Roomba and by similar robots produced by other companies significantly differ (Palleja et al., 2010) demonstrates that finding the optimal solution/s of this problem is far from trivial.

The Task, the Environment and the Robot

To evolve robots that are robust with respect to environmental variations we evaluated each robot for 3 trials/cleaning sessions. At the beginning of each trial, the initial position and orientation of the robot in the environment, and

the specific characteristics of the environment, like wall dimensions, in which it was situated in were randomly varied within limits.

Each trial lasted 6 minutes and 15 seconds. This represents a rather short period of time, although performing a precise comparison with the time required by commercial robots to clean completely or almost completely a surface with similar properties is impossible due to the lack of similar data (for some indications see Palleja et al. (2010)).

To compute the cleaning performance we calculated the percentage of 20x20cm non-overlapping areas visited by the robot at least once during a trial. We used a concave environment (Figure 1) constituted by a large central area and by four peripheral corridors that represent a room-like environment. The average environment had a central area with a size of 6.8m and four corridors with a size of 3.78m in total. The exact size of the environment however was randomly set at the beginning of each trial. This was realized by varying the height and width of the central area and of corridors of $\pm 33\%$ and $\pm 18\%$, respectively, during different trials. The
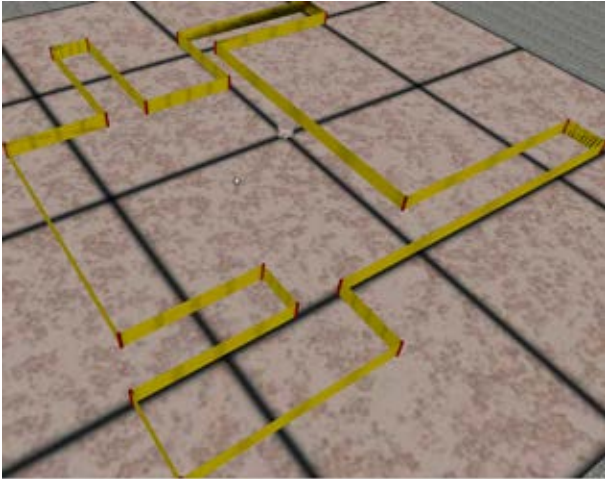
Figure 1: Example of the concave environment used

robot used was a MarXbot (Bonani et al., 2010), a differential drive wheeled robot with a diameter of 17cm. The robot is equipped with 24 infrared sensors evenly distributed along the robot's body and capable of detecting objects in a range of 10cm. Moreover, it was equipped with a rotating laser sensor capable of detecting obstacles at longer distance. Experiments were run in simulation by using the FARSA open-software tool (Massera et al., 2013) that includes an accurate simulator of the robot and of the environment.

The robots' neural controller

The robots are provided with a feedforward neural network controller without recurrence. In both experiments the robots are equipped with eight sensory neurons that encode the average activation state of eight groups of three adjacent

infrared sensors each and two motor neurons that encode the desired speed of the two robots wheels. The sensory neurons are fully connected with the motor neurons and to hidden neurons (if present), and the hidden neurons are fully connected to the motor neurons. Hidden and motors neurons are provided with biases. The state of the hidden and motor neurons is computed on the basis of the logistic function. The state of the sensory neurons, and the desired speed of the robots wheels are updated every 50ms. Experiments have been replicated in the following two experimental conditions:

(S) **Simple**: The robots are only provided with the infrared sensors.

(T) **Time**: The robots are provided with an additional sensory neuron that encodes the time passed since the beginning of the current cleaning session (trial), i.e. whose activation state linearly varies between 1.0 and 0.0 during the course of the trial. This sensor has been added to enable the robot to vary its behavior during the course of a cleaning session. Notice that this sensor enables the robot to access information extracted from the robot's internal environment (e.g. a robot clock situated inside the robot body) while the other sensors enable the robot to access information extracted from the external environment.

The connection weights and biases, that determine the robots behavior, are initially set randomly and evolved as described in the section below.

The evolutionary algorithm

The initial population consists of 20 randomly generated genotypes, which encode the connection weights and biases of 20 corresponding individual robots (each parameter is encoded by 8 bits and normalized in the range [-5.0, +5.0]). Every generation, each individual is evaluated for three trials in environments that randomly varied in dimension within the limits indicated above. The fitness of each trial is calculated by counting the percentage of 20x20cm portions of the environment that are visited by the robot at least once during the trial. The total fitness is calculated by averaging the fitness obtained during the three trials. All individuals are allowed to generate an offspring that is also evaluated for three trials. The 20 offspring are generated by creating a copy of the parent genotype and by mutating each bit with a 2% probability. The offspring genotype is used to replace the genotype of the worst parents or discarded depending on whether or not offspring outperform some of the parents. The genotypes of the initial population were generated randomly. Each evolutionary experiment was replicated 20 times starting from different randomly generated initial populations.

Results

In this section we first describe the performance achieved in the different experimental conditions. As we will see, the

cleaning task in this concave environment requires the exhibition of at least two sub-behaviors that differ in forms and functions: an exploration behavior that enables the robot to explore the large central area and a wall-following behavior that enables the robot to explore the peripheral areas and the borders of the central area. The possibility to discover and to display these two behaviors rather than a single undifferentiated behavior crucially depends on the characteristics of the robots neural controller as demonstrated by the fact that the behavior and the performance significantly vary in the two experimental conditions.

Then we will discuss the mechanisms that support behavioral differentiation and arbitration by analysing the behavioral solutions found in the different experimental conditions. As we will see, the two most important mechanisms that support the evolution of multiple behaviors are the ability to perceive and to generate affordances (i.e. opportunities for behaviors) and the possibility to flexibly and properly handle behavioral transitions.

Performance and efficacy of modular versus non-modular solutions

By post-evaluating the best robot of the last generation of each replication for 500 trials we can osee how the evolved robots reach close to optimal performance in the Temporal (T) experimental conditions and relatively low performance in the case of the simple (S) condition (Figure 2, top). The performance of each experimental condition statistically differs from each other (Mann-Whitney U, p<0.05). The performance obtained in the experiments in which the robots were also provided with the internal neurons (Figure 2, bottom) does not significantly differ from the experiments without internal neurons (Mann-Whitney U, p>0.05).

The analysis of the behaviors displayed by the best robots of the last generation indicates that the performance level correlates with the ability of the robots to display multiple behaviors. This is clearly illustrated by the behavior displayed by the best (S) and (T) robots that achieved a fitness of 67.4% and 82.8%, respectively. While (S) displays a single uniform behavior along the trial (figure 3, top), (T) is capable of performing two well-differentiated behaviors (Figure 3, bottom).

Indeed, the best robot with a simple architecture (S) always behaves in the same manner during the successive phases of the trial (Figure 3, top-left). In particular it avoids walls and obstacles by sharply turning with an angle of 45-90 degrees (depending on the relative angle with which the robot approaches the obstacle) and moves straight when it is far from obstacles. Through the exhibition of this behavior the robot spends most of the time exploring the large central portion of the environment and only occasionally it explores the peripheral corridors when it happens to approach them with a direction that is almost orthogonal to the entrance of the corridor. The robots of the other replications of the ex-

periments show qualitatively similar behaviors (results not shown). The best robot with the time neuron architecture
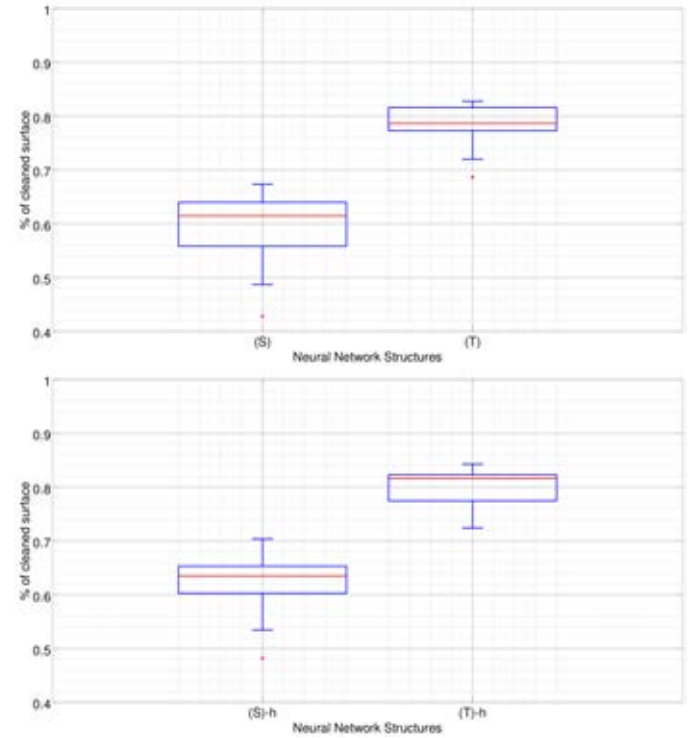
Figure 2: Boxplots of performance in the cleaning task. The top and bottom figures report the results obtained without internal neurons and with internal neurons, respectively. The boxplots display the performance of the best robot of the last generation in the two experimental conditions, i.e. in the simple (S) and temporal (T) conditions. Each box displays the performance of the best robot of 20 replications of each experiment. The performance is indicated by the percentage of cleaned cells within the walls. The value corresponding to optimal performance is unknown but is reasonably below 1.0 given that the robots have a rather limited cleaning time.

(T), instead, shows two well differentiated behaviors: (i) an initial exploration behavior that is realized by producing a progressively larger curvilinear trajectory that enables the robot to explore the large central portion of the environment, and (ii) a wall-following behavior that enables it to explore all the peripheral areas of the environment (Figure 3, top-right). Although the way in which the exploration behavior is realized varies in different replications of the experiment, well-differentiated exploration and wall-following behaviors are clearly observable in all cases (results not shown). The high performance of these robots is due to their ability to display different behaviors, which are specialized for the exploration of large open areas and peripheral areas, and to carefully tune the time duration of the two behaviors. Indeed, the relative duration of the two behaviors determines

whether the robot spends enough time exploring the central large area while keeping enough time to explore all the peripheral areas of the environment. A qualitative analysis of the first 10 replications showed that in the best two robots, that clearly outperform the best robots of the other 8 replications, the transition occurs at 3.17±0.11min. This transition time is optimal or nearly optimal as demonstrated by the fact that post-evaluation tests performed by slowing down or speeding up the robots internal clock and consequently the behavior transition led to significantly worse performance (results not shown).
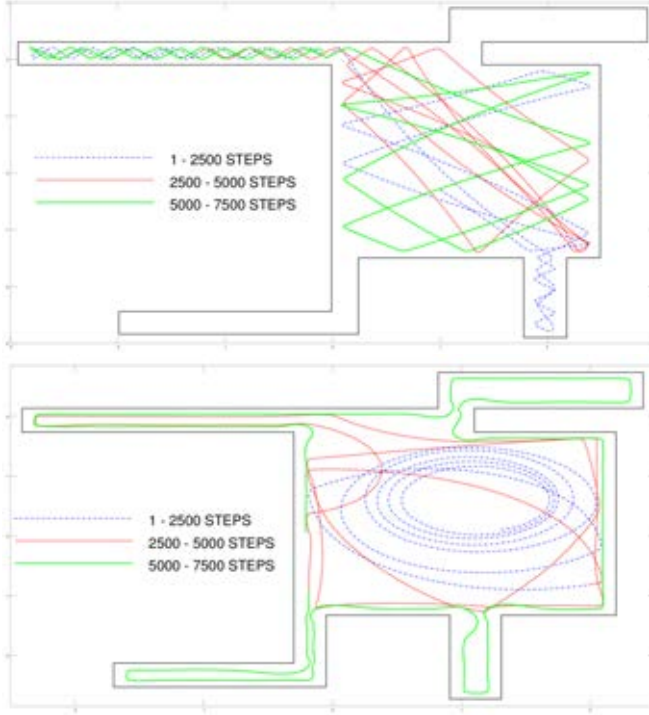
Figure 3: Typical trajectories displayed by the best robots of the two experimental conditions without hidden units. The portions of the trajectory produced during the first, second, and third part of the trial (i.e. from step 1 to 2500, from step 2501 to 5000, and from step 5001 to 7500, respectively) are shown with different colours and line style.

On the mechanisms supporting behavior differentiation and arbitration

We have seen how controllers presenting the ability to display multiple behaviors, can enable the adaptive robots to achieve better performance and that the emergence of this ability depends on the characteristics of robots neural controllers. We will now focus on the mechanisms supporting behavior differentiation and arbitration.

Before entering into this, it is important to point out that the behavior displayed by an embodied and situated agent is a dynamical process unfolding in time that results from the robot/environmental interactions. This implies that the organization of behavior/s vary at different time scales. Moreover, this implies that the sensory states experienced by the robot at a given time step are co-determined by the actions produced by the robot during previous robot/environmental interactions. If we use the term affordance introduced by Gibson (1979) to indicate sensory states that elicit the production of behaviors, this implies that the affordances are not only extracted through sensors from the internal and/or the external environment but are also generated by the robot itself through actions.

The analysis of the behavior exhibited by the robots at a short time scale (i.e. at a time scale of seconds) indicates that in all experimental conditions robots tend to exhibit at least two different low-level behaviors: (i) an obstacle-avoidance behavior that consists in turning while the robot detects an obstacle on its frontal side, and (ii) a move-forward behavior that consists in moving straight or almost straight while the robot does not detect obstacles in its frontal side. This implies that at a short time scale all robots of all experimental conditions displayed a certain kind of functional modularity. The reasons that explain why this type of modularity always evolve are that it plays a fundamental role (i.e. it enables the robot to avoid being stuck and to keep exploring the environment) and that it is supported by the availability of always available and easy to use affordances. Indeed, independently from the way in which the robot behaves, it will always experience a lack of activation on the frontal infrared sensors when the robot/environment context affords a move-forward behavior and an activation on the frontal infrared sensors when the robot/environmental context affords an obstacle-avoidance behavior. The infrared sensors therefore always enable the robot to perceive when the former or the latter behavior should be produced and when the transition between the two behaviors should occur.

This ideal situation, however, in which the robot can rely on robust and ready-to-use affordance states only characterize few lucky cases (incidentally, this probably explains why the combination of obstacle-avoidance and navigation behaviors represents a widely used experimental scenario in robotics). In other cases, the affordance states supporting behavior differentiation and arbitration should be extracted through internal elaboration and/or generated through the exhibition of appropriate behaviors.

As we have seen in the previous section, the studied cleaning task requires behavioral diversification also at a higher time scale, e.g. it requires the exhibition of an exploration and a wall-following behavior lasting for minutes. In this case however, the robot cannot rely on ready-to-use affordances that indicate when the robot should display the first or the second behavior and when the robot should switch from one behavior to the other. To achieve this kind of modularity the evolving robots should find a way to: (i) keep

producing the same behavior for a prolonged period of time, (ii) switch behavior at the right moment, and (iii) realize a suitable transition during behavior switch. We will illustrate in details how the evolved robots manage to master these requirements in the next three sub-sections.

Notice that the evolution of context-dependent behaviors require the concurrent development of two interdependent skills, the ability to produce a new behavior and the ability to regulate appropriately when the new behavior should be exhibited (West-Eberhard, 2003).

Producing behaviors for prolonged periods of time

All evolved robots solve the problem of producing a given behavior for a prolonged period of time by realizing each behavior in a way that ensures that they keep experiencing stimuli of the right type during the execution of that behavior. In cases in which the robots should exhibit two differentiated behaviors, i.e. an exploration and a wall following behavior, this implies that they should realize the former and the latter behaviors in a way that ensures that they keep experiencing stimuli of type 1 and 2 while they exhibit the former or the latter behavior, respectively, and should react to the stimuli of the two types by producing actions that enable them to keep producing the former or the latter behaviors, respectively. The two classes of stimuli thus assume the role of affordance for the first and for the second behaviors, respectively. These affordances are not directly available from the environment, as in the case of the states affording the obstacle avoidance and move-forward behavior discussed above, but are generated by the robots themselves through their actions (i.e. through the ability to realize each behavior in a way that ensure that the robot keeps experiencing the corresponding affordances). This form of dynamical stability presents some similarities with the one that can be obtained in situated agents through homeokinesis (Der and Martius, 2012), a task-independent learning process that can enable situated robot to synthesize temporarily stable behaviors, despite the mechanism and the processes through which this is realized are completely different.

All (S) and (T) robots exploit this affordance generation mechanism. However, in the case of the (T) robots, they also exploit an additional mechanism that contributes to enable the robots to keep producing each behavior for a prolonged period of time.

The problem of producing the same behavior for a prolonged period of time is also solved by exploiting the cue provided by the state of the temporal neuron. Indeed, whether the robot keeps producing the exploration behavior or switches to the wall-following behavior also depends on the state of the temporal neuron (see Figure 4). The state of the time neuron influences the duration of the exploration behavior only during a critical phase, i.e. when the state of the time neuron is smaller than 0.6 and greater than 0.4. During the rest of the trial the ability of the robot to keep

producing the exploration behavior or the wall-following behavior rely on the affordance generation mechanism described above. Interestingly, in the case of the best (T) robot, the temporal neuron is also used to progressively vary over time the way in which the exploration behavior is realized so to regulate the probability that the robot keep experiencing sensory state affording the execution of the exploration or wall-following behaviors. Indeed, by initially moving forward and turning left of several degrees, the robot eliminates completely the possibility to encounter a wall on its left side (i.e. the possibility to experience stimuli affording the alternative wall-following behavior). Then, by moving forward and progressively reducing the angle of turn over time, the robot becomes progressively less adverse with respect to the possibility to experiencing stimuli affording the wall following behavior. This brings us to the question of how robots manage to switch behavior.
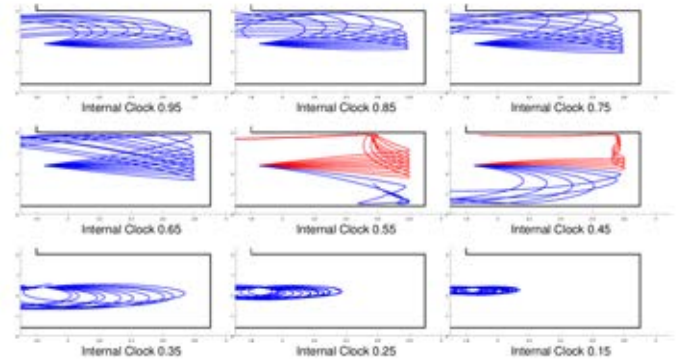
Figure 4: Behavior produced by the best (T) robot during different trials in which it started from the same initial position with systematically varied orientations and systematically varied state of the time neuron. The red and blue lines represent the trajectories produced by the robot during trials in which it switches or does not switch to the wall-following behavior, respectively. The black lines represent the walls. For sake of clarity we only show the local portion of the environment in which the robot is located.

Switching between alternative behaviors

The problem of switching between different behaviors is also solved through affordance generation. To understand how robots can act in a way that enable them to both experience stimuli affording the current behavior and stimuli affording the alternative behavior, we should reformulate the definition of affordance generation in probabilistic terms. Evolved robots solve the problem of producing a given behavior for a prolonged period of time and the problem of switching behavior by realizing each behavior in a way that ensures that they keep experiencing stimuli affording the current behavior with a given high probability and

stimuli affording the alternative behavior with a given low probability, respectively.

In the case of the robot evolved in the (T) experimental condition, the switch is regulated by both the stimuli experienced by the robot (i.e. by affordance generation) and by the cue provided by the robots internal clock. This double regulation enables the best (T) robot to carefully balance the time allocated to the two types of behavior and to reduce the variability among trials (i.e. the transition occurs 3.17 ± 0.11min). The double regulation process is demonstrated by the analysis of the trajectories produced during a series of trials in which the robot always start from the same position and the orientation of the robot and the state of the time neuron are systematically varied. As shown in the Figure 4, whether the robot switches the wall-following behavior depends both on the state of the internal clock and on the state of the infrared sensor when the robot approaches the wall. Overall this shows that whether the switch between the two behaviors occurs or not depends both on the state of the internal clock and on the way in which the exploration behavior is realized which, in turn, influences the type of stimuli that the robot experiences. As mentioned above, in the case of the best (T) robot, the state of the time neuron is not only used to regulate the probability that the robot switches behavior directly (the probability that the robot initiate a wall following behavior in a given relative position in the environment) but is also used to regulate the way in which the exploration behavior is realized which in turn influences the probability that the robot will later experience stimuli affording the wall-following behavior.

Realize suitable transitions during behavior switch

The connectedness of behaviors, i.e. the fact that alternative behaviors are semi-discrete and semi-dissociable units that are only partially independent, implies that the transitions between behaviors should be handled with care. In the case of our experiments, in particular, the transition between the exploration and the wall-following behavior requires special care since the latter behavior can only be produced when the robot is located near a wall and when the wall to be followed is located on a specific side of the robot. Indeed, the analysis of the evolved robots shows that the way in which the behavior transitions are handled in evolved robots has an important impact on robots performance.

The best solution to the transition problem was discovered by the two best replications of the (T) robot (see Figure 3, bottom). Indeed, as we mentioned above, this robot exploits the cue provided by the internal clock to gradually modifying the exploration behavior so to ensure that the robot will always reach a relative location with respect to the walls from which the wall-following behavior can be effectively triggered during the critical period (i.e. during 3.17 ± 0.11min). Overall this leads to an extremely timely, smooth and effective transition that enables these robots to outperform all others robots.

The importance of realizing smooth transitions and the importance of executing preparatory actions can be appreciated by observing the cases in Figure 4, in which the values of the internal clock are set to 0.35, 0.25 and 0.15. In natural conditions, when the internal clock assumes these values, the robot always produces the wall-following behavior. The robot, however, is only able to initiate a wall-following behavior when it is located near a wall that is situated on its right side. If this prerequisite is not satisfied the wall-following behavior will not be exhibited. In normal conditions this problem never arise since the robots have evolved the ability to perform, during the execution of the exploration behavior, the preparatory actions that enable the successive execution of the wall-following behavior.

Conclusions

In this paper we showed how robots evolved for the ability to perform a cleaning task can develop functional modular solutions that involve the exhibition and the alternation of differentiated behaviors playing specialized functions (i.e. cleaning large open areas and cleaning narrow peripheral areas, respectively).

The development of differentiated behaviors, in our experiments, is not realized through the subdivision of the control system into modules and/or through the utilization of differentiated training processes. Instead, it simply originates as a consequence of the adaptive advantage provided by the possibility to display multiple differentiated behaviors. Indeed, robots displaying multiple differentiated behaviors achieved better performance with respect to robots displaying a single behavior.

This approach provides a series of advantages with respect to structural modular approaches in which the synthesis of robots displaying multiple behaviors is realized by using well-separated control modules that are responsible for the production of different corresponding behaviors and that eventually are designed and or trained independently. In particular it releases the designer from the burden to identify the way in which the overall problem can be decomposed in sub-problems to be solved through the exhibition of specialized behaviors. More importantly, it enables the adapting robots to develop behaviors that are not only optimized with respect to the capability to accomplish the corresponding functions but that are also optimized with respect to their ability to operate effectively together. More specifically the adapted behaviors are realized in a way that ensure a smooth transition between alternative sub-behaviors and in a way that ensure that the preparatory actions that are necessary to initiate or to carry on a given behavior are realized before the robot initiates that behavior.

The analysis of the obtained results indicates that the mechanisms that support the evolution of functionally modular solutions are the ability to perceive affordances (i.e. per-

ceptual states encoding opportunities for behaviors) and the ability to realize smooth and effective transitions between different behaviors.

The perception of affordance constitutes a prerequisite for the possibility to develop differentiated behavior and for the possibility to effectively arbitrate them, i.e. selecting the behavior that is appropriate for the current robot/environmental context and regulating the duration of each behavior. Interestingly, the basic mechanism that is used by evolving robots to perceive affordances is affordance generation, i.e. the ability to realize each behavior in a way that ensures that the robot keeps experiencing sensory state affording the current behavior with a given high probability and sensory states affording alternative behaviors with a given low probability.

The limitations of this affordance generation mechanism, e.g. the inability to finely tune the duration of behaviors, are overcome by using additional regulatory processes that rely on internal cues. In particular, in the case of the best evolved robot this is realized by complementing the basic affordance generation mechanism with two additional regulatory processes. One of them consists in using the state of the internal clock to progressively vary the way in which the exploration behavior is realized so to progressively increase the probability that the robot will experience stimuli affording the wall-following behavior. The other additional regulatory process consists in using the state of the internal clock to vary qualitatively the way in which the robot reacts in a specific environmental situation (e.g. to determine whether the robot avoid an obstacle by turning left or right that in turn determine whether the robot will keep producing the exploration behavior or will switch to the wall-following behavior).

Overall this implies that behavior arbitration in the best evolved robots is realized through the combined effects of multiple partially redundant regulatory processes that operate through weak interactions.

Future studies should investigate whether this approach can enable evolving robots to find effective solutions in more complex tasks/scenarios.

References

Arkin, R. C. (1998). *Behavior-based Robotics*. MIT Press.

Bonani, M., Longchamp, V., Magnenat, S., Retornaz, P., Burnier, D., Roulet, G., Vaussard, F., Bleuler, H., and Mondada, F. (2010). The marXbot, a miniature mobile robot opening new perspectives for the collective-robotic research. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4187–4193.

Brooks, R. (1986). A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation*, 2(1):14–23.

Der, R. and Martius, G. (2012). *The Playful Machine: Theoretical Foundation and Practical Realization of Self-Organizing Robots*. Springer Science & Business Media.

Gibson, J. J. (1979). *The ecological approach to visual perception*. Houghton Mifflin.

IRobot (2013). Our history. `http://www.irobot.com/about-irobot/company-information/history.aspx`. [Online; accessed 29-April-2015].

Izquierdo, E. and Bhrmann, T. (2008). Analysis of a Dynamical Recurrent Neural Network Evolved for Two Qualitatively Different Tasks: Walking and Chemotaxis. In *AL-IFE*, pages 257–264.

Massera, G., Ferrauto, T., Gigliotta, O., and Nolfi, S. (2013). FARSA: An Open Software Tool for Embodied Cognitive Science. In *Advances in Artificial Life, ECAL 2013*, pages 538–545. MIT Press.

Palleja, T., Tresanchez, M., Teixido, M., and Palacin, J. (2010). Modeling floor-cleaning coverage performances of some domestic mobile robots in a reduced scenario. *Robotics and Autonomous Systems*, 58(1):37–45.

Petrosino, G., Parisi, D., and Nolfi, S. (2013). Selective attention enables action selection: evidence from evolutionary robotics experiments. *Adaptive Behavior*, 21(5):356–370.

Rahim, S. A., Yusof, A. M., and Brunl, T. (2014). Genetically Evolved Action Selection Mechanism in a Behavior-based System for Target Tracking. *Neurocomput.*, 133:84–94.

Schrum, J. and Miikkulainen, R. (2012). Evolving Multimodal Networks for Multitask Games. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(2):94–111.

Seth, A. K. (2011). Optimised agent-based modelling of action selection. In *Modelling Natural Action Selection*. Cambridge University Press.

von Hofsten, C. and Ronnqvist, L. (1988). Preparation for grasping an object: a developmental study. *Journal of Experimental Psychology. Human Perception and Performance*, 14(4):610–621.

West-Eberhard, M. J. (2003). *Developmental Plasticity and Evolution*. OUP USA, Oxford ; New York.

Williams, P. and Beer, R. (2013). Environmental feedback drives multiple behaviors from the same neural circuit. In *Advances in Artificial Life, ECAL*, volume 12, pages 268–275.

Genetics

Towards the construction of a DNA genome replication system for an artificial cell

Yoshihiro Sakatani[1] and Norikazu Ichihashi[1, 2]

[1] Graduate School of Information Science and Technology, Osaka University, Yamada-oka 1-5, Suita, Osaka, Japan
[2] Graduate School of Frontier Bioscience, Osaka University, Yamada-oka 1-5, Suita, Osaka, Japan
ichihashi@ist.osa-ka-u.ac.jp

Abstract

In vitro constitution of biological functions is a useful strategy to understand the physicochemical principles underlying biological functions (Luisi and Stano, 2013). To date, various cellular functions have been artificially constituted, including self-replication of genetic information, a gene expression, and so on. As a self-replication system of genetic information, our group has reconstituted an RNA genome replication system coupled with translation (Ichihashi, et al. 2013). However, all living organisms have the DNA genomes, which replicate with DNA replicases translated from the genomes via mRNA transcription. A plan for the construction of such a DNA replication system coupled with transcription and translation has been proposed using rolling-circle-type replication scheme approximately 10 years ago (Fig.1, Forster and Church, 2006), although it has not been realized yet. In this study, we attempted to constitute the transcription-and translation-coupled DNA replication (TTcDR) system and perform an evolution experiment to make the system recursive. The aim of the present work is to show a method of the construction of a DNA genome replication system which leads to the *in vitro* construction of an artificial cell.

We prepared a circular DNA encoding phi29 DNA polymerase gene under control of T7 promoter. The circular DNA was mixed with the reconstituted cell-free translation system derived from *Escherichia coli* (Shimizu, et al. 2001), T7 RNA polymerase and RNase inhibitor. To perform rolling-circle replication, we also added dNTPs, a random oligo DNA as a primer, and yeast pyrophosphatase in the mixture (Dean, et al. 2001). We first attempted to optimize the concentrations of several components (T7 RNA polymerase, RNase inhibitor, dNTPs, random DNA oligo, yeast pyrophosphatase, NTPs and tRNA). We found that the optimum concentrations of NTPs, tRNA, T7RNA polymerase and RNase inhibitor are in narrow ranges, indicating that these components are inhibitory to the TTcDR reaction at high concentrations. At the optimized concentrations of all components, the replication product DNA increased approximately 100-fold compared to original conditions. The kinetics of DNA replication showed a concave curve, suggesting that DNA replication accelerated due to the increasing phi29 DNA polymerase over time (Sakatani et al, 2015). This is consistent with the expected kinetics of the TTcDR reaction.

A shortcoming of this TTcDR system is the lack of recursiveness. The initial template DNA is circular, whereas the product is a linear concatemer. To make this system recursive, the linear DNA product must be circularized. We next attempted to circularize the product DNA using Cre recombinase, which has been reported to circularize the linear DNA produced by phi29 DNA polymerase (Huovinen et al, 2011), as proposed previously (Forster and Church, 2006). We added Cre recombinase into the optimized TTcDR system but found that Cre recombinase significantly inhibits the DNA replication catalyzed by phi29 DNA polymerase. This result indicates that Cre recombinase and phi29 polymerase does not work simultaneously. Therefore, to make this DNA replication system recursive, we have to add Cre recombinase after replication and remove it before the next round of replication.
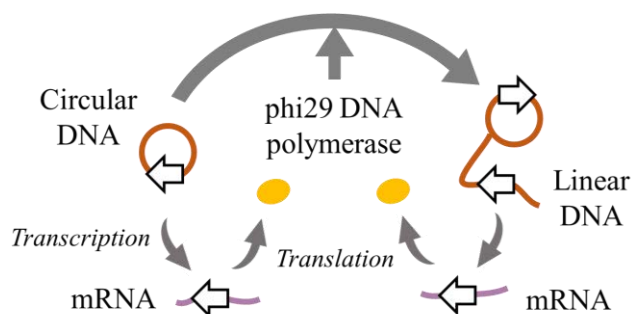
Figure 1. Schema of the TTcDR system. This system consists of a circular DNA encoding phi29 DNA polymerase gene, T7 RNA polymerase, and a reconstituted translation system. T7 RNA polymerase transcribes mRNA from the circular DNA and phi29 DNA polymerase is translated. The polymerase initiates DNA polymerization to produce a long linear DNA. The replication product is further used as a template for transcription to produce phi29 DNA polymerase.

To establish a more automatic recursive replication, which would be suitable for an artificial cell, we next attempted to develop a Cre-resistant phi29 polymerase by using evolutionary engineering. We first established a cycle to repeat the TTcDR reaction in the presence of Cre recombinase (Fig. 2). This cycle consisted of four stages. (i) Encapsulation: the TTcDR system was encapsulated in a water-in-oil emulsion with Cre recombinase. A water droplet was expected to contain less than one DNA molecule. (ii) DNA replication: the emulsion was incubated for the TTcDR reaction. (iii) DNA recovery: the droplets were collected and the product linear DNA was amplified by PCR. (iv) Circularization: after converting the both ends of the DNA into sticky ends, the liner DNA was circularized by ligation. The new circular DNA was then re-encapsulated into the emulsion for the next round of the replication cycle. The product DNA concentration was measured using quantitative PCR. In this system, mutations are introduced into the DNA through polymerization error. If a mutant DNA that encodes Cre-resistant phi29 polymerase appears, it should dominate the population. We repeated this cycle for 40 times and found that the average replication ability of the DNA population increased gradually even in the presence of increasing amount of Cre recombinase. This indicated that more Cre-resistant mutant DNAs were obtained.

In this study, we constructed the TTcDR system and obtained mutants of the DNA genome, the replication of which is tolerant to Cre recombinase. These results provide a step toward the *in vitro* construction of an artificial cell containing a recursive replication system of a DNA genome.

References

Dean, F. B., Nelson, J. R., Giesler, T. L., and Lasken, R. S. (2001). Rapid amplification of plasmid and phage DNA using Phi 29 DNA polymerase and multiply-primed rolling circle amplification. *Genome Research*, 11:1095–1099.

Forster, A. C. and Church, G. M. (2006). Towards synthesis of a minimal cell. *Molecular Systems Biology*, 2:45.

Huovinen, T., Julin, M., Sanmark, H., Lamminmäki, U. (2011) Enhanced error-prone RCA mutagenesis by concatemer resolution. *Plasmid*, 66:47–51.

Ichihashi, N., Usui, K., Kazuta, Y., Sunami, T., Matsuura, T., and Yomo, T. (2013). Darwinian evolution in a translation-coupled RNA replication system within a cell-like compartment. *Nature Communications*, 4:2494.

Sakatani, Y., Ichihashi N., Kazuta, Y., and Yomo, T. (2015) A transcription and translation-coupled DNA replication system using rolling-circle replication, *Scientific Reports* 5:10404.

Shimizu, Y., Inoue, A., Tomari, Y., Suzuki, T., Yokogawa, T., Nishikawa, K., and Ueda, T. (2001). Cell-free translation reconstituted with purified components. *Nature Biotechnology*, 19:751–755.

Stano, P. and Luisi, L. (2013). Semi-symthetic minimal cells: origin and recent developments. *Current Opinion in Biotechnology*, 24:633–638.

Figure 2. Cycle of the TTcDR for evolution experiment. (i) Encapsulation: the TTcDR system is encapsulated in a water-in-oil emulsion with Cre recombinase. (ii) DNA replication: incubation for the TTcDR reaction. (iii) DNA recovery: the droplets are collected and the product DNA was amplified. (iv) Circularization: the product linear DNA is circularized by ligation.

Critical Mutation Rate has an Exponential Dependence on Population Size for Eukaryotic-Length Genomes

Elizabeth Aston[1], Alastair Channon[1], Roman V. Belavkin[2], Rok Krasovec[3] and Christopher G. Knight[3]

[1] School of Computing and Mathematics, Keele University, ST5 5BG, UK
[2] School of Engineering and Information Sciences, Middlesex University, London, NW4 4BT, UK
[3] Faculty of Life Sciences, The University of Manchester, M13 9PT, UK
{e.j.aston, a.d.channon}@keele.ac.uk, r.belavkin@mdx.ac.uk, {rok.krasovec, chris.knight}@manchester.ac.uk

Abstract

The critical mutation rate (CMR) determines the shift between survival-of-the-fittest and the survival of individuals with greater mutational robustness (the "flattest"). Small populations are more likely to exceed the CMR and become less well adapted; understanding the CMR is crucial to understanding the potential fate of small populations under threat of extinction. Here we present a simulation model capable of utilising input parameter values within a biologically relevant range. A previous study identified an exponential fall in CMR with decreasing population size, but the parameters and output were not directly relevant outside artificial systems. The first key contribution of this study is the identification of an inverse relationship between CMR and gene length when the gene length is comparable to that found in biological populations. The exponential relationship is maintained, and the CMR is lowered to between two to five orders of magnitude above existing estimates of per base mutation rate for a variety of organisms. The second key contribution of the study is the identification of an inverse relationship between CMR and the number of genes. Using a gene number in the range for *Arabidopsis thaliana* produces a CMR close to its known mutation rate; per base mutation rates for other organisms are also within one order of magnitude. This is the third key contribution of the study as it represents the first time such a simulation model has used input and produced output both within range for a given biological organism. This novel convergence of CMR model with biological reality is of particular relevance to populations undergoing a bottleneck, under stress, and subsequent conservation strategy for populations on the brink of extinction.

Introduction

Fitter genotypes can be outcompeted by genotypes with greater robustness when the mutation rate exceeds a critical mutation rate (CMR); in terms of fitness landscapes, narrow high fitness peaks may be lost, while broader, lower peaks are maintained by a population of reproducing sequences. This so called "survival-of-the-flattest" has been observed in *in silico* evolving systems (Wilke, 2005). CMR has an exponential dependence on population size in both haploid (Channon et al., 2011) and diploid populations (Aston et al., 2013); as population size falls, the CMR above which fitter alleles are lost transitions unexpectedly from near-constant (the previous assumption in evolutionary biology) to drop exponentially for small populations. It has been verified that this model closely reproduces the established mathematical relationship between population size and "error threshold" (ET. No mathematical model has yet been derived for the CMR) (Aston et al., 2013). It is therefore possible that CMRs in small populations could be within the range of biological mutation rates. However, biological organisms typically have lengths and numbers of genes orders of magnitude higher than those used in models of ETs or CMRs, so how relevant such models are to real biological populations remains an open question.

From Artificial to Biological Evolution: Mutation of Genes in Nature

To bridge the gap between artificial and biological evolution it is paramount that, when implemented as a simulation, a model can be given parameter values within the range observed in biological organisms and subsequently output biologically realistic results. The models defined in Aston et al. (2013) used arbitrary values for parameters such as sequence length, selected for their suitability to provide results within a small timeframe.

Whitlock et al. (2003) performed computer simulations to investigate the effects of varying the strength of selection and mutational effects among dimensions. They used a model based on Fisher's model of the geometry of adaptation (Fisher, 1930), but used a hyperellipse in which the strength of selection along any axis was drawn from an exponential distribution. They concluded that changing from a hypersphere to a hyperellipse, and thus introducing dimensions with stronger selection than others, had a negligible effect on their results. It was therefore decided to focus on the parameters of mutation rate, gene length, and gene number; assuming equal strength of selection is not expected to affect the credibility of the results.

Mutation Rates

The mutation rate used in the simulation model is analagous to the biological per base mutation rate (see Table 1). Bac-
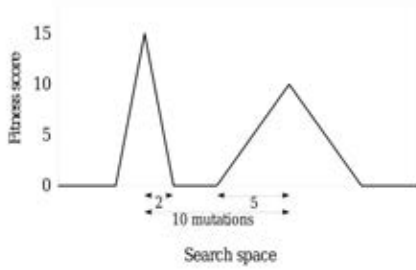
Figure 1: **Two-peak fitness landscape**, with one narrow peak of high fitness (Peak 0), and one broader peak of lower fitness (Peak 1). The fitness score is relative, and the width and distance between the peaks are given in terms of Hamming distance. Diagram adapted from Wilke (2005).

terial species were not included due to their use of lateral gene transfer which is not currently included in the simulation model. Viruses, which are known to live very close to the ET (Eigen and Schuster, 1979), were not included due to the complexity and variety of reproduction techniques which include incorporation into a host genome. Nachman and Crowell (2000) obtained an estimate of the average mutation rate per nucleotide by comparing pseudogenes (genes that do not code for proteins or are never expressed) in humans and chimpanzees. Baer et al. (2007) brought together the results of theoretical and empirical studies to list mutation rate estimates in a number of multicellular eukaryotes. Drake et al. (1998) list mutation rate estimates from studies using mutation accumulation and radiation experiments. Lynch (2010a) also lists mutation rates from various sources. Xue et al. (2009) obtained an estimate for the base substitution rate in the human Y chromosome through direct sequencing. Kumar and Subramanian (2002) conducted a computational analysis of 5669 genes from species of placental mammals. Keightley et al. (2009) did whole-genome shotgun sequencing of three mutation accumulation lines of *Drosophila melanogaster*, while Keightley et al. (2014) sequenced two parents and 12 offspring. Denver et al. (2004) provide a direct estimate of the mutation rate from a set of *Caenorhabditis elegans* mutation accumulation lines. Haag-Liautard et al. (2008) and Ossowski et al. (2010) provide estimates using mutation accumulation lines. Durbin et al. (2010) examine variation in the sequence of the human genome. Lynch et al. (2008) provide a mutation rate estimate from complete genome sequencing of *Sacccharomyces cerevisiae*. Lynch (2010b) used existing data to estimate the mutation rate of various eukaryotes.

Genetic Sequences

Derelle et al. (2006), Sharma et al. (2005) and Lewin (2008) list the length of various genes for various biological organisms at between approximately 1000 to 140,000 bp; the sequence length of 30 bp used to produce the results in As-

ton et al. (2013) is small when compared with the length of genes found in a wide range of natural species.

Aston et al. (2013) used a two-peak landscape, with the height of peak 0 constant at 15 and the radius 2, the height of peak 1 constant at 10 and the radius 5, and the Hamming distance between the peaks set at 10 (Figure 1). If each peak in the two-peak landscape is considered to be a different set of alleles (variant of a gene), estimates of genetic distances between alleles for various genes can be seen to be analogous to the distance between the peaks. They fall within the range of 1 and 56 polymorphisms (Bryan et al., 2000; Ramkumar et al., 2010). Similarly, the number of polymorphisms was estimated to be at most 13 (including non-coding regions) within various human genes studied by Cargill et al. (1999). In both cases the value of 10 used for the distance between peaks in Aston et al. (2013) is close to the range of numbers listed therefore it was decided to keep this number constant. Varying the distance between the peaks may be an interesting future study.

Longer sequences means more bases to potentially mutate each generation, leading to the formation of three hypotheses.

Hypothesis 1

According to the drift-barrier hypothesis, drift prevails over selection to determine mutation if the magnitude of the selection coefficient s is less than $1/Ne$ (where Ne is effective population size). The strength of selection that reduces mutation rate through mutation-selection balance is countered by Ne-dependent genetic drift (Sniegowski and Raynes, 2013). Following this population size dependence, we hypothesise that, for varying gene lengths and numbers, the CMR will vary with population size, and that this will occur in line with the exponential model identified in Aston et al. (2013).

Hypothesis 2

Drake summarized all studies up to 1990 and concluded that the per nucleotide per generation mutation rate u varies inversely with genome size G in microbes (Drake, 1991; Sung et al., 2012). Eigen and Schuster (1979) theoretically determined the ET in terms of selection pressure and sequence length. Using this model, Ochoa et al. (2000) and Ochoa (2006) found that longer sequence lengths lead to lower ETs in genetic algorithms, defined by the equation $p = \frac{ln(\sigma)}{L}$ where p is the ET on a single peak landscape, L is sequence length, and σ is selection strength which is kept constant. Nowak (1992) theoretically determined the ET in terms of the relative fitness of mutant and wild type (a_1 and a_2 respectively, where a_2 is assumed to have the lower fitness) and the sequence length (m). Giving the ET as $1 - q_{crit} = \left(\frac{a_1}{a_2}\right)^{\frac{1}{m}}$, it can be seen that increasing m will decrease the ET. In accordance with this and with Drake

173

(1991), it is expected that increasing the sequence length will also lower the CMR.

Hypothesis 3

We hypothesise that increasing the number of genes (while keeping gene length constant) will further lower the CMR as it will increase the overall sequence length. Gene numbers within biological ranges are expected to lead to CMRs close to the range of mutation rates observed for biological species; it is expected biological organisms will be evolving close to the mutation rate that results in the greatest levels of adaptation.

Simulation Model

The system used a two-peak fitness landscape (Figure 1), with the height of peak 0 constant at 15 and the radius 2, the height of peak 1 constant at 10 and the radius 5, and the Hamming distance between the peaks set at 10 as per Aston et al. (2013). Each individual consisted of one randomly assigned maternal and one paternal sequence of alphabet size 4, and each sequence was split into n genes of length L. Each gene had an associated target sequence of length L corresponding to peak 0 and a target sequence corresponding to peak 1. For example, if n is set to 4, there will be target sequences corresponding to peaks $0_1 1_1$, $0_2 1_2$, $0_3 1_3$, and $0_4 1_4$. For simplicity, each peak 0 was set to be all 0s and each peak 1 was randomly generated to be Hamming distance 10 away. Recombination was limited to one event per replication as it was not the focus of the study.

The dominance parameter λ was set to equal a fraction below 1.0 (0.999999999999999 specifically). This sets the relative importance of the maternal and paternal alleles while preventing either allele from drifting neutrally; if $\lambda=1.0$ the fitness of only one allele is taken into account, while the other can be anywhere in the fitness landscape. For each individual, the fitness of each of its n genes was calculated as the Hamming distance of its maternal and paternal sequences relative to each peak. The fitness values relative to peak 0 were compared with the fitness values relative to peak 1 and the highest of these selected to give a single fitness value for both the maternal and paternal sequences. The resulting maternal and paternal fitnesses were compared and subsequently designated as $f_{\max}$ and $f_{\min}$. The final relative fitness of each gene was calculated as $f = (\lambda \times f_{\max}) + ((1 - \lambda) \times f_{\min})$. The overall fitness of the individual was then taken to equal the minimum fitness out of the n genes present. The simulation was run for a range of population sizes to confirm the curves observed in previous experiments (Channon et al., 2011; Aston et al., 2013) are observed as the length and number of genes is increased.

To allow the simulation to complete within a realistic time frame, it was optimised to cease running when any one gene had lost peak 0; this was all the information required to determine the CMR, which was recorded as the mutation rate at which 95% of 2000 runs lost peak 0 within 10,000 generations for any of the possible n genes. Launching the simulation for various combinations of parameter values was also optimised to allow the mutation rate being tested for a given gene number to progress to the next mutation rate once 100 out of the possible 2000 runs (corresponding to 5%) have kept peak 0 for the duration of the simulation. Once this threshold has been exceeded, less than 95% of the 2000 runs will have lost peak 0, and the CMR will not have been reached. While this helped significantly with run time, further optimisation will be required in the future; it is currently not feasible to run the simulation for a wide range of population sizes and mutation rates for gene numbers at the upper end of the biological range.

Two approaches were taken when selecting parameter values within the biological range. Firstly, a gene length of 1000 bp was selected as a small yet biologically realistic gene length. This provided a gene length small enough to allow the simulation to run for a range of gene numbers within a realistic time frame. Secondly, based on the information in Table 1 and the value of 2232 bp mean gene size given by Derelle et al. (2006), *Arabidopsis thaliana* (thale cress) was selected as a model organism with a relatively short gene length. It is a plant native to Eurasia, with an effective population size of between 250,000 to 300,000 (Cao et al., 2011), known to contain 25,498 genes encoding proteins from 11,000 families (The Arabidopsis Genome Initiative, 2000). More current estimates of gene number are slightly higher but still within a close range for the purpose of the model (Bevan and Walsh, 2005). The simulation was run for population size 10 with a gene length of 1000 (as per the previous runs) or 2000 (range of *A.thaliana*), but with 25,000 genes to bring the gene number into the range of *A.thaliana*.

Results

Increasing the gene length decreses the CMR in

line with the exponential model

Figure 2 shows the CMR for two of the sequence lengths studied; increasing sequence length decreases the CMR in a single-gene-per-individual diploid *in silico* evolving system modelled on the biological process of meiosis, while maintaining the exponential relationship with population size presented in Aston et al. (2013). *A.thaliana* has a gene length of 2232 bp (Derelle et al., 2006), the average gene length in humans is 27 kbp (Lewin, 2008), the upper bound for the usual gene length range for flies and mammals is 100 kbp (Lewin, 2008), and the longest gene in the collagen family is 132.83 kbp (Sharma et al., 2005); the length of genes present in biological species varies greatly. Table 1 was used

Table 1: **Mutation rates for various eukaryotic species.** Mutation rate estimates are specified as the number of times a single base will mutate spontaneously. If a timeframe (per generation, per cell division) is specified this is listed in the *Unit* column. * refers to mutation rates used for reference in Figure 3.

Species	Genome size (Mbp)	Mutation rate	Base/genome	Unit	Source
Human	3080	1.00E-08 - 2.50E-08 *	Per base	Per generation	Nachman and Crowell (2000); Durbin et al. (2010); Lynch (2010a)
Human	3080	1.75E+02	Per genome	Per generation	Nachman and Crowell (2000)
Human	3080	5.00E-11 - 6.00E-02	Per base	Per cell division	Drake et al. (1998); Lynch (2010a)
Human	3080	1.60E-01	Per genome	Per cell division	Drake et al. (1998)
Human (Y chromosome)	58	3.00E-08	Per base	Per generation	Xue et al. (2009)
Human, chimpanzee	3080	3.00E+00	Per genome	Per generation	Baer et al. (2007)
Drosophila melanogaster	120	4.65E-09 - 6.20E-08	Per base	Per generation	Haag-Liautard et al. (2008); Keightley et al. (2009); Lynch (2010a); Keightley et al. (2014)
Drosophila melanogaster	120	9.90E-01 - 1.20E+00	Per genome	Per generation	Baer et al. (2007); Haag-Liautard et al. (2008)
Drosophila spp.	120	7.00E-02	Per genome	Per generation	Baer et al. (2007)
Drosophila melanogaster	120	1.30E-10 - 3.40E-10	Per base	Per cell division	Drake et al. (1998); Lynch (2010a)
Quail, chicken	1050	4.90E-01	Per genome	Per generation	Baer et al. (2007)
Sheep, cow	2870	9.00E-01	Per genome	Per generation	Baer et al. (2007)
Old World Monkey		1.90E+00	Per genome	Per generation	Baer et al. (2007)
Mouse, rat	2640	9.10E-01	Per genome	Per generation	Baer et al. (2007)
Mouse	2640	1.80E-10	Per base	Per cell division	Drake et al. (1998)
Mouse	2640	1.10E-08	Per base	Per generation	Drake et al. (1998)
Saccharomyces cerevisiae	12.1	3.30E-10	Per base	Per generation	Lynch (2010a)
Saccharomyces cerevisiae	12.1	3.30E-10	Per base	Per cell division	Lynch et al. (2008)
Average mammalian		2.20E-09 *	Per base	Per genome/year	Kumar and Subramanian (2002)
Mammalian upper bound		2.61E-09	Per base	Per genome/year	Kumar and Subramanian (2002)
Caenorhabditis elegans	100	8.40E-09 - 2.10E-08	Per base	Per generation	Denver et al. (2004); Haag-Liautard et al. (2008); Lynch (2010b)
Caenorhabditis elegans	100	2.90E+00	Per genome	Per generation	Lynch et al. (2008)
Arabidopsis thaliana	157	7.10E-09 *	Per base	Per generation	Ossowski et al. (2010)
Arabidopsis thaliana	157	6.50E-09	Per base	Per generation	Lynch (2010b)

to identify a known mutation rate for *A.thaliana*, the average mammal, and humans as 7.1×10^{-9}, 2.2×10^{-9}, and 2.5×10^{-8} respectively (per base, per generation). Figure 3 shows each of these mutation rates plotted against their respective gene lengths, along with the maximal CMR produced when the simulation was run with sequence lengths of 2000, 27000, 100000, and 150000. The maximal CMR represents the value at which each curve has levelled out (e.g., Figure 2), applicable to the range of population sizes normally expected for each species without threat of extinction (where population size refers to a local population rather than the total number of individuals globally). It was taken to be the CMR at population size 1000. Note the log scale used for the mutation rate as this enables the difference between the curves and the biological mutation rates to be seen clearly.

While Figure 3 is promising in that none of the biological mutation rates are higher than the respective CMRs produced by the simulation, both sets of mutation rates are between two and five orders of magnitude from each other.

Increasing the number of genes produces

biologically realistic CMRs

As increasing gene length has been seen to decrease CMR, increasing the number of genes was also expected to decrease CMR. The simulation model was run with a minimal yet biologically realistic gene length of 1000, with gene

number doubling from *n*=1 up to *n*=8192. The CMR was recorded as the mutation rate at which 95% of 2000 runs lost peak 0 for any of the possible *n* genes. Figure 4 shows the CMR decreases by up to three magnitudes as gene number increases from 1 to 8192, bringing the CMR to within an order of magnitude of the biological mutation rates listed in Table 1. Curve fitting using R showed the results follow quadratic curves; these can be seen to become closer as population size is increased, indicating the decrease in the rate of change of CMR with increasing population size seen in Figure 2.

Population size 10 was also run with 25,000 genes of length 1000 or 2000 to bring the gene number to within the correct range for *A.thaliana* (The Arabidopsis Genome Initiative, 2000; Bevan and Walsh, 2005). Increasing the gene length decreased the CMR further to within an order of magnitude of the per base per generation mutation rate for *A.thaliana* which is given as 7.1×10^{-9} (Table 1). Figure 4 shows per base mutation rate estimates for *A.thaliana*, *C.elegans*, and *D.melanogaster* taken from Table 1, each of which are within an order of magnitude of the simulation results for 25,000 genes for population size 10. It is notable that the genome size estimates for multicellular eukaryotes used in Figure 4 are based on numbers of protein coding genes. Protein coding sequences account for a relatively small proportion of the total genome length in such organisms (1.2% in humans (Consortium, 2012)), but much more of the sequence is functional at some level, probably at least
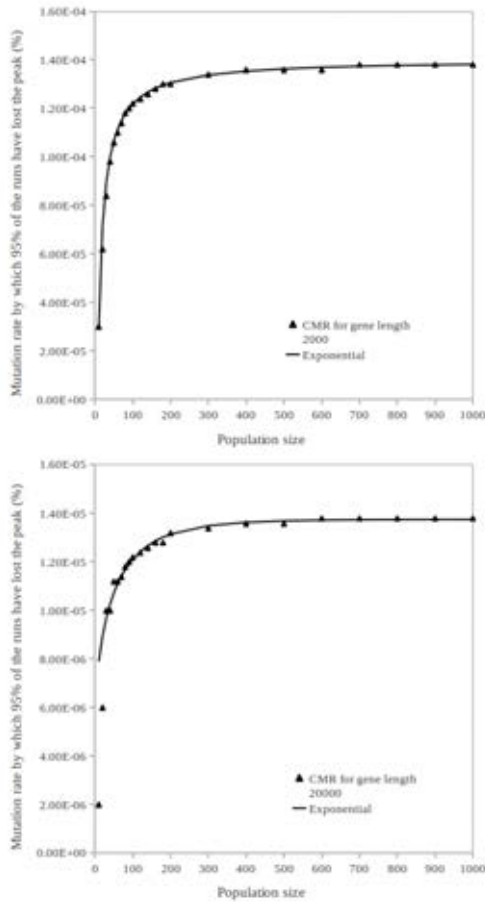
Figure 2: **CMR when the GA was run for one gene with a sequence length of 2000 and 20000.** The exponential line was obtained using the equation $y = A - B * e^{-((N/C)^D)}$ (with N being population size), and the parameters determined by curve-fitting using R with a least squares method.

9% (Ward and Kellis, 2012), with estimates of up to 80% in humans (Consortium, 2012; Lu et al., 2015) (albeit this last figure is likely to be a substantial over-estimate (Graur et al., 2013)). This means that the genome size at which these biological mutation rates are plotted in Figure 4 is a minimal estimate, the true value being substantially, perhaps an order of magnitude, higher, therefore putting their observed mutation rates closer to the CMRs estimated by simulation.

Discussion

Aston et al. (2013) showed that population size influences the CMR that can be tolerated before fitter individuals are outcompeted by those that have a greater mutational robustness in both haploid and diploid artificial populations, a result which has now been demonstrated to have relevance beyond artificial systems. Gene lengths given in Derelle et al. (2006), Sharma et al. (2005) and Lewin (2008) show that the
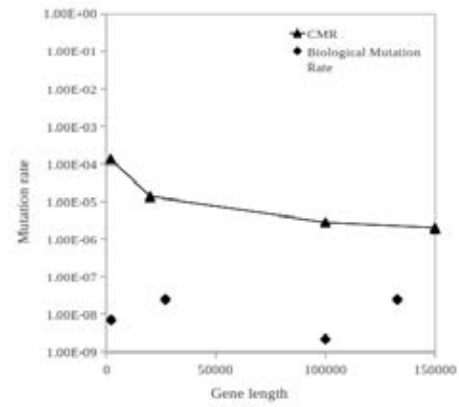
Figure 3: **Maximal CMR plotted alongside biological per base per generation mutation rates for one gene with varying sequence lengths.** Maximal CMR is the CMR recorded for population size 1000 in the simulation, representing the point at which the curve for each gene length has levelled out (e.g., Figure 2). The biological mutation rates were taken from Table 1 for eukaryotic species with comparable gene lengths to the sequence lengths used in the simulation.

sequence length of 30 used in Aston et al. (2013) is significantly smaller than the length of genes observed in biological species. The mutation rates in Table 1 are also many orders of magnitude lower than the CMR reported in Aston et al. (2013). Hypothesis 1 stated that the CMR will always have an exponential dependence on population size, while hypothesis 2 stated increasing the sequence length will lower the CMR; Figure 2 supports these hypotheses as it shows that increasing the gene length by a factor of 10 decreases the CMR by a factor of 10, with each gene length resulting in an exponential fall in CMR with decreasing population size. A change in order of magnitude can also be seen in the biological values. For example, the mean gene length of *A.thaliana* is given as 2232 bp (Derelle et al., 2006), while the average gene length of humans is just over 10 times longer at 27 kbp (Lewin, 2008). In Table 1, the per base per generation mutation rate for *A.thaliana* is given as 7.1×10^{-9}, while the per base per generation mutation rate for humans is an order of magnitude higher at 2.5×10^{-8}. Figure 3 shows that, when compared with mutation rates for biological species with comparable gene lengths, the CMR is always higher as expected. This is a key contribution of the study as it indicates the CMR exhibits a comparable relationship with gene length as previously determined for the ET (Nowak, 1992; Ochoa et al., 2000; Ochoa, 2006); the consistency between the known results for the ET and our new results for the CMR increases confidence in the study.

While it is clear that increasing sequence length decreases the CMR, the CMR remains between two and four orders of magnitude higher than the biological mutation rates (Figure 3). Hypothesis 3 stated that increasing the number of genes
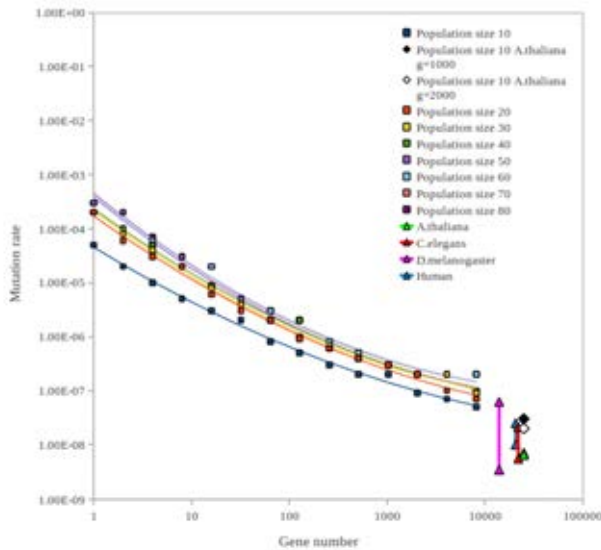
Figure 4: **CMR plotted alongside gene number for varying population sizes.** Data are shown for population sizes 10 to 80 with results plotted on a log log scale. Gene length was kept constant at 1000, while gene number was doubled from 1 up to 8192. The corresponding quadratic lines were obtained by curve-fitting using R (specifically Qmod <- lm(log(ydata)~ log(xdata) + I(log(xdata)^2))). Population sizes shown represent the steep part of the curve in Figure 2 before it levels out. Population size 10 was also run with 25,000 genes, correct range for *A.thaliana*. Gene length was set to 1000 to match the other runs or 2000 to bring it closer to *A.thaliana*'s gene length. For reference, the range of per base mutation rates from Table 1 is shown for *A.thaliana*, *C.elegans* (nematode worm), *D.melanogaster* (fruit fly) and humans (with gene number estimates from The Arabidopsis Genome Initiative, (2000), Nam and Bartel (2012), Ashburner and Bergman (2005), and Consortium (2012) respectively). The mean gene size of *A.thaliana* is 2232 bp (Derelle et al., 2006), the median gene length for *C.elegans* is ~1700 b (Cutter et al., 2009), the average gene length for *D.melanogaster* is 1130 b, and for humans 27 kb (Lewin, 2008).

will lower the CMR. Gene numbers within biological ranges were expected to lead to CMRs close to the range of mutation rates observed for biological species. Consistent with this, Figure 4 demonstrates that when gene length is kept constant, doubling the number of genes leads to a reduction in the CMR at which 95% of runs lose peak 0 for at least one gene. The magnitude of this reduction is variable, but occurs across all population sizes shown in Figure 4. The population sizes shown represent the steepest part of the curve in Figure 2 before it levels out.

It is expected that biological organisms have evolved to mutate below the CMR; mutation in loss of function alleles will have less of an impact on fitness compared with the same level of mutation in a functional allele. This means peaks of lower fitness and greater mutational robustness can be expected to exist in real life fitness landscapes (independent of the potential effect of epistasis). Real biological organisms therefore have the potential to lose higher fitness peaks at the CMR. There is a lower limit on mutation rate as defined by the drift-barrier hypothesis therefore it is expected biological mutation rates will exist somewhere between this lower limit and the CMR. At some point(s) in parameter space biological mutation rates and CMRs will come close; it is expected mutation rates will be just below the CMR in at least some cases.

Figure 4 shows a drop in CMR in the order of three magnitudes as gene number increases from 1 to 8192. This is a key contribution of the study as it brings the CMR to within an order of magnitude of the biological mutation rates listed in Table 1. The decreasing CMR shown in Figure 4 indicated that increasing the gene number further would bring the CMR directly into the range of biological mutation rates. To test this, population size 10 was run with 25,000 genes of length 1000 or 2000 to bring the gene number and length to within the correct range for *A.thaliana*. This decreased the CMR further to within an order of magnitude of the per base per generation mutation rate for *A.thaliana* (Table 1). Figure 4 also shows per base mutation rate estimates for *C.elegans*, humans, and *D.melanogaster* taken from Table 1, all of which are also within an order of magnitude of the simulation results for 25,000 genes. The mutation rates for *A.thaliana* and *C.elegans* are at or below the predicted CMR while *D.melanogaster* is slightly higher but likely to be below the predicted CMR for a population size greater than 10 based on the trend in Figure 4. This is an important contribution; it is a demonstration that, in a system in which an individual's fitness is dependent on the minimum fitness of its *n* constituent genes, it is possible to input biologically realistic parameter values for a specific organism into the simulation model and produce a CMR within the range of current biological estimates of mutation rate for that organism.

Bringing the CMR into the biological range is a very important step in the development of an *in silico* model to directly model the evolution of biological species. Future work will require further optimisation of the simulation model to increase run time feasibility. The current study had a high level of neutrality due to the small width of the peaks relative to the size of the adapting sequences. Varying the width of the peaks and distance between them provides a potential future study into the effects of neutrality on the CMR. It should also be noted that eukaryotic organisms such as those discussed here have their DNA organised into chromosomes, for example, the five chromosomes of *A.thaliana* (The Arabidopsis Genome Initiative, 2000). This gap in the current model presents a potential for further development of the model and future study of the effect

of recombination on the CMR. Prediction of the CMR for populations of varying sizes will enable identification of the optimum mutation rate, a crucial parameter in the evolution of small populations where CMR is known to vary significantly (Aston et al., 2013); this has the potential to influence understanding of populations undergoing a bottleneck, under stress, and subsequent conservation strategy for populations on the brink of extinction.

References

Ashburner, M. and Bergman, C. M. (2005). Drosophila melanogaster: A case study of a model genomic sequence and its consequences. *Genome Research*, 15:1661–1667.

Aston, E., Channon, A., Day, C., and Knight, C. G. (2013). Critical mutation rate has an exponential dependence on population size in haploid and diploid populations. *PLoS ONE*, 8(12):e83438.

Baer, C. F., Miyamoto, M. M., and Denver, D. R. (2007). Mutation rate variation in multicellular eukaryotes: causes and consequences. *Nature Reviews Genetics*, 8:619.

Bevan, M. and Walsh, S. (2005). The Arabidopsis genome: A foundation for plant research. *Genome Research*, 15:1632–1642.

Bryan, G. T., Wu, K., Farrall, L., Jia, Y., Hershey, H. P., McAdams, S. A., Faulk, K. N., Donaldson, G. K., Tarchini, R., and Valent, B. (2000). A single amino acid difference distinguishes resistant and susceptible alleles of the rice blast resistance gene pi-ta. *The Plant Cell*, 12:2033–2045.

Cao, J., Schneeberger, K., Ossowski, S., Günther, T., Bender, S., Fitz, J., Koenig, D., Lanz, C., Stegle, O., Lippert, C., Wang, X., Ott, F., MÜller, J., Alonso-Blanco, C., Borgwardt, K., Schmid, K. J., and Weigel, D. (2011). Whole-genome sequencing of multiple Arabidopsis thaliana populations. *Nature Genetics*, 43(10):956–963.

Cargill, M., Altshuler, D., Ireland, J., Sklar, P., Ardlie, K., Patil, N., Lane, C. R., Lim, E. P., Kalyanaraman, N., Nemesh, J., Ziaugra, L., Friedland, L., Rolfe, A., Warrington, J., Lipshutz, R., Daley, G. Q., and Lander, E. S.

(1999). Characterization of single-nucleotide polymorphisms in coding regions of human genes. *Nature Genetics*, 22:231–238.

Channon, A., Aston, E., Day, C., Belavkin, R. V., and Knight, C. G. (2011). Critical mutation rate has an exponential dependence on population size. In *Advances in Artificial Life, ECAL 2011: Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems*.

Consortium, E. P. (2012). An integrated encyclopedia of DNA elements in the human genome. *Nature*, 489(7414):57–74.

Cutter, A. D., Day, A., and Murray, R. L. (2009). Evolution of the Caenorhabditis elegans genome. *Molecular Biology and Evolution*, 26(6):1199–1234.

Denver, D. R., Morris, K., Lynch, M., and Thomas, W. K. (2004). High mutation rate and predominance of insertions in the caenorhabditis elegans nuclear genome. *Nature*, 430:679–682.

Derelle, E., Ferraz, C., Rombauts, S., Rouzé, P., Worden, A. Z., Robbens, S., Partensky, F., Degroeve, S., Echeynié, S., Cooke, R., Saeys, Y., Wuyts, J., Jabbari, K., Bowler, C., Panaud, O., Piégu, B., Ball, S. G., Ral, J. P., Bouget, F. Y., Piganeau, G., De Baets, B., Picard, A., Delseny, M., Demaille, J., Van de Peer, Y., and Moreau, H. (2006). Genome analysis of the smallest free-living eukaryote ostreococcus tauri unveils many unique features. *Proceedings of the National Academy of Sciences of the United States of America*, 103(31):11647–52.

Drake, J. W. (1991). A constant rate of spontaneous mutation in DNA-based microbes. *Proceedings of the National Academy of Sciences of the United States of America*, 88:7160–7164.

Drake, J. W., Charlesworth, B., Charlesworth, D., and Crow, J. F. (1998). Rates of spontaneous mutation. *Genetics*, 148:1667–1686.

Durbin, R. M., Altshuler, D., Abecasis, G. R., Bentley, D. R., Chakravarti, A., Clark, A. G., and Collins, F. S. (2010). A map of human genome variation from population-scale sequencing. *Nature*, 467:1061–1073.

Eigen, M. and Schuster, P. (1979). *The hypercycle*. Springer, New York.

Fisher, R. A. (1930). *The genetical theory of natural selection*. Oxford University Press.

Graur, D., Zheng, Y., Price, N., Azevedo, R. B., Zufall, R. A., and Elhaik, E. (2013). On the immortality

of television sets: "function" in the human genome according to the evolution-free gospel of ENCODE. *Genome Biol Evol*, 5(3):578–90.

Haag-Liautard, C., Coffey, N., Houle, D., Lynch, M., Charlesworth, B., and Keightley, P. D. (2008). Direct estimation of the mitochondrial dna mutation rate in drosophila melanogaster. *PLoS Biology*, 6(8):e204.

Keightley, P. D., Ness, R. W., Halligan, D. L., and Haddrill, P. R. (2014). Estimation of the spontaneous mutation rate per nucleotide site in a drosophila melanogaster full-sib family. *Genetics*, 196(1):313–320.

Keightley, P. D., Trivedi, U., Thomson, M., Oliver, F., Kumar, S., and Blaxter, M. L. (2009). Analysis of the genome sequences of three drosophila melanogaster spontaneous mutation accumulation lines. *Genome Research*, 19:1195–1201.

Kumar, S. and Subramanian, S. (2002). Mutation rates in mammalian genomes. *Proceedings of the National Academy of Sciences of the United States of America*, 99(2):803–808.

Lewin, B. (2008). *Genes IX*. Jones and Bartlett Learning.

Lu, Q., Hu, Y., Sun, J., Cheng, Y., Cheung, K.-H., and Zhao, H. (2015). A statistical framework to predict functional non-coding regions in the human genome through integrated analysis of annotation data. *Scientific Reports*, 5:10576.

Lynch, M. (2010a). Evolution of the mutation rate. *Trends in Genetics*, 26:345–352.

Lynch, M. (2010b). Rate, molecular spectrum, and consequences of human mutation. *Proceedings of the National Academy of Sciences of the United States of America*, 107(37):16013–16015.

Lynch, M., Sung, W., Morris, K., Coffey, N., Landry, C. R., Dopman, E. B., Dickinson, W. J., Okamoto, K., Kulkarni, S., Hartl, D. L., and Thomas, W. K. (2008). A genome-wide view of the spectrum of spontaneous mutations in yeast. *Proceedings of the National Academy of Sciences of the United States of America*, 105(27):9272–9277.

Nachman, M. W. and Crowell, S. L. (2000). Estimate of the mutation rate per nucleotide in humans. *Genetics*, 156:297–304.

Nam, J. and Bartel, D. P. (2012). Long noncoding RNAs in c. elegans. *Genome Research*, 22:2529–2540.

Nowak, M. A. (1992). What is a quasispecies? *Trends in Ecology and Evolution*, 7:118–121.

Ochoa, G. (2006). Error thresholds in genetic algorithms. *Evolutionary Computation*, 14(2):157–182.

Ochoa, G., Harvey, I., and Buxton, H. (2000). Optimal mutation rates and selection pressure in genetic algorithms. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2000)*.

Ossowski, S., Schneeberger, K., Lucas-Lledó, J. I., Warthmann, N., Clark, R. M., Shaw, R. G., Weigel, D., and Lynch, M. (2010). The rate and molecular spectrum of spontaneous mutations in Arabidopsis thaliana. *Science*, 327:92–94.

Ramkumar, G., Biswal, A. K., Madhan Mohan, K., Sakthivel, K., Sivaranjani, A. K. P., Neeraja, C. N., Ram, T., Balachandran, S. M., Sundaram, R. M., Prasad, M. S., Viraktamath, B. C., and Madhav, M. S. (2010). Identifying novel alleles of rice blast resistance genes pikh and pita through allele mining. *International Rice Research Notes*.

Sharma, V. K., Brahmachari, S. K., and Ramachandran, S. (2005). (TG/CA)n repeats in human gene families: abundance and selective patterns of distribution according to function and gene length. *BMC Genomics*, 6:83.

Sniegowski, P. and Raynes, Y. (2013). Mutation rates: How low can you go? *Current Biology*, 23(4):R147–R149.

Sung, W., Ackerman, M. S., Miller, S. F., Doak, T. G., and Lynch, M. (2012). Drift-barrier hypothesis and mutation-rate evolution. *Proceedings of the National Academy of Sciences of the United States of America*, 109(45):18488–18492.

The Arabidopsis Genome Initiative (2000). Analysis of the genome sequence of the flowering plant Arabidopsis thaliana. *Nature*, 408:796–815.

Ward, L. D. and Kellis, M. (2012). Evidence of abundant purifying selection in humans for recently acquired regulatory functions. *Science*, 337(6102):1675–8.

Whitlock, M., Griswold, C., and Peters, A. (2003). Compensating for the meltdown: The critical effective size of a population with deleterious and compensatory mutations. *Annales Zoologici Fennici*, 40:169–183.

Wilke, C. O. (2005). Quasispecies theory in the context of population genetics. *BMC Evolutionary Biology*, 5:44.

Xue, Y., Wang, Q., Long, Q., Ng, B. L., Swerdlow, H., Burton, J., Skuce, C., Taylor, R., Abdellah, Z., Zhao, Y., Asan, MacArthur, D. G., Quail, M. A., Carter, N. P., Yang, H., and Tyler-Smith, C. (2009). Human Y chromosome base-substitution mutation rate measured by direct sequencing in a deep-rooting pedigree. *Current Biology*, 19:1453–1457.

In Silico Experimental Evolution suggests a complex intertwining of selection, robustness and drift in the evolution of genetic networks complexity

Yoram Vadée-Le-Brun and Jonathan Rouzaud-Cornabas and Guillaume Beslon

Universitée de Lyon, CNRS, INRIA Beagle team,
INSA-Lyon, LIRIS, UMR5205, F-69621, France
guillaume.beslon@inria.fr

Abstract

Using the RAevol model we investigate whether the molecular complexity of evolving organisms is linked to the "complexity" of their environment. Here, the complexity is considered as the number of different states environments can have. Results strikingly show that the number of genes acquired by an organism during its evolution does not increase when the number of states of the environment increases but that the connectivity of their genetic regulation network actually does. On the opposite, we show that the mutation rate has an important influence on the gene content. We interpret these results as a complex intertwining of direct selective pressures (the more genes, the better the organisms can be) and robustness and drift thresholds that limit the maximum number of genes at different values depending on the mutation rates.

Introduction

Since the huge diversity in terms of genome size or number of genes, even between prokaryotes, has been discovered two questions keep interesting the scientific community. On one hand, the origin of this diversity and on the other hand, whether genome size and number of genes scales with the apparent complexity of the organism.

This second question raised an interesting development. First, the tentative of linking genome size and apparent complexity for organisms was a total failure. This failure is known as the C-value paradox (Eddy, 2012). Then, the discovery of genes and non-coding DNA seemed to solve this paradox. It was proposed that it is the number of genes that scales with the apparent complexity of an organism. However, this tentative also failed leading to the "N-value paradox" (Claverie, 2001). Finally, the most recent notion of "gene regulatory network" introduced an important complexification of the genotype-to-phenotype mapping, apparently resolving the N-value paradox. Indeed, the phenotypic complexity of an individual can be explained by the number of different states its gene regulatory network can have, as shown in (Lohaus et al., 2007). Moreover, further studies of prokaryotes regulation networks have shown that the number of transcription factors scale quadratically with the total number of genes (Molina and van Nimwegen, 2009).

Strikingly, the question of the origins of the diversity of genome size and number of genes undergoes less development as it is generally assumed that the different complexity of the environment faced by an organism could account for the complexity of the genotype and of the regulation network. Indeed it is quite intuitive that the more environmental conditions an organism is likely to face, the more enzymes it needs and the more transcription factors it needs to regulate its metabolic activity (Maslov et al., 2009). This intuition is supported by experimental data where the classification of bacteria according to their lifestyle shows a correlation between the variability of their environment and their number of genes (Parter et al., 2007).

However, a correlation is not a causal effect. In particular, there are many other differences between these bacteria as their population size, mutation rate... In order to understand the causality link between the environmental complexity and the genotypic complexity, one should search for organisms of similar biology, mutation rate and population size that have evolved in environments of various complexity, an almost impossible quest. An alternative is to use *in silico* experimental evolution (ISEE) to address the question. Indeed, ISEE allows to simulate long term evolution in perfectly controlled environments with an artificial chemistry that, though abstract, is shared by all the lineages. In other words, using ISEE one can compare the fate of organisms that differ by only one single parameter, being the mutation rate, the population size or the complexity of the environment they have to cope with to survive. For instance, (Bentkowski et al., 2015) developed a model of evolution to address this question. They simulated environments with different variability and observed that in more variable environments organisms have more genes than in simpler environment. However, this model does not take into account for regulation (although a quadratic regulatory cost is assumed), leading to the direct necessity to have more genes if the environment varies more or more frequently.

In this paper, we propose to address the question of the link between environmental complexity and number of genes (and more generally other indicator of complexity)

using the RAevol model (Beslon et al., 2010a,b). Indeed, this model is able to simulate rich environmental dynamics and uses individuals with explicit genome coding for a metabolism and for a gene regulatory network that can adapt in real time this metabolism to different environmental conditions. In a preliminary study, RAevol has already been used to study the link between the mutation rate and the complexity of genomes and regulation networks: By evolving similar organisms with different mutation rates but a constant stable environment, we have been able to show that, at least in constant conditions, the mutation rate drives the complexity of the genome and the complexity of the regulation network (Beslon et al., 2010b). In particular, we have shown that, even in a constant environment, lower mutation rates lead to a larger number of genes and to a quadratic increase of the number of transcription factors, strikingly reproducing many observations by acting on only one single parameter (Knibbe et al., 2011). Yet, this preliminary study did not allow to conclude on the link between environmental complexity and genotypic complexity as it was conducted in a constant environment.

In the experiments presented here we evolved organisms in different environmental conditions to be able to test whether an increase of environmental complexity effectively leads to an increase of genomic/transcriptomic complexity. As we already have shown that the mutation rate is likely to be an important factor, we explored simultaneously three different environmental complexity and four different mutation rates. For each combination of these two parameters, we evolved organisms for 300,000 generations and measured the size of their genotype and the complexity of their regulation networks.

This paper is organized as follows: In the next section, we briefly present the RAevol platform. We then present our experimental design followed by the results of the simulations and the discussion. Finally, we conclude and describe our future research directions.

The Aevol - RAevol platform

Aevol is an *in silico* experimental evolution platform (Hindré et al., 2012; Batut et al., 2013). It was designed to study how the evolutionary conditions shape the molecular structure of an evolving organism (*e.g.*, DNA length, genes number, operonic structures...) due to direct and indirect selective pressures. In Aevol, a population of individuals evolves through a classical mutation-selection process. The specificity of Aevol lies in the genotype-to-phenotype mapping that finely models what is observed in bacteria. A circular double-stranded DNA sequence is transcribed into a set of mRNAs. These mRNAs are then parsed in search for Coding DNA Sequences (CDSs – the "genes") that are translated into proteins through an artificial genetic code. Finally, the proteins are combined to compute the individual's phenotype. We refer the reader to previously published work for

a complete description of the model and the results obtained so far (Knibbe et al., 2007, 2008; Parsons et al., 2010; Batut et al., 2013; Misevic et al., 2015).

RAevol is an extension of Aevol (Beslon et al., 2010a,b). It uses the same genome model and the same genetic code. However, in RAevol, proteins are able to act as transcription factors (TFs) beside their metabolic activity. When acting as a TF, a protein may up- or down-regulate the transcription of other genes, ultimately controlling the concentration of the proteins encoded by these genes. In other words, in RAevol, each individual owns a genetic regulation network that may dynamically modify its phenotype depending on the environmental conditions. Importantly, in RAevol, the phenotype of an organism is no longer a static function (as it is in Aevol). Rather, it becomes a dynamic function that can be evaluated (by comparing it to a target function) at different time steps during what can now be considered as the "life" of the individual. Technically, RAevol extends Aevol by adding a "transcriptional regulatory code". In RAevol, the target phenotype may change during the life of the individual, either deterministically or randomly (Figure 1). Moreover, specific proteins with no metabolic activity can be added into the individual in order to allow it to sense this variation in real time. The individual must (and can) dynamically adapt to the current target by switching between different stable states of its regulation network. The final fitness of an individual is the mean value of its fitness measured at each evaluation time step.

Experimental design

For this study, our starting point was the work done in our previous experiment with constant environments (Beslon et al., 2010b). In this previous work, population of 1000 organisms were evolved for 15,000 generations in a constant environment under 6 different mutation rates ranging from 2.10^{-4} to 5.10^{-6} mut/bp/generation. We used a similar experimental design but adapted it to test evolution in variable environments. In particular we significantly increased the length of the evolution experiment. Since evolution is likely to be much more slower in variable environments than in constant ones, we let each simulation evolve for 300,000 generations. We tested 4 mutation rates (5.10^{-4}, 1.10^{-4}, 5.10^{-5} and 5.10^{-6} mut/bp/generation).

Another difference between both experiments is that we now use a fitness proportionate selection scheme instead of the exponential-rank-based selection process used in (Beslon et al., 2010b). The fitness proportionate selection scheme is more realistic from a biological point of view since it allows evolution to switch from directional selection to purifying selection. Finally, the maximum value for the protein pleiotropy (w_{max}) has been increased from 0.03 to 0.05 to limit the number of genes and allow for faster computations (both changes reduces the maximum number of genes an organism can – or needs to – acquire during its
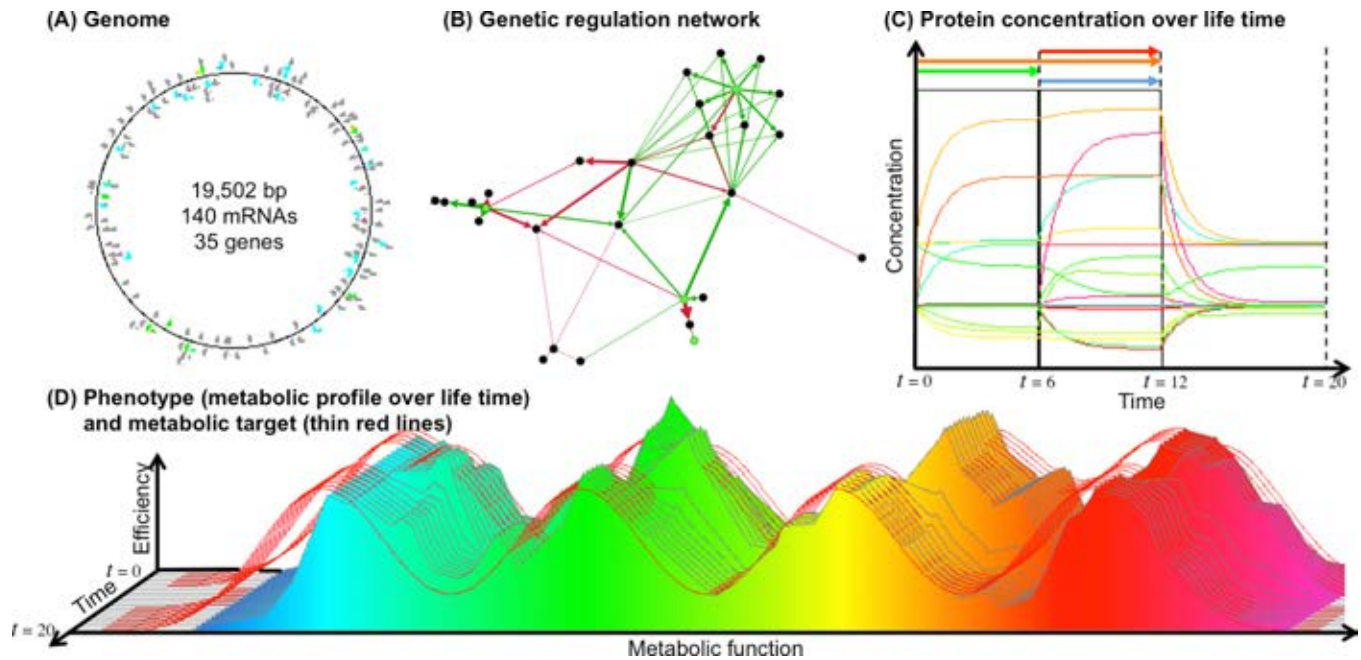
Figure 1: Example of an individual after 300,000 generations in the environment 16 (16 different conditions and 4 signals). **(A)** Genome, mRNAs and genes (colors code for the basal transcription rate. The small color dots are the genes). **(B)** Regulation network (arrows indicate links between genes and mRNAs; Red arrows are inhibiting links, green arrows are activating links. Black dots are the genes, green dots are the signaling proteins. **(C)** Protein concentrations during the life of the individual (colors indicate the function of the protein). The four signaling proteins are displayed above the graph. **(D)** Dynamic phenotype of the individual (same color code as panel C; thin red lines indicate the target at each time step). This individual evolved in the environment 16 (i.e. the four Gaussians can switch independently between their resting and their active states). Here the green Gaussians is active from $t = 0$ to $t = 6$; The orange Gaussian is active from $t = 0$ to $t = 12$ and the blue and red Gaussians are active from $t = 7$ to $t = 12$. From $t = 13$ to $t = 20$ all Gaussians are inactive. At each environmental change the regulation network (B) modifies the transcription levels, hence the protein concentrations (C), resulting in a phenotypic adaptation (D).

evolution).

Finally, in order to be able to compare the evolutionary outcome in different environments complexity, we carefully designed our dynamic environments. In a previous study with RAevol (Vadée-Le-Brun et al., 2015) we used an environment represented by 4 Gaussians which maximum values were changed randomly during the evolutionary process. We kept the same idea of having 4 Gaussians each one associated to a signaling protein informing the individuals for its variation. However, the environmental conditions were created by moving the mean of the Gaussians along the x axis: The Gaussians all have the same height (0.3) and standard deviation (0.05). At their "resting state", they are regularly spread along the x axis ($m = 0.2$, 0.4, 0.6 and 0.8). The constant environment is thus the same as in (Vadée-Le-Brun et al., 2015). But Gaussians are able to switch to an "active state" by increasing their mean value (m) of a small amount (0.05) and simultaneously sending a signaling protein to the individuals (Figure 1) such that it can trigger a change of the state of its regulation network. The lateral variation has been chosen instead of the vertical variation used so far because

it keeps the total area of each phenotypic target constant. Indeed, Aevol/RAevol are prone to a "filling artifact": Since the maximum area of a gene is bounded, the larger the area of the phenotypic target, the higher the numbers of genes individuals are likely to acquire. By using a lateral variation scheme, we kept the area of the target constant whatever the number of Gaussian that are in their "resting" or in their "active" state. Using this variation scheme and considering that the number of environmental conditions is a proxy for the environmental complexity, we are able to build a large variety of environment which complexity varies between 1 to 16 states depending on which Gaussians are allowed to be in an active state and on possible coordination in the resting-active switch of the Gaussians. Here we tested three different complexity levels: 1 (actually a constant environment), 2 (only the Gaussian 3 is allowed to be in its active state) and 16 (all 4 Gaussians switch independently between their resting and active states). For all environments but the constant one, the switch between different targets happens with 10% chance at each time step. As individuals live for 20 time steps, on average each individual faces 2 switches of phenotypic tar-
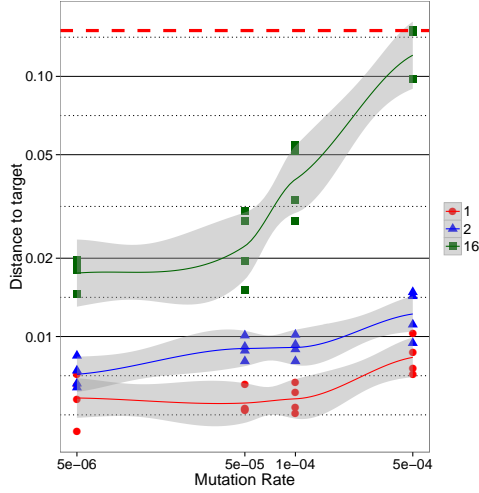
Figure 2: Mean distance to target of the best individuals from the 4 seeds between 299,000 and 300,000 generations. Both axes are in log scale. The horizontal red dashed line is the distance to target of an individual having no gene.

Figure 3: Distribution of the number of genes of each simulation versus its mutation rate in log log scale. For each dot, the corresponding value is the mean value of genes number of the best individual between 299,000 and 300,000 generations. For each environment a log log linear regression is plotted.

get during its lifetime (note that the environmental variation is synchronized for all the individuals of the population) but this number can widely vary. We label each environment by its number of phenotypic targets.

Finally, all simulations were repeated four times with different random drawing for each couple of environmental complexity and mutation rate leading to a total of $4 \times 3 \times 4 = 48$ simulations. Note that four repetitions is not high enough to obtain statistically founded comparison between the different situations but, given the structure of RAevol, we were not able to increase the number of repetitions as the amount of computational power requested to conduct the experiments started to become prohibitive[1].

Results

Among the 48 simulations, 46 resulted in well-adapted individuals that were able to properly fit the phenotypic target. The two simulations that "failed" are simulations with the highest mutation rate in the most complex environment. In this case, as Figure 2 shows it, the individuals were not able to reduce significantly the distance to the target. There genetic structure thus drifted eventually leading to too small genomes that can collapse to 0 genes.

Figure 3 shows the number of genes for each mutation rate and for each environmental complexity. It shows the same trend we observed previously (Beslon et al., 2010b): The number of genes increases as the mutation rate de-

creases. However, the number of genes is lower than what was observed in (Beslon et al., 2010b), in particular under low mutation rates. This is likely to be due to the selection scheme we use and to the larger value of w_{max}. Interestingly, the evolution in complex environments (environment "16", black squares) results in a similar linear trend (in log-log) but with a very different coefficient since the highest mutation rates lead to very low number of genes in this environment. This confirms the previous observation that in complex environments organisms are not able to evolve efficiently under high mutational pressure.

For a better readability, we plot all the following results in bar plots figure grouped by mutation rates. Figures 4 and 5 summarize the effect of the different parameter tested on the genomic structure. Figures 6 and 7 summarize their effect on the structure of the regulation network. Two opposite trends are visible in these figures. First, and quite surprisingly, the number of genes decreases when the variability of the environment increases and this effect is much stronger for the most complex environment. Furthermore this effect tends to be enhanced for high mutation rates. Second, when looking at the size of the non-coding sequences (Figure 5), the mean connectivity of the network (Figure 6) and the mean link value (Figure 7) we see an opposite trend: complex environments lead to longer non-coding sequences (when in simpler environments we only observe large non-coding sequences under a 5.10^{-6} mutation rate), higher connectiv-

---

[1]48 simulations, 300,000 generations and 1000 individuals living for 20 time steps indeed leads to a total of 288 billions of time steps at each of which we need to compute the network dynamic and the resulting phenotype of an individual.

Figure 4: Mean number of genes of best individuals from the 4 seeds between 299,000 and 300,000 generations for all mutation rates and environments. The error bars represent the standard deviation between seeds.

Figure 5: Mean size of non coding sequences in bp of best individuals from the 4 seeds between 299,000 and 300,000 generations for all mutation rates and environments. The error bars represent the standard deviation between seeds.

ity[2] and stronger links in the regulation network (except for the highest mutation rate). Finally, a more complex trend is observed when looking at the size of the transcribed sequences (Figure 8): As already shown in (Parsons et al., 2010), in "simple" environments, the higher the mutation rate, the longer the RNAs. However, for the complex environments this trend is inverted and the RNAs tend to be smaller as the mutation rate increases.

Figure 9 summarizes the two main trends observed in our experiments. It clearly shows a general tendency to complexity increases as the mutation rate decreases. Yet, the environmental complexity acts in two opposite ways: It tends to *decrease* the number of genes (Figure 9, left panel) but to *increase* the network connectivity (Figure 9, right panel).

Discussion

So as a summary of these results: the variability of the environment tends to reduce the number of genes but increases the non-coding part of the genome. This effect increases with the mutation rate. On the contrary, it tends to increase the connectivity of the gene regulatory network and the intensity of the links and to decrease the size of mRNAs (when the gene regulatory network is efficient enough). Globally, individuals in a more variable environment evolve gene regulatory networks that are smaller but more connected and with stronger links.

Such complex effects are likely to originate from the intertwining of multiple forces acting on genomic and tran-

---

[2]Note that, in RAevol, the probability for a coding sequence to be a transcription factor is quite high. Thus a 0.1 connectivity should be considered as a neutral regulation.

scriptomic complexity. Indeed, one can identify four main forces that either limit or increase the molecular complexity of the organisms:

Selective forces It is the most intuitive effect: The more complex the environment, the more genes are needed to fit the different environmental states and the more regulation is needed to switch between environmental states.

Robustness thresholds Under high mutation rates, long DNA molecules are impossible to transmit to the next generation hence limiting the maximum complexity of the genome. This effect is due to the classical error threshold that limit the length of the coding sequences (Eigen, 1971) and to a more recently identified threshold imposed on the total chromosome length by the rate of chromosomal rearrangements (Knibbe et al., 2007; Fischer et al., 2014).

Indirect selection for evolvability When evolving in continuously varying environments, selection can indirectly favor individuals that increase their rate of variation because they are able to continuously produce mutants that may follow the environmental variations (Earl and Deem, 2004).

Drift barrier The level of selection may impose a barrier to the evolution of the genomic content: If the contribution of all new genes to the fitness is quasi-neutral given the level of selection, then the selective forces vanish, thus imposing a limit to the acquisition of new genes.

We suggest that the observed effects are a complex combination of these four forces. Indeed, as we already have
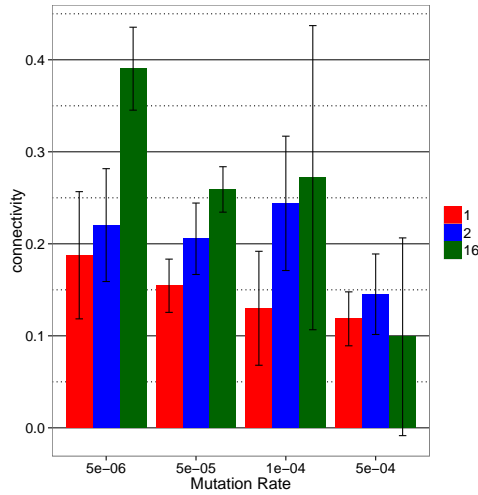
Figure 6: Mean connectivity of the gene regulatory network of best individuals from the 4 seeds between 299,000 and 300,000 generations for all mutation rates and environments. The error bars represent the standard deviation between seeds.

Figure 7: Mean intensity of links in the gene regulatory network of best individuals from the 4 seeds between 299,000 and 300,000 generations for all mutation rates and environments. The error bars represent the standard deviation between seeds.

shown it with RAevol in (Beslon et al., 2010b), the robustness threshold imposes a severe limit to the size of the genomes and to the number of genes. This explains the general trend on the number of genes observed in Figures 3, 4 and 9. Indeed, being driven by the spontaneous mutation rate, as stated by Fischer et al. (2014), this effect is likely to be independent from the selection process, hence from the complexity of the environment. Now, as also state by Fischer et al. (2014), when the genome size is far from the threshold, selective forces can fully play their role. This is indeed the case when the mutation rate is low. Interestingly, in this situation, the regulation network appears to be more connected (Figures 6, 7 and 9) suggesting that under low enough mutation rates selective forces drive the complexity of the regulation network. However this complexity is only visible in terms of component connectivity but not in terms of component number (contrary to the number of genes, the connectivity can be increased without increasing the length of the genome). The combination of these two effects can also explain the trend observed on the length of the RNAs. Indeed, as the mutation rate increases the length of the RNAs increases. As the gene size is constant, the RNA size is an indicator of operon structures that allow for a more compact genome at the cost of less regulation possibility. Interestingly, in the more variable environment, the increase of mutation rate has no impact on the RNA size, suggesting a selection for a more active regulation.

The last two surprising results are the large increase of the size of the non-coding sequences under complex environments and of course the lower number of genes observed
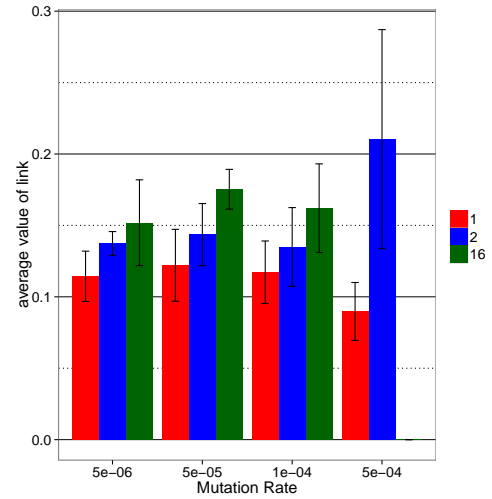
in more complex environments. Although more speculative, we could propose two hypothesis to explain these results.

First, when the complexity of the environment and the mutation rate simultaneously increase, the selective force and the robustness threshold become more and more antagonistic. Ultimately, as stated above, the robustness threshold imposes a severe limit to the complexity of the genotype, forbidding a regulation network to evolve. In highly variable environments the genomes ultimately collapse leading to a quick drop of the number of genes (Figure 4). In such a situation, the sole option for the evolution is to increase the level of variability. Since the mutation rates are fixed in our simulation, this can only be done by increasing the size of the non-coding sequences, hence increasing the dynamic of the gene repertoire. In conclusion, when the robustness threshold contradicts the selective forces, the response of evolution is an indirect selection for evolvability.

Second, we observe that the number of genes accumulated in environment 16 is always lower than what is observed in environment 2 but that the difference tends to decrease as the mutation rate decreases. We propose that, under low mutation rates, the complexity limit is no more imposed by robustness constraints but rather by the drift barrier: As the metabolic effect of a gene tends to decrease with the number of genes (as each gene will fill smaller and smaller gap between the phenotype and the phenotypic target) there must be a upper limit to the number of genes the selection can act on (indeed, Figure 2 shows that the mean distance to target saturates when the mutation rate is decreased). But the drift barrier is likely to be more stringent in more vari-
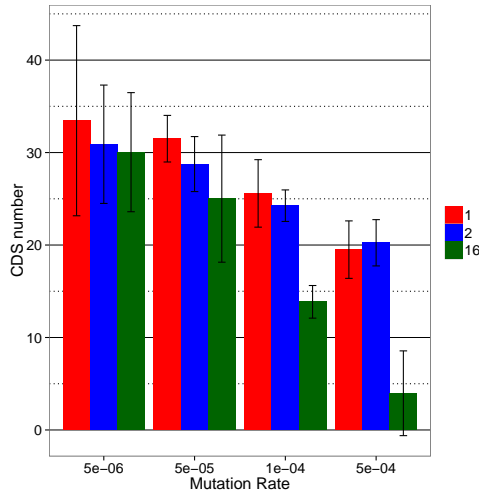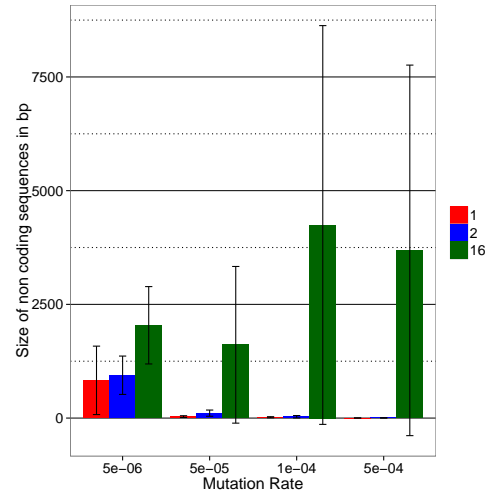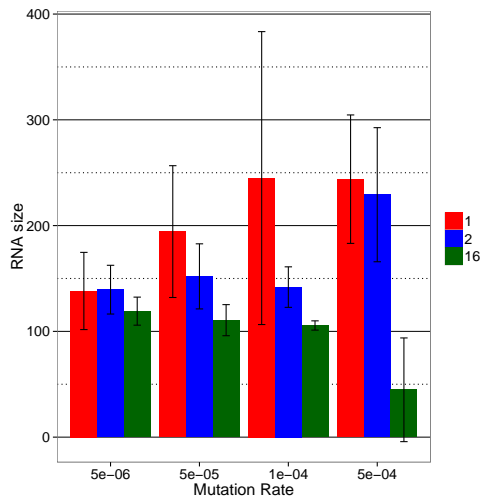
Figure 8: Mean size of mRNAs in bp of best individuals from the 4 seeds between 299,000 and 300,000 generations for all mutation rates and environments. The error bars represent the standard deviation between seeds.

able environments since the selection pressure on individual genes also depends on the fraction of time they are useful. In other words, the more variable the environment, the lower the selection pressure on the genes corresponding to the variable part of the phenotypic target the lower the number of genes the organisms can accumulate. To test this hypothesis we run simulations in environment 2 with a higher selection pressure for the two highest mutation rates (5.10^{-5} and 5.10^{-6}). In these simulations, we used a selection coefficient of 2,000 instead of 750 with again 4 repetitions each. We indeed observed an increase of the genes number. Moreover, the gain was lower for the 5.10^{-5} mutation rate (from 26 to 29 genes) than for the 5.10^{-6} mutation rate (from 31 to 43 genes), supporting the hypothesis that for a 5.10^{-5} mutation rate the number of genes is limited by both the robustness threshold and the drift barrier but that, for a mutation rate of 5.10^{-6} only the drift barrier is active.

Conclusion and perspectives

In this paper, we experimentally addressed the question of the impact of environmental variability on evolution (*i.e.*, whether more variable environments imply more complex genomes and genes regulatory networks). To evaluate this question, we used an *in-silico* experimental evolution platform, RAevol, to test three environments of increasing variability in four different mutation rates.

Our simulations confirm that the size of the gene repertoire is bounded by the mutation rate and that this also limits the complexity of the gene regulatory network. They also show that environmental variability indeed increases the connectivity and the intensity of links of the gene reg-

ulatory network but decreases its size. Moreover, environmental variability increases the genome size by increasing the non-coding part of the genome of individuals who fail to regulate their phenotype according to environmental variations. Finally, we discussed our results and proposed that the molecular complexity of an organism is a complex combination of direct selective pressure, indirect selective pressure for evolvability and robustness and drift thresholds.

The most striking results obtained here is that the gene content of our organisms does not follow the environment complexity. However this result is not so surprising if one realizes that complexity here is defined as "variability". Indeed, as G.E. Hutchinson (1957) stated, environments are complex objects composed of multiple factors (temperatures, light intensity, etc.). What we here defined as a more complex environment is not necessarily what evolution "perceives" as the highest complexity! Indeed, environmental features that don't lead to selectable traits (because of drift or robustness constraints) are simply ignored by evolution. Since drift and robustness levels are dependent on the evolutionary conditions (population size, mutation rates...) one can then argue that the complexity of an environment depends on the organism that evolves in it! Clearly, in our simulations, the *dynamic* complexity of environment 16 is more complex than the one of environment 2 and, of course, 1. But on a more global point of view, the constant environment may be considered as more complex because the target function is more strongly selected. The genetic elements managing the dynamic part of the phenotype are indeed more numerous in environment 16 but the gene content is larger in constant environments.

The main limit of these results is the low number of parameter tested and, the very low number of repetitions for each combination of parameters. We now need to validate statistically our results and hypothesis by running more repetitions and more scenarios: environments of intermediate complexity (4 and 8), lower mutation rates and higher selection strength. That objective has recently been taken a step further with the release of Aevol 5.0 that includes an optimized parallel version of the simulator.

Availability

Aevol is available under GPL license at the project website: http://www.aevol.fr. RAevol is currently in beta-version and is available upon request from the authors.

References

Batut, B., Parsons, D. P., Fischer, S., Beslon, G., and Knibbe, C. (2013). In silico experimental evolution: a tool to test evolutionary scenarios. *BMC Bioinformatics*, 14 (S15):S11.
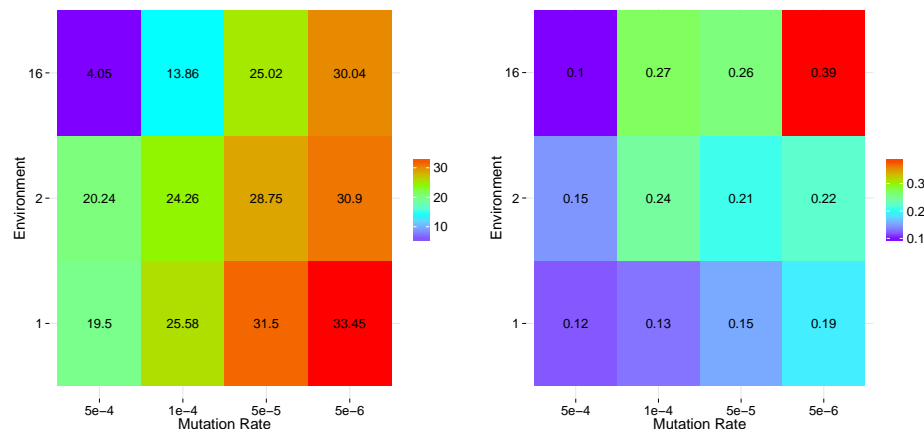
Figure 9: Left: Mean genes number for each environment and each mutation rate. Right: Mean connectivity for each environment and each mutation rate. Both heatmaps show a general Left-to-Right increase of genomic and transcriptomic complexity. However, the gene number increases when the environmental complexity is decreased while the connectivity increases when the environmental complexity increases (at least for low mutation rates).

Bentkowski, P., Van Oosterhout, C., and Mock, T. (2015). A model of genome size evolution for prokaryotes in stable and fluctuating environments. *Genome Biology and Evolution*, 7(8):2344–2351.

Beslon, G., Parsons, D. P., Peña, J.-M., Rigotti, C., and Sanchez-Dehesa, Y. (2010a). From digital genetics to knowledge discovery: Perspectives in genetic network understanding. *Intelligent Data Analysis Journal*, 14(2):173–191.

Beslon, G., Parsons, D. P., Sanchez-Dehesa, Y., Peña, J.-M., and Knibbe, C. (2010b). Scaling laws in bacterial genomes: A side-effect of selection of mutational robustness. *BioSystems*, 102(1):32–40.

Claverie, J.-M. (2001). What if there are only 30,000 human genes? *Science*, 291(5507):1255–1257.

Earl, D. J. and Deem, M. W. (2004). Evolvability is a selectable trait. *PNAS*, 101(32):11531–11536.

Eddy, S. R. (2012). The c-value paradox, junk dna and encode. *Current Biology*, 22(21):R898 – R899.

Eigen, M. (1971). Self-organization of matter and the evolution of biological macromolecules. *Naturwissenschaften*, 58(10):465–523.

Fischer, S., Bernard, S., Beslon, G., and Knibbe, C. (2014). A model for genome size evolution. *Bulletin of Mathematical Biology*, 76:2249–2291.

Hindré, T., Knibbe, C., Beslon, G., and Schneider, D. (2012). New insights into bacterial adaptation through in vivo and in silico experimental evolution. *Nature Reviews Microbiology*, 10:352–365.

Hutchinson, G. E. (1957). Cold spring harbor symposium on quantitative biology. *Concluding remarks*, 22:415–427.

Knibbe, C., Coulon, A., Mazet, O., Fayard, J.-M., and Beslon, G. (2007). A long-term evolutionary pressure on the amount of noncoding DNA. *Molecular Biology and Evolution*, 24(10):2344–2353.

Knibbe, C., Fayard, J.-M., and Beslon, G. (2008). The topology of the protein network influences the dynamics of gene order: From systems biology to a systemic understanding of evolution. *Artificial Life*, 14(1):149–156.

Knibbe, C., Parsons, D. P., and Beslon, G. (2011). Parsimonious modeling of scaling laws in genomes and transcriptomes. In *European Conference on Artificial Life (ECAL)*, pages 414–415.

Lohaus, R., Geard, N. L., Wiles, J., and Azevedo, R. B. R. (2007). A generative bias towards average complexity in artificial cell lineages. *Proceedings of the Royal Society of London B*, 274(1619):1741–1751.

Maslov, S., Krishna, S., Pang, T. Y., and Sneppen, K. (2009). Toolbox model of evolution of prokaryotic metabolic networks and their regulation. *PNAS*, 106(24):9743–9748.

Misevic, D., Frénoy, A., Lindner, A. B., and Taddei, F. (2015). Shape matters: Lifecycle of cooperative patches promotes cooperation in bulky populations. *Evolution*, 69(3):788–802.

Molina, N. and van Nimwegen, E. (2009). Scaling laws in functional genome content across prokaryotic clades and lifestyles. *Trends in Genetics*, 25(6):243–247.

Parsons, D. P., Knibbe, C., and Beslon, G. (2010). Importance of the rearrangement rates on the organization of transcription. In *Proceedings of Artificial Life XII*, pages 479–486.

Parter, M., Kashtan, N., and Alon, U. (2007). Environmental variability and modularity of bacterial metabolic networks. *BMC Evolutionary Biology*, 7(1):1–8.

Vadée-Le-Brun, Y., Rouzaud-Cornabas, J., and Beslon, G. (2015). Epigenetic inheritance speeds up evolution of artificial organisms. In *European Conference on Artificial Life*, pages 439–446.

Reductive evolution towards primitive life: What will we see?

Atsushi Shibai[1], Daisuke Motooka[2], Shota Nakamura[2] and Saburo Tsuru[1]

[1]Graduate School of Information Science and Technology, Osaka University, Yamadaoka 1-5, Suita, Osaka, Japan
[2]Research Institute for Microbial Diseases, Osaka University, Yamadaoka 1-5, Suita, Osaka, Japan
tsuru-saburo@ist.osaka-u.ac.jp

Abstract

Observing the gradual transition from "life" to "non-life" tells us a lot about the features of living systems. Based on this idea, we started simplifying natural cells by inactivating their genes randomly through experimental evolution. *Escherichia coli* (*E. coli*) cells were cultured in a high-mutagenicity environment to accumulate replication errors on their genomes. As a result, we observed dozens of mutations, which were supposed to deactivate gene expression. In addition, the gene inactivation accumulates time proportionally without growth defects so far. These results suggest that naturally isolated cells are highly redundant in an experimental environment—implying the possibility of further simplification deleting hundreds or thousands of genes.

Introduction

What is life? In order to answer this fundamental question, there have been many attempts to create primitive living systems from simpler components—with many challenges. For example, building artificial cells from chemical materials in a test tube provided experimental platforms to observe the unsophisticated cellular behaviors (Kurihara, et al. 2015). This bottom-up approach (blue arrow in Fig.1) would allow us to observe the gradual transitions from non-life to life. However, there are many technical difficulties in this approach to upgrading such highly primitive artificial cells toward complex and modern forms. On the other hand, the top-down approach (green arrow in Fig.1) can avoid such difficulties. This approach simplifies the modern cells by reducing their sophisticated genes to only retain primitive cellular functions. For example in nature, *Pelagibacter ubique*, which has smallest number of genes among free-living bacterial strains (1354 genes, Luo, 2015), is supposed to have lost thousands of genes through reductive evolution. This bacteria shows slower growth compared to other related species regardless of the amount of environmental resources—implying the loss of the sophisticated ability to respond to environmental changes. In the same way, gene loss in bacterial genomes would result in more primitive behavior in various aspects. Thus, both approaches are significant in exploring the transition between living and non-living modes of primitive cells.

Here, we followed the latter approach to obtain gradually simplified derivatives of *E. coli* by accumulating extensive, random mutations in the genome—most of which are destructive (Eyre-Walker and Keightley, 2007; Kacar and Gaucher, 2012). Accordingly, the functional genes in the genome were expected to gradually decrease from the initial 4000. In this abstract, we describe the design and progress of the experiment, and discuss future issues and prospects.
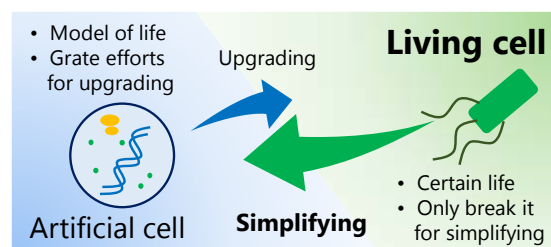
Figure 1: Simplifying natural cells has some advantages over the opposite approach for observing the transition between living and non-living systems.

Evolutionary Experiments

Ultra violet (UV) irradiation was used in increasing doses as to increase the mutation rate. Previous research showed experimental evolution with periodic UV irradiation can increase gene inactivation in the *E. coli* genome (Shibai, et al. 2014). We used the *E. coli* MDS42 strain where 15 % of its genes were manually deleted from the progenitor (Pósfai, et al. 2006). Cells were cultured in minimal medium and showed exponential growth. On the other hand, increasing dosages of UV irradiation kills cells at an exponential rate. Thereby, cell concentration was constantly measured to control the timing of UV irradiation and prevent both extinction and saturation. The cells were subcultured every four days and glycerol-stocked at the same time. The experiment was replicated six times and continued for 168 days.

Analysis of Evolutionary Changes

We conducted whole genome resequencing for the evolved lineages on the 56th and 168th days. Mutations were detected by comparing the DNA alignments with the ancestor's. Mutations that caused stop codons at abnormal positions were counted as nonsense mutations. Additionally, the mutations that caused hazardous gaps in codon reading frames were counted as missense mutations. We regarded nonsense and

missense mutations as inactive mutations. Genes with at least one inactive mutation were counted as inactivated genes, though it is still not certain whether their functions were completely lost. As a result, 19 to 94 genes were inactivated over 168 days in each lineage (Fig.2). The cell lineage with the least number of inactivated genes (Series-1) had highly aggregative growth by the 56th day already. The maximum growth rate of each lineage after the evolutionary experiment was 0.68 to 0.86 [h^{-1}]—not significantly declined from ancestor's (0.73 [h^{-1}]).
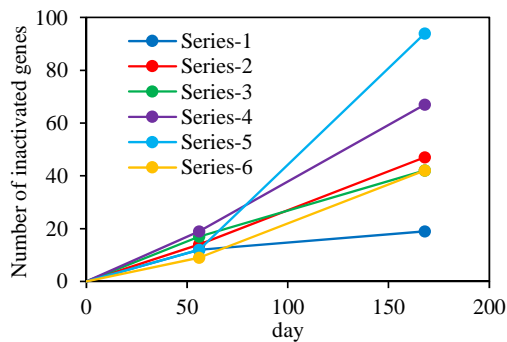
Figure 2: Number of inactivated genes during evolution

Discussion

Cells were estimated to grow more than 3000 generations over 168 days, assuming the growth rates higher than 0.6 [h^{-1}]were maintained. Dozens of genes accounting for 0.5 to 2.5% of the all genes on the ancestral genome displayed traits for accumulated inactivation. Surprisingly, even though the loss of function occurred at such a scale, a significant drop in the maximum growth rate was not observed. This implies that many of the genes impacting the growth of *E. coli* in this evolutionary experiment were redundant. One of the possible causes of this redundancy is that the natural environments were more complex than the experimental environments, so that the *E. coli* can still function with the loss of those genes obtained through natural history. To examine this, additional analysis on the function of the inactivated genes is needed.

The lineage with the least number of gene inactivation (Series-1) obtained the trait of increased aggregation of cells. We inferred that the outer cells of the aggregates protected inner cells from the harmful UV irradiation, so that the outer cells died and inner cells survived with a decreased number of inactive mutations. Such an adoptive evolution obtaining multicellular-like behavior is interesting though it does not meet the purpose of this study.

What will we see if we continue this evolutionary experiment over a longer period of time? Through first-order approximation, the number of functional genes will be about 1500 in 20 years. That is comparable to *Pelagibacter ubique*. With 10 years of further evolution, the number of functional genes would reach the level of *Mycoplasma genitalium*, which has the smallest genome even among parasites (487 genes, Choe, et al. 2016)—though the time proportional accumulation of gene inactivation would not last so long. How

would the behavior of the reductively evolved *E. coli* differ from the behavior of natural organisms with a similar number of genes? Also, as a complex adaptive system, how would its features, like energy efficiency or evolvability change? In order to answer these open questions, discussing how living systems should be understood is needed side by side with the decades-long experimental evolution.

Materials and Methods

In all the experiments of this study, cells were cultured in mM63 minimal medium at 37 °C with shaking.

Evolutionary Experiments

E. coli MDS42 was used as the ancestral strain. The bacteria were incubated in a quartz test tube. Optical density (OD) of the cells were measured every 3 min. Intensive UV irradiation was conducted when the OD value increased by 0.002 from the former irradiation. We irradiated UV from the bottom of the tube. Irradiation was applied for the dosages that killed the ancestral cells so that the survival rates were $10^{-2} \sim 10^{-3}$ at a time. Aliquot of the culture was transferred into a new tube with fresh media at 4-day intervals, so that dilution rate was 10^{-2}. The cells were glycerol-stocked at the same time.

Analysis of Evolutionary Changes

Purified genomic-DNA samples of the cells were sequenced by Illumina Miseq. Base-pair substitutions and short insertions/deletions were identified using SAMtools.

Maximum growth rates were measured as the increased rate of the OD value during the exponential growth phase in the absence of UV irradiation.

References

Kurihara, K., Okura, Y., Matsuo, M., Toyota, T., Suzuki, K. and Sugawara, T. (2015). A recursive vesicle-based model protocell with a primitive model cell cycle. *Nature Communications*, 6:8352.

Eyre-Walker, A. and Keightley, P. D. (2007). The distribution of fitness effects of new mutations. *Nature Reviews Genetics*, 8:610-618.

Kacar, B. and Gaucher, E. A. (2012). Towards the Recapitulation of Ancient History in the Laboratory: Combining Synthetic Biology with Experimental Evolution. In Adami, C., Bryson, D. M., Ofria, C., and Pennock, R. T., editors, *Artificial Life 13*, pages 11–18. MIT Press, Cambridge, MA.

Shibai, A., Tsuru, S., Ying, B., Motooka, D., Gotoh, K., Nakamura, S., and Yomo, T. (2014). Mutation accumulation in bacteria exposed to UV radiation. In Sayama, H., Rieffel, J., Risi, S., Doursat, R. and Lipson, H., editors, *Artificial Life 14*, pages 757–758. MIT Press, Cambridge, MA.

Pósfai, G., Plunkett III, G., Fehér, T., Frisch, D., Keil, G. M., Umenhoffer, K., Kolisnychenko, V., Stahl, B., Sharma, S. S., de Arruda, M., Burland, V., Harcum, S. W. and Blattner, F. R. (2006). Emergent Properties of Reduced-Genome *Escherichia coli*. *Science*, 312:1044-1046.

Luo, H. (2015). Evolutionary origin of a streamlined marine bacterioplankton lineage. *The ISME Journal*, 9:1423–1433.

Choe, D., Cho, S., Kim, S. C. and Cho, B. (2016). Minimal genome : Worthwhile or worthless efforts toward being smaller? *Biotechnology Journal*, 11:199–211.

Open-ended Evolution

Bio-Reflective Architectures for Evolutionary Innovation

Simon Hickinbotham and Susan Stepney

Department of Computer Science, University of York, UK
York Centre for Complex Systems Analysis
email: sjh518@york.ac.uk

Abstract

Computational reflection uses software architectures that are capable of self-modification at runtime. These systems have implementations between two extremes: procedural reflection, in which unlimited self-modification is available at the expense of infinite recursion; and declarative reflection, which uses pre-defined metrics to drive the self-modification and is hence limited in scope. Biological processes also exploit the concept of reflection, where natural selection drives the process of modification. The concept of a 'program' in computing has an analogy with an individual member of a species. The process of life is discretised into a series of autonomous systems, each of which creates modified versions of itself as offspring. This paper unifies the concept of computational reflection with biological systems via a new analysis of von Neumann's Universal Constructor. The result is a bio-reflective architecture that is capable of unconstrained self-modification without the problems of infinite recursion that exist in the computational counterparts. The new architecture is a blueprint for applications in Artificial Life studies, Evolutionary Algorithms, and Artificial Intelligence.

Introduction

In this paper we unify certain concepts from computational reflection (Maes, 1987; Smith, 1984) and Artificial Life (ALife). These concepts address how self-representation, autonomy and evolution contribute to 'living' systems. Each of these topics has aspects that are represented in the idea of *reflection* – computing which is 'about itself' – and the manner in which the genotype simultaneously specifies and is maintained by the phenotype.

As we describe below, computational reflection and biological systems have many things in common. A model of computational reflection gives new insight into biological systems. In addition, biological systems give a new perspective on the nature of computational reflection.

We introduce this topic with a summary what reflection means in computer science, and then go on to discuss the implications in ALife.

Computation without reflection: First we present a highly simplified model of *Conventional Computing* (CCOMP), so that the concepts we present below have a clear conceptual base.

In CCOMP, computers run *programs* that process *data*. On execution, both the program and the data are held as binary digits in RAM. The CPU 'reads' the program, which 'acts upon' the data. For our purposes, we can consider that the CPU executes one instruction at a time, and that instruction works on one data word. This is possible because the sequence of operations is specified by the program, and the data is organised into a set of related structures in RAM. If the program is written correctly, it processes the data in the manner intended, even though the CPU never 'sees' the entire program or data at any one time. Although there is no physical distinction between the program and its data, it is usual for the two to be treated separately. The data is *processed* by the program, meaning that some if it is changed or manipulated to form the output; the program is fixed.

The number of instructions needed to do anything useful to data is usually very large. In order to make it easier to write useful programs *high level languages* have been developed, which group sets of instructions together into useful commands. In this way, modern programming languages make it possible to write programs without intimate knowledge of the hardware that the programs run on.

We illustrate this concept in figure 1. This shows the relationship between the code base, the interpreter, and the code that is executing. The *code base* is the program written in some language. It becomes *executing code* via the action of the interpreter. (By *interpreter* we mean whatever process is accessing the code base and executing it.)

Although the von Neumann architecture that forms the basis of CCOMP has program-data equivalence at the word level, there is not usually a direct way for the executing code to reflect aspects of its computation back to the code base or the interpreter. Although some well known languages such as Java support such reflection, there is no requirement to use reflection when writing progams. Such feedback is needed if the program is to be verified, maintained and improved. In the absence of automated feedback mechanisms, these tasks are carried out by human programmers. Models
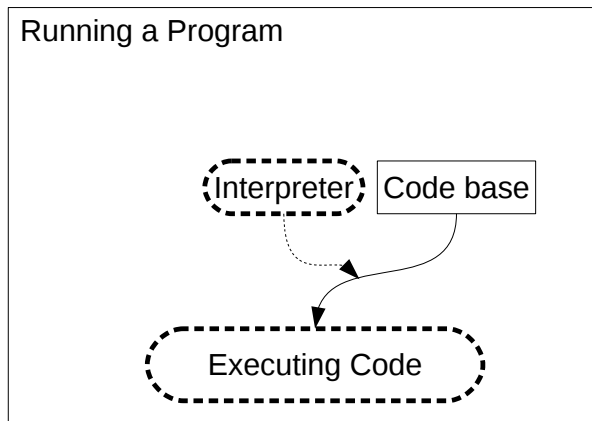
Figure 1: Running a computer program without reflection. The code base is analysed by the interpreter, and run as executing code. Solid boxes are data, Dashed boxes are running programs. Solid arrows indicate the provision of data. Dashed arrows indicate an action upon a process.

of computational reflection attempt to provide this feedback at runtime, which is guaranteed to provide the current context of the computation as it is being performed. Here we begin to see the relationship of reflection to ALife: the current context is the environment in which ALife systems must survive, and the process of life is the equivalent of the computational concept of runtime.

Next, we discuss reflection in abstract terms, and relate it to a series of related concepts in ALife. Then we review the way CCOMP has used reflection in different programming paradigms. The aim is to gather a set of observations on how reflection might work in ALife, which we present in the fourth section. We end with a discussion, and some proposals on ways to implement reflective ALife systems.

Computational reflection

"A reflective system contains structures which represent aspects of itself" (Maes, 1987). Reflection is any act of computing that is 'about itself': it is computation *about* the computation that is being performed, without direct reference to the *goal* of the computation. It is *self-inspection at runtime*, and a candidate definition of what comprises a living system. Self-inspection is of no use unless it is possible to act upon the outcome of the inspection, so CCOMP reflection allows *self-modification*: the ability to create (reify) new sorts of first class objects. We link this feature to living systems in the latter half of this paper.

Reflection requires that a representation of the code is available as data at runtime; the reflective process uses this to deduce which aspects of the program have particular computational features. A data structure representing (a model of) the program itself is created during execution of the pro-
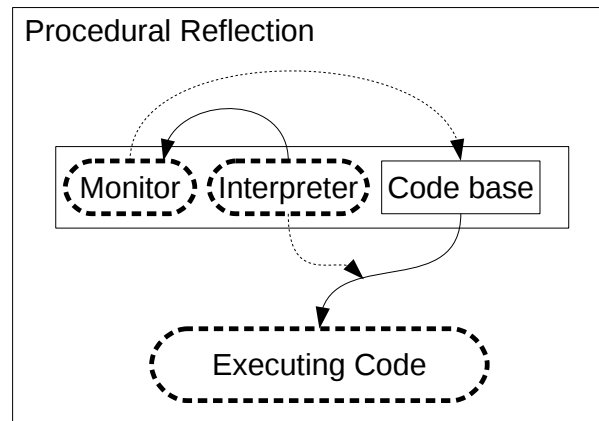
Figure 2: Procedural reflection. Key as in figure 1.

gram, and is used to modify the execution of the program at run-time. A special kind of interpreter (a) gives the running system access to data representing the system; (b) establishes the *causal connection* between the 'executing code' (the running system) and the 'base code' (the *system representation*). Causal connection guarantees that modifications to the executing system are reflected in the code base. All reflective operations depend on the maintenance of the causal connection to ensure that the code base remains a faithful representation of the executing code. Different methods of reflection use different ways to present the program representation as data to the running code (Maes, 1987).

A reflective system can bring about modifications to itself because it is able to generate and analyse data about its own computation. It is able to detect an issue in the executing code and modify the code base. The reflective interpreter reinterprets the code base *during execution*. By endowing a computational process with the power to monitor the computation that is being performed, systems are (theoretically) more able tolerate faults, organise their processing, and even organise their code base in the light of changing conditions (Smith, 1984). How this is achieved depends upon the mode of reflection being carried out. Two sub-classes of computational reflection are described below. The first, *procedural reflection*, allows reasoning about computation by running a model of the interpreter on a model of the code whilst the code is executing. The second, *declarative reflection*, attempts to avoid the costs of procedural reflection by abstraction of the properties of the executing code.

Procedural reflection

Procedural reflection encapsulates the role of the interpreter *within* the executing program, and assigns extra duties to it (Maes, 1987). The components of a procedural reflective architecture are shown in figure 2. The components of standard computation from figure 1 are all present. An interpreter process carries out the execution of the program, but
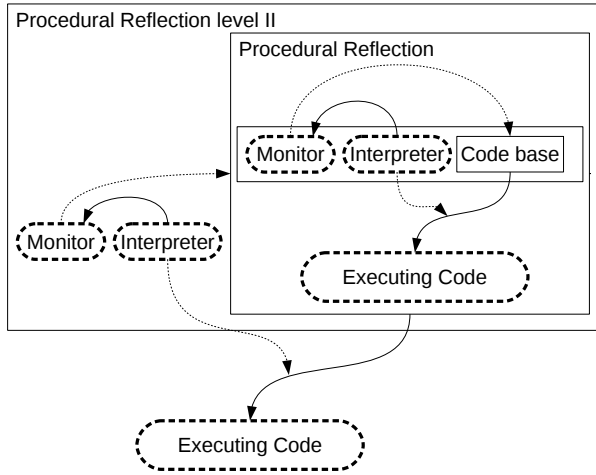
Figure 3: Meta-circularity of procedural reflection shown as two reflective layers. Key as in figure 1.

Figure 4: Declarative reflection. Key as in figure 1.

it also makes information about the computation available to a monitoring process, shown as *Monitor* in the figure. The encapsulated interpreter must guarantee a causal connection between the code base and the executing code. The monitor is able both to reason about the execution of the program, and to act upon this reasoning by making changes to the code base, so changing the execution of the program at run-time.

Procedural reflection requires that a (more or less) complete representation of the program is contained in the reflective layer. This means that it is possible to *generate* the layer below from the representation in the current layer using the interpreter. There is a clash here because the twin goals of *specifying* the system and being able to *reason* about it can be incompatible, leading to duplication of information at the very least.

Challenges in procedural reflection: The method of self-representation offered by procedural reflection has its challenges, centering on the problem of exactly *when* to spawn a process that is 'about' another process, since each process has a computational cost in terms of RAM and CPU. We enter the domain of *meta-circularity* when we realise that the reflective layer in figure 2 is itself a running program. If the interpreter has to have a complete representation of the relationship between the code base and the running program, it follows that a fully reflective architecture would need a second reflective layer, figure 3. Since the reflective representation is part of the running code, it must *also* be monitored, at a higher level. Following this reasoning, we see that the recursion in this model can extend *ad infinitum*, whereby a hierarchy of processes are spawned, each monitoring the process below and with only the process at the bottom doing any actual work. This problem is avoided by bending the rules slightly, letting the interpreters represent only parts of
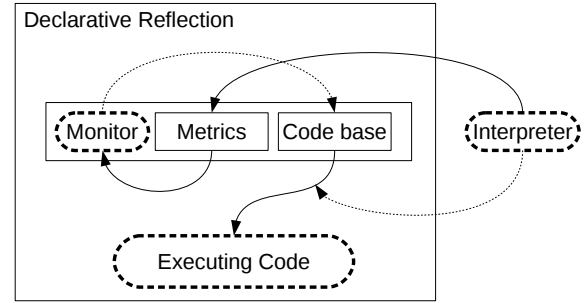
the system at each level and eventually deciding that further recursion is no longer fruitful.

In addition, the role of the interpreter in procedural reflection is complex since it has two tasks to perform: interpreting the code base, and feeding back information on the computation to the monitor. If we forsake the embedding of the interpreter within the reflective layer, some of these problems can be avoided.

Declarative reflection

Declarative reflection avoids the need to specify the code base exactly, and instead seeks to generate useful statements about the system (Maes, 1987), for example, information about the time and space complexity of the executing process. The interpreter sits outside the reflective process, and merely provides it with a set of metrics. The monitor can then act on these metrics and change the code base, which is then used as executing code by the interpreter (figure 4).

The advantage of this approach is that the danger of infinite recursion of reflective layers is greatly reduced (although still possible), and the duties of the interpreter are more clearly defined.

Challenges in declarative reflection The benefits of declarative reflection come at the expense of the ability of the reflective system to detect appropriate conditions that should be acted upon. The metrics can describe only *what* has been done by the system; it is much harder to give a description of *how* the effect has happened, making it more difficult for the Monitor to decide how to implement changes.

For this reason alone, reflective architectures are rarely purely declarative. Most reflective architectures use elements of both procedural reflection and of declarative reflection.

Reflective properties of ALife systems

We seek ways to apply reflective ideas from CCOMP directly to ALife systems, in the hope that the advantages of reflection can be emulated. However, reflection in biology is *different* from reflection in CCOMP. We are trying for a

'unified' treatment to reflective processes in biology, and so possibly find ways to improve reflection both in ALife systems and CCOMP generally.

Having reviewed procedural and declarative reflection, we must also describe an alternative approach to reflection based on phenomena and techniques observed in and inspired by biological systems. But first, we briefly review some of the issues in computational reflection from a biological perspective.

The absence of a 'designer':   How can a biological system be 'about itself' in the absence of a pre-defined purpose? There are two parts to this question. First we consider how a design is specified, then we consider how this goal is met. (See also Dennett (1971) for a discussion of design stances.)

Biological systems use the genotype as a specifier of the phenotype via what Pattee (1982) calls the 'symbol-matter articulation', in which the specification of a machine (the symbol side) and its implementation (the matter side) are related to one another. This articulation is analogous to the requirement for causal connection in reflective languages, but it is at its strongest where the system exhibits *semantic closure* (see later).

Thus, in biology the design of the system seems to be absent from the model of reflection. But what is the goal of self-inspection and self-modification if there is no design to which the system can be compared? The answer is that biological systems introduce 'purpose' via evolution, by using populations of solutions and applying selection to them. Our goal in defining a bio-reflective architecture is to describe how these phenomena combine to yield a reflective system.

Rejecting declarative reflection: Declarative reflection offers a means of avoiding having a sophisticated interpreter in each reflective layer, and reduces the risk of infinite recursion. However, the declarative approach is rigid: it is difficult to detect when it is yielding insufficient information about the system, and it is difficult to implement new declarative statements when required.

Declarative reflection is like extrinsic fitness functions in Evolutionary Algorithms: it runs the risk of over-specifying the problem at hand and ignoring innovative solutions. The declarative approach is problematic in ALife because it adheres to an unchanging (and unchangeable) description of what the design is, implicit in the metrics that are used to monitor the system. It is difficult to define exactly what the declarative statements should be *a priori*, and so it becomes difficult to define what should be measured in order to detect what changes would be beneficial. The declarative approach embeds too much of the reflection in the 'physics' of the system (Hickinbotham et al., 2016), since the metrics are not under control of the 'biology', and so cannot be changed to improve its representation of the running system.

Rejecting procedural reflection: From the perspective of computer science, the two disadvantages of procedural reflection are that it places too many demands on the interpreter to allow an efficient implementation, and that the meta-circularity of the system leads potentially to infinite recursion.

The goal of reflection is to bring about automation in improvement of a computational system, in the same manner as natural selection in biological systems. Biological systems also exhibit recursion in that each organism is created by an earlier organism via a replication process. A key point is that biological systems are organised such that for most of its lifetime an organism is *autonomous*. In CCOMP, reflection requires the the ability to recursively spawn new reflective layers *ad infinitum*, but the 'recursion' in biology is the phylogeny of the individual. In bio-reflection, the individual does not need to spawn instances of its ancestors in order to monitor its state, since the relevant information is packaged up in its genome.

Reflectionless self-replicators   Reflection involves holding a model of the code base and maintaining a causal connection between this and the actual execution of the model on the CPU. There is a direct analogy here with the relationship between the genotype and the phenotype in biology.

In ALife systems, models of biology are subject to experiments by computer simulation. Unlike biology, everything about these model systems is knowable, but everything (including all of the relevant physics) must be initialised, parameterised and implemented. There are also many assumptions about the appropriate representation of such simulations in mainstream computers. The attraction of this approach is that it makes clear the relationship between mechanisms of biological innovation and how these models of biology could be applied to (models of) computation.

Many self-replicating ALife systems exist, but these tend to be modelled on a hypothetical 'RNA world' in which each entity inspects an instance of itself in order to create a copy (Ray, 1991; Ofria and Wilke, 2004; Hickinbotham et al., 2010). These are *automata chemistries* (Dittrich et al., 2001): artificial agent-based systems in which each agent is a program.

Many ALife platforms contain instances of agents that iteratively manufacture copies of themselves. A mechanism for changing the copies, usually called *mutation*, is introduced in order to explore the design space of the system's universe. In this sense the program that the agents are running is *self-modifying*. This process of self-modification is central to mechanisms of reflection.

Although these systems have shown innovation, there seems to be an upper bound on the level of complexity they can attain, even though there is no theoretical limit on the innovation. Could this be related to reflection? We address this question by turning to the work of Von Neumann
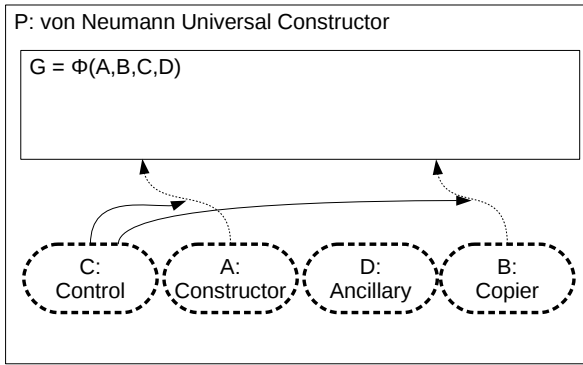
Figure 5: von Neumann's Universal Constructor Architecture. Key as in figure 1.

Figure 6: Comparison of components of procedural reflection (black text) from figure 2 and the Universal Constructor (red text). Key as in figure 1.

et al. (1966), in his *theory of self-reproducing automata*. The point, also made recently by the McMullin group in Dublin (Baugh, 2015; Hasegawa, 2015), is that these systems tend to reproduce by a process of *self-inspection*. von Neumann indicated that there are limitations to reproduction by this method, linked to the difficulty of 'reading' a machine of arbitrary complexity. We argue in addition that these systems are *reflectionless*; although they appear to be 'about' themselves, they are merely sophisticated *quines* (self-copying automata with no inputs) that make no reference to an abstract model of what they are. We have made similar points in (Hickinbotham et al., 2011). We emphasise here that although a system may have program-data equivalence, it is not guaranteed to be 'reflective'; to achieve this, further conditions must be met.

Universal constructors von Neumann's observations about the limitations of reproduction by self-inspection led to the development of his theory of self reproducing automata. He defined a set of sub-assemblies that together formed a Universal Constructor (UC). The original was cellular automaton-based, but the ideas translate to ALife and biological systems. We follow the notation of McMullin (2012) in the following.

The von Neumann architecture comprises four machines A, B, C, D plus their machine descriptions $\Phi(A, B, C, D)$, figure 5. The process of self-reproduction is divided into two parts, which allows machines of arbitrary complexity to be duplicated. Only one entity in the system is copied by inspection. This is G, which consists of an abstract description of everything else in the system: $G = \Phi(A, B, C, D)$. The remaining four machines function as follows. A is the *Constructor*, which can read G and construct (or express) functioning machines from their descriptions. B is the *Copier*, which can create copies of G by inspection (for this reason, G is usually a one-dimensional sequence of instructions). The operation of A and B with respect to G is governed by C, a *Control* structure. D is *Ancillary Machinery*, which
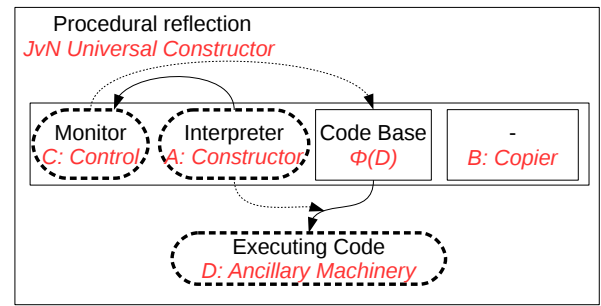
carries out any other function of the system irrespective of the self-replicating assemblages just described.

We illustrate the overlap between von Neumann's Universal Constructor architecture and procedural reflection in figure 6. The layout of this figure follows the procedural reflection diagram in figure 2, and adds the UC nomenclature in red. All of the components of UC bar one are present in procedural reflection, but the naming conventions are different. What we have called the Monitor is called the Controller in the UC, but their roles are identical: to orchestrate the operations of the other sub-assemblies in the overall machine. The Interpreter is mirrored in the UC as the Constructor, which takes a description of a machine and creates the machine based on that description, in the same way that an Interpreter reads source code and creates a working manifestation of the code on a conventional computer architecture. The Code Base in the procedural architecture is represented in the UC as the symbol $\Phi(D)$. Both of these labels represent the abstract concept of a *description of a functioning machine*: D is the functioning machine, and Φ is a description operator. The executing code in the procedural reflection model is referred to as 'D: Ancillary machinery' in the UC nomenclature. The only component that UC adds to the procedural reflection model is the *Copier*, and a description of all the machines, not just the ancillary D, in G. The copier is responsible for duplicating the machine descriptions in Φ. We describe its role in bio-reflection below.

The layout of figure 6 is an unsatisfactory description of bio-reflection because it falls victim to the meta-circular architecture in the same way as shown as in figure 3, but the conceptual link is important for what follows. The UC nomenclature in this figure already gives some clues about what is missing from the model.

Bio-reflection

Having made some observations about reflection in ALife, we now propose a new architecture of self-modification, which we call *bio-reflection*, since its development from
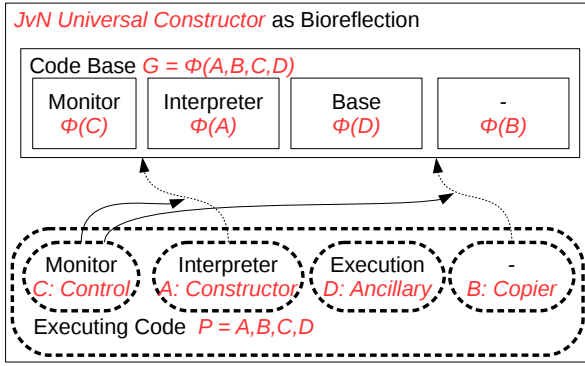
Figure 7: Bio-reflective Architecture. Universal Constructor terms are shown in red. Key as in figure 1.

CCOMP reflection is inspired by biological processes.

Inspection of the mapping between the UC architecture and the procedural reflection architecture in figure 6 shows that some of the components of UC are missing from it. Firstly there is no machine description for anything but the Ancillary Machinery in a single reflective layer. Secondly, there is no Copier.

The absence of descriptions of all the machine appears to be the feature that forces reflective systems into a recursive situation of figure 3. von Neumann solved this problem by specifying that *all* the components in the architecture be represented by abstract descriptions.

The copier is missing from the reflective architecture for two reasons. First, it is easy to copy source code in CCOMP. Second, the concept of reflection requires that the interpreter acts upon the source code *while the program is running*. From this perspective, why would one bother to have an extra machine in the architecture?

Why is a copier important? This is related to the absence of a designer. It is hard to say what a living organism is *about*, because there is no designer, and hence no *purpose* to the organism. This makes the concept of *control* rather more vague: what is it that the entity is being controlled *for*?

Due to mutation, variation in the expression of $\Phi(X)$ means that it is likely that a machine will fail sooner or later. Having a population of individuals insures against that. A population of machines is inevitably exposed to selection. By making the copier an intrinsic part of the machinery, we can ensure that successful individuals are reproduced in the population.

So bio-reflection works on *populations* of machines, which gives the control component of the architecture we seek. This is part of the way we avoid infinite meta-circular reflection. But we have to accept that populations have their own associated computational costs.

The bio-reflective architecture follows the UC architecture, but with some external considerations that we describe below. The overlap between the bio-reflective architecture and UC is shown in figure 7. There are two classes of entity: the *code base* and the *executing code* as in computational reflection. The difference is that *all* the executing machines in the system have a representation in the code base layer. The idea is that, following the UC architecture, the machine descriptions in the code base are sufficiently rich to allow the machines to be constructed from them, but sufficiently simple to be easy to copy by the copy machine. In this way, the system is self-contained (semantically closed), and there is no requirement for a recursive pattern of reflection to further organise the self-modification. Reflective actions that would have happened at a higher level of reflection are handled via changes to the representation in the code base (via the copier), and changes to the way that representation is interpreted by the constructor. These two components together guarantee causal connectedness in a semantically closed manner, without the need for any external specification (beyond the physics of the system, which should be minimised).

Facets of bio-reflection   Mutation is different from self-modification, because of absence of a design: mutations merely modify, then *selection* identifies which of the modifications are improvements. Much of what a monitor has to do are moved to an external process that runs at the population level.

In order to generate the running code, several things must be brought to bear on the description of the machine. The functionality of the Interpreter is the most relevant to this discussion, but this functionality depends on the 'physics' of the system, which is not described the genotype, but is implicitly referenced by it. In this way, the abstract description is incomplete, but consistent with the executing machines. This feature allows us to sidestep the recursion that exists in computational reflection.

The architecture allows new kinds of machines to be reified via two routes. The first route involves the inaccurate copying of the code base via the action of the copier, leading to mutation in one of the machines. The second is a special case of this: an inaccurate reification of the interpreter machine has the potential to change the way the entire code base is interpreted, leading to a change in the way *all* machines are reified.

In this way, the semantics of the code base are self-contained because the machine that interprets the code base is encoded in the code base. This feature guarantees causal connectedness, but also allows the meaning of a code base to be interpreted differently depending on the nature of the interpreting machine, in what Pattee calls *semantic closure*. ALife systems that reproduce via self inspection do not have this feature, and so cannot be said to be reflective.

Semantic Closure and Causal Connectedness Causal connection is a major component of semantic closure, but it says nothing about the semantics of the system and how they arise. In the bio-reflective architecture, semantic closure means that the semantics of the machine descriptions are *embodied* in the relationship between the executing machines and their descriptions. This feature of the system is most prominent in the encoding of the interpreter, which has to read a description of itself, and construct a copy of itself from that description. If, by mutation or other error during construction, the function of the interpreter changes, then the whole meaning of Φ is changed.

Examples of such a change of meaning are well known in biology (Foster, 2007). For example, the 'SOS response' to DNA damage in *E. coli* involves the expression of RNA polymerases that are more able to successfully copy damaged DNA, but with lower fidelity, thus increasing the mutation rate whilst the population is under stress. A bio-reflective ALife system has the potential to emulate such phenomena.

A different continuum Maes (1987) indicates that most computational reflective architectures were somewhere between the Procedural and Declarative extremes. In most reflective systems, the code base is represented sufficiently explicitly to allow the reflection to occur, and the 'missing' parts of the code base are represented by declarative statements. The bio-reflective approach offers a different perspective on this: in biological systems the code base (genome) is augmented by the physical and chemical processes in the cell, allowing the enzyme 'machines' to be constructed by the ribosome; in computational ALife systems, the virtual physico-chemical processes are necessarily less complex, consisting of the function of opcodes, and the 'given' computational machinery (registers, stacks, etc) assigned to each individual.

Another continuum is to do with the relationship between mutation and selection: mutation pushes newly created automata towards a random/disordered state, but selection ensures that the reflective processes of interpretation and monitoring are maintained.

Discussion

Reflection and the design stance If computation is to be about itself, we need some definition of what the 'self' is. Here, biology is more straightforward, because the design of the organism is self-contained. The issue is more complicated in software because it is engineered: it has a designer, a purpose, and an implementation in source code.

We have an immediate difference between engineered reflection and emergent reflection. Engineered reflection requires the conscious act of building a reflective system. Computation can happen without reflection *precisely because* the computation is engineered: it has a designer. One

of the problems is that reflection within these architectures holds the designer as a 'third option' in which the ultimate design can reside. The AI community have reflection to be about debugging, optimising etc, which is the origin of Smith's 'computing about itself' metaphor. *Reflective acts* are anything that is done 'about' the computation, such as debugging, optimising etc. This assumes the design is 'known' and that we are attempting to refine the implementation to reflect the design.

How can biological processes be reflective if they are not designed? This is a core question that we must answer if we are to build a bridge between computer and biological sciences. Firstly, we note that in some ways, the absence of a designer makes reflection simpler: the 'ultimate' design is simply the system itself; whether the design resides in the genotype or the phenotype is immaterial. Whereas in CCOMP, programs that did things could exist before reflective processes were available, in biology *this was not possible*: how could a process emerge that was 'doing' something before the system became self-referential? Without the self-referential process, the biology is nothing more than complex (carbon) chemistry. Only when a specification and an interpreter became available did life truly emerge.

We are not refuting RNA world with this argument; we are merely stating that even there, some RNA would be template, and some would be machine.

On the 'self' in Computer Science Unlike biological organisms, computer programs are *designed*. Following from this, Smith (1984) noted that we have an issue: what *is* the design of the program? Is it the concepts in the programming team's heads? The source code? The executing code? The answer appears to be a combination of all three: the concepts give the broad thrust of what needs to be done; the source code is an instantiation of these ideas, plus bugs; the executing code is what actually happens, which is what the source code is trying to persuade the interpreter to do. But some of the actions of the interpreter then become part of the design, and these are not necessarily in the source code.

McMullin's lab has attempted to build instances of the von Neumann replicating architecture in Tierra and Avida. Both implementations of the von Neumann architecture in these systems tend to collapse to their original RNA world configurations of replication, unless strict constraints are placed on the evolution. For example in (Baugh, 2015), the system could only be made viable by deleting any offspring with a different length from the parent. These results are important, because they allow us to identify features of automata chemistries that foster the more sophisticated self-reproducing entities described by von Neumann.

The model we propose could be implemented as an emulation of biology by allowing the monitor to act as a gene regulator only, or towards CCOMP-style learning system, by configuring the Monitor to pass 'message sends' to the

selection process. In either configuration, the reflective acts are fully autonomous and internally consistent.

Conclusion

The dream of AI is to have systems that adjust themselves to meet our needs, be they robotic, informational or biological. The foundation of these systems is that they are reflective: they are able to reason about themselves. This problem is encountered in ALife, Artificial Intelligence and CCOMP, as noted by Pattee (1982). By considering ALife as reflective systems, we are more able to draw from this wider body of research and move the field forward.

It is remarkable that the Universal Constructor design from 1949 is still relevant today. By casting it as a reflective system, new emphasis can be placed on the components of an ALife system, suggesting new avenues for research in both Artificial Life and Computer Science. To quote a review of the previous draft of this paper: "Interesting questions present themselves: should we be using reflective languages to build reflective replicators? Would self-awareness in a replicator, i.e., introspection into its own method of replication/ecological niche, enable an enhanced form of autoconstructive evolution a la Spector and Robinson (2002)? A Lamarckian-Darwinism where organisms simulate their offspring in sandboxes and hack their own genomes accordingly?"

CCOMP reflection does not account for copying of code bases in the reflective act, and the von Neumann architecture does not make clear the role of the monitor, or take natural selection into account within the model. With our bio-reflective architecture presented here, we have arrived at a consistent representation that clarifies the distinction between CCOMP and biology, and provides an intellectual basis for future ALife implementations.

Acknowledgements

References

Baugh, D. (2015). *Implementing von Neumann's architecture for machine self reproduction within the Tierra artificial life platform to investigate evolvable genotype-phenotype mappings*. PhD thesis, Dublin City University.

Dennett, D. C. (1971). Intentional systems. *The Journal of Philosophy*, 68(4):87–106.

Dittrich, P., Ziegler, J., and Banzhaf, W. (2001). Artificial chemistries – a review. *Artificial Life*, 7(3):225–275.

Foster, P. L. (2007). Stress-induced mutagenesis in bacteria. *Critical reviews in biochemistry and molecular biology*, 42(5):373–397.

Hasegawa, T. (2015). *On the evolution of genotype-phenotype mapping: exploring viability in the Avida artificial life system*. PhD thesis, Dublin City University.

Hickinbotham, S., Clark, E., Nellis, A., Stepney, S., Clarke, T., and Young, P. (2016). Maximising the adjacent possible in automata chemistries. *Artificial Life*, 22(1):49–75.

Hickinbotham, S., Clark, E., Stepney, S., Clarke, T., Nellis, A., Pay, M., and Young, P. (2010). Diversity from a monoculture: Effects of mutation-on-copy in a string-based artificial chemistry. In *ALife XII*, pages 24–31. MIT Press.

Hickinbotham, S., Stepney, S., Nellis, A., Clarke, T., Clark, E., Pay, M., and Young, P. (2011). Embodied genomes and metaprogramming. In *ECAL 2011*, pages 334–341. Springer.

Maes, P. (1987). Concepts and experiments in computational reflection. *ACM Sigplan Notices*, 22(12):147–155.

McMullin, B. (2012). Architectures for self-reproduction: Abstractions, realisations and a research program. In *ALife XIII*, pages 83–90. MIT Press.

Ofria, C. and Wilke, C. O. (2004). Avida: A software platform for research in computational evolutionary biology. *Artificial Life*, 10(2):191–229.

Pattee, H. H. (1982). Cell psychology: an evolutionary approach to the symbol-matter problem. *Cognition and Brain Theory*, 5(4):325–341.

Ray, T. S. (1991). An approach to the synthesis of life. In C. Langton, C. Taylor, J. D. Farmer, S. Rasmussen, editor, *ALife II*, pages 371–408. Addison-Wesley.

Smith, B. C. (1984). Reflection and semantics in Lisp. In *Proc. 11th ACM SIGACT-SIGPLAN symposium on Principles of Programming Languages*, pages 23–35. ACM.

Spector, L. and Robinson, A. (2002). Genetic programming and autoconstructive evolution with the push programming language. *Genetic Programming and Evolvable Machines*, 3(1):7–40.

Von Neumann, J., Burks, A. W., et al. (1966). Theory of self-reproducing automata. *IEEE Transactions on Neural Networks*, 5(1):3–14.

The Limits of Decidable States on Open-Ended Evolution and Emergence[*]

Santiago Hernández-Orozco[1], Francisco Hernández-Quiroz[2] and Hector Zenil[3,4]

[1]Posgrado en Ciencias e Ingeniería de la Computación, UNAM, Mexico.
[2]Departamento de Matemáticas, Facultad de Ciencias, UNAM, Mexico
[3]Department of Computer Science, University of Oxford, UK
[4]Algorithmic Nature Group, LABORES, Paris, France
[1]hosant@ciencias.unam.mx, [2]fhq@ciencias.unam.mx, [3]hector.zenil@algorithmicnaturelab.org

Abstract

Using algorithmic complexity theory methods, we propose a robust computational definitions for open-ended evolution (OEE) and adaptability of computable dynamical systems. With this framework, we show that decidability imposes absolute limits to the growth of complexity on computable dynamical systems up to a logarithm of a logarithmic term. Conversely, systems that exhibit open-ended evolution must be undecidable and have irreducible behaviour through the evolution of the system. Complexity is assessed in terms of three measures: sophistication, coarse sophistication and busy beaver logical depth.

Introduction and Preliminaries

Broadly speaking, a dynamical system is one that changes over time. Prediction of the future behaviour of a dynamical system is a main issue for science generally: scientific theories are tested upon the accuracy of their predictions; and establishing invariable properties through the evolution of a system is an important goal. Limits to this predictability are known in science. For instance, chaos theory establishes the existence of systems in which small deficits in the information of the initial states makes accurate predictions of future states unattainable. However, on this document we focus on systems for which we have unambiguous, finite (on size and time) and complete descriptions of their initial states and their behaviour: computable dynamical systems.

Since their formalization by Church and Turing, the class of computable systems have shown that, even without information deficits (i.e., with complete descriptions), there are future states that cannot be predicted, in particular the state known as *halting state* (Turing, 1936). We will use this result to show how prediction imposes limits to the growth of complexity during the evolution of a system.

The relationship between dynamical systems, computability and Turing machines, along with the implied unpredictability of their behaviour, was observed by Moore

(Moore, 1991) and Wolfram (Wolfram, 2002). Delvenne, Kurka and Blonde (Delvenne et al., 2006) have explored robust definitions for computable (effective) dynamical systems and universality generalizing Turing's halting states, along with conditions and implications for universality, decidability and their relationship with chaos. Undecidability of certain properties for certain analytic dynamical systems have been studied by Bournez (Bournez et al., 2013). The definitions and general approach used in this paper differ from those sources, but are ultimately related.

Computable Functions

In a broad sense, an object x is *computable* if it can be described by a Turing machine (Turing, 1936); for example if there exists a Turing machine that produces x as an output. Is clear that any finite string on a finite alphabet is a computable object. Following Turing's tradition, we provide below a more formal definition.

As usual, we can define a 1 to 1 mapping between the set of all finite binary strings $\mathbb{B}^* = \{0,1\}^*$ and the natural numbers by the relation induced by the lexicographic order of the form: $\{(``", 0), (``0", 1), (``1", 2), (``00", 3), ...\}$. Using this relation we can see all natural numbers (or positive integers) as binary strings and vice versa. Accordingly all natural numbers are computable.

A string p is a *valid program* for the Turing machine T if during the execution of T with p as input all the characters in p are read. We call $T(p)$ the output of the machine, if it stops. A Turing Machine is *prefix-free* if no valid program can be a proper substring of another valid program (but can be a postfix of one). We call a valid program a *self delimited object*. Note that, given the relationship between natural numbers and binary strings, the set of all valid programs is an infinite proper subset of the natural numbers.

Formally, a function $f : \mathbb{N} \to \mathbb{N}$ is *computable* if there exist a Turing Machine T such that $f(x) = T(x)$. A Turing Machine U is called *universal* if there exist a computable function g such that for every Turing machine T there exist a string $\langle T \rangle \in \mathbb{B}^*$ such that $f(x) = U(\langle T \rangle g(x))$, where $\langle T \rangle g(x)$ is the concatenation of the strings $\langle T \rangle$ and $g(x)$.

---

[*]The adapted definitions and complete proofs used in this article can be found at the extended version of this article http://complexitycalculator.com/algorithmicOEE.zip

Given the previous case, $\langle T \rangle$ and $g(x)$ are called a *codification or representations* of the function f and the natural number x, respectively. From now on we will denote by $\langle f \rangle$ and $\langle x \rangle$ the codification of f and x. The codification $g(x)$ is *unambiguous* if is injective.

For functions with more than one variable, if x is a pair $x = (x_1, x_2)$, we say that the codification $g(x)$ is unambiguous if its injective and the inverse functions $g_1^{-1} : g(x) \mapsto x_1$ and $g_2^{-1} : g(x) \mapsto x_2$ are computable. If x is a tuple $(x_1, ..., x_i, ..., x_n)$, then the codification $g(x)$ is unambiguous if the function $(x, i) \mapsto x_i$ is computable.

A sequence of strings $\delta_1, \delta_2, ..., \delta_i, ...$ is computable if the function $\delta : i \mapsto \delta_i$ is computable. A real number is computable if its decimal expansion is a computable sequence. For complex numbers and higher dimensional spaces, we say that they are computable if each of its coordinates are also computable.

Finally, for each of the described objects, we call the representation of the associated Turing machine *the representation of the object for the reference Turing machine U*, and we define computability of further objects by considering their representations. For example, a function $f : \mathbb{R} \to \mathbb{R}$ is computable if the mapping $\langle x_i \rangle \mapsto \langle f(x_i) \rangle$ is computable and we will denote by $\langle f \rangle$ the representation of the associated Turing machine, calling it the codification of f itself.

Algorithmic Descriptive Complexity

Given a prefix-free universal Turing Machine U with alphabet Σ, the *algorithmic descriptive complexity* (also known as Kolmogorov complexity and Kolmogorov-Chaitin complexity (Kolmogorov, 1965; Chaitin, 1982) of a string $s \in \Sigma^*$ is defined as

$$K_U(s) = \min\{|p| : U(p) = s\},$$

where U is a universal prefix-free Turing Machine and $|p|$ is the number of characters of p.

The algorithmic descriptive complexity measures the minimum amount of information needed to fully describe a computable object within the framework of a universal Turing machine U. If $U(p) = s$ then the program p is called a description of s, the first of the smallest descriptions (on alphabetical order) is denoted by s^* and by $\langle s \rangle$ a non necessarily minimal description computable over the class of objects. If M is a Turing machine, a program p is a description or codification of M for U if for every string s we have that $M(s) = U(p\langle s \rangle)$. In the case of numbers, functions, sequences and other computable objects we consider the descriptive complexity of its smallest description. For example, for a computable function $f : \mathbb{R} \to \mathbb{R}$, $K(f)$ is defined as $K(f^*)$ where $f^* \in \mathbb{B}^*$ is the first of the minimal descriptions for f.

Of particular importance for this document is the *conditional descriptive complexity*, which is defined as:

$$K_U(s|r) = \min\{|p| : U(pr) = s\},$$

where pr is the concatenation of p and r. This measure can be interpreted as the *smallest amount of information needed to describe s given a full description of r*. We can think of p as a program with input r.

One of the most important properties of the descriptive complexity measure is its *stability*: the difference between the descriptive complexity of an object, given two universal Turing machines, is at most constant. Therefore the reference machine U is usually omitted in favor of the *universal measure K*. From now on we will omit the subscript from the measure.

Randomness   A string x is known as *r-random* or *incompressible* if $K(x) \geq |x| - r$. This definition states that a string is random if it does not has a significantly shorter complete description than the string itself. A simple counting argument shows the existence of random strings. Now, is easy to verify that every string x has a self delimited computable unambiguous codification with strings of the form $1^{\log|s|}0|s|s$ (Li and Vitányi, 1997, section 1.4). Therefore, there exist a natural r such that if x is r-random then $K(x) = |x| - r + O(\log|x|)$, where $O(\log|x|)$ is a positive term. We will say that such strings hold the randomness inequality *tightly*.

Let M be a halting Turing Machine with description $\langle M \rangle$ for the reference machine U. A simple argument can show that the halting time of M cannot be a large *random* number: let U^H be a Turing Machine that emulates U while counting the number of steps, returning the execution time upon halting; if r is a large random number then M cannot stop in time r, otherwise the program $\langle U^H \rangle \langle M \rangle$ will give us a *short* description of r. This argument is summarized by the following inequality:

$$K(T(M)) \leq K(M) + O(1), \tag{1}$$

where $T(M)$ is the number of steps that took the machine M to reach the halting state, the *execution time* of the machine M.

Computable Dynamical Systems

Formally, a *dynamical system* is a rule of evolution in time within a state space; space that is defined as the set of all possible states of the system (Meiss, 2007). For this work we will focus in a functional model for dynamical systems with a constant initial state and variables representing the previous state and the time of the system. This model allows us to set halting states for each time on a discrete scale in order to study the impact of the descriptive complexity of time during the evolution of a discrete computable system.

A deterministic discrete space system is defined by an *evolution function (or rule)* of the form $M_{t+1} = S(M_0, t)$, where M_0 is called the *initial state* and t is a positive integer called the *time* variable of the system. The sequence of

states $M_0, M_1, ..., M_t, ...$ is called the evolution of the system. Given a reference universal Turing Machine U, if S is a computable function and M_0 is a computable object, we will say that S is a *computable dynamical system*. An important property of computable dynamical systems is the uniqueness of the successor state which implies that equal states must evolve equally given the same evolution function, in other words:

$$M_t = M_{t'} \implies M_{t'+1} = M_{t+1}. \tag{2}$$

The converse is not necessarily true.

Now, a *complete description of a computable system* $S(M_0, t)$ should contain enough information to compute the state of the system at any time and hence it must entail the codification of its evolution function S and a description of the initial state M_0, which is denoted by $\langle M_0 \rangle$. As a consequence, if we only describe the system at time t by a codification of M_t, then we do not have enough information to compute the successive states of the system. So we will define the *complete description* of a computable system at the time t as a unambiguous codification of the ordered pair composed by $\langle S \rangle$ and $\langle M_t \rangle$, i.e. $\langle (S, \langle M_t \rangle) \rangle$, with $\langle (S, \langle M_0 \rangle) \rangle$ representing the initial state of the system. It is important to note that, for any computable and unambiguous codification function g of the stated pair, we have

$$K(\langle (S, \langle M_t \rangle) \rangle) \leq K(S) + K(M_0) + K(t) + O(1);$$

as we can write a program that uses the descriptions for S, M_0 and t to find the parameters and then evaluate $S(M_0, t)$, finally producing M_t.

Open-Ended Evolution in Computable Dynamical Systems

Defining and establishing the properties required for a system to exhibit **Open-ended evolution (OEE)** is considered an open question (Bedau et al., 2000; Soros and Stanley, 2014; Standish, 2003) and OEE has been proposed as a required property of evolutionary systems capable of producing life (Ruiz-Mirazo et al., 2002). This has been implicitly verified by various experiments *in-silico* (Lindgren, 1992; Adami and Brown, 1994; Lehman and Stanley, 2008; Auerbach and Bongard, 2014).

A line of thought posits that open-ended evolutionary systems tend to produce families of objects of increasing *complexity* (Bedau, 1998; Auerbach and Bongard, 2014). Following this idea, OEE in a computable dynamical system can be characterized as a process that has the property of producing families of objects of increasing *complexity*. Formally, given a *complexity measure* C, we say that a computable dynamical system S exhibits *open-ended evolution* with respect to C if for every time t there exists a time t' such that the complexity of the system at the time t' is greater than the complexity at the time t, i.e. $C(S(M_0, t)) <$ $C(S(M_0, t'))$, where a complexity measure is a (not necessarily computable) function that goes from the state space to a positive numeric space.

The existence of such systems is trivial for complexity measures on which any infinite set of the natural numbers (not necessarily computable) contains a subset where the measure grows strictly:

Lemma 1. *Let C be a complexity measure such that any infinite set of natural numbers has a subset where C grows strictly. Then a computable system $S(M_0, t)$ is a system that produces an infinite number of different states if and only if it exhibits OEE for C.*

Given the previous lemma, a trivial computable system that simply produces all the strings in order exhibits OEE on a class of complexity measures that includes algorithmic description complexity. However, intuitively, we conjecture that such systems have a much simple behaviour compared to what we observe on the natural world and the cited artificial life systems. We can avoid some of these issues with a stronger version of OEE.

Definition 2. *A sequence of naturals $n_0, n_1, ..., n_i, ...$ exhibits strong open-ended evolution* (strong OEE) *with respect to a complexity measure C if for every index i there exists an index i' such that $C(n_i) < C(n_{i'})$, and the complexity of the sequence $C(n_0), C(n_1), ..., C(n_i), ...$ does not drop significantly, i.e. $i \leq j$ implies $C(n_i) \leq C(n_j) + \gamma(j)$ where $\gamma(j)$ is a positive function that does not grow significantly.*

It is important to note that, while the definition of OEE allows significant drops of complexity during the evolution of a system, strong OEE requires for the complexity of the system to not decrease *significantly* during its evolution. In particular we will ask for $C(n_j) - \gamma(j)$ to not be upper-bounded for any infinite subsequence.

Various complexity measures have been proposed that assign low complexity to an infinity of natural numbers deemed to be *simple*. Two examples of such measures are logical depth (Bennett, 1988) and sophistication (Koppel, 1988). Nonetheless, if C is a complexity measure capable of measuring OEE then there must exist infinite sets where C grows strictly. A trivial counting argument shows that the algorithmic descriptive complexity is unbounded in any infinite set, therefore is also unbounded in any set where any other complexity measure grows strictly. Formally:

Lemma 3. *If a system S exhibits OEE (and strong OEE) for a complexity measure C then it also shows OEE with respect to the descriptive complexity K.*

Given the previous lemma, the results shown in the next section can also be extended to any other complexity measure capable of showing OEE.

A Computational Model for Adaptation

Lets start by characterizing the evolution of an organism or a population by a computable dynamical system. It has been argued that, in order for *adaptation* and survival to be possible, an organism must contain a representation of the environment so that, given a reading of the representation, the organism can choose a behaviour accordingly (Zenil et al., 2012). The more approximate this representation is, the better the organism is adapted to its environment. If the organism is computable, this information can be codified by a computable structure. We will denote this structure by M_t, where t stands for the time corresponding to each of the stages of the evolution of the organism. This information is then processed following a finitely specified unambiguous set of rules that, in finite time, will determine the adapted behaviour of the organism according to the information codified by M_t. We will denote this behaviour (or a theory explaining it) with the program p_t. An adapted system is one that produces an acceptable approximation of its environment. An environment can also be represented by a computable structure E. In other words, the system is adapted if $p_t(M_t)$ produces E. Based on this idea we propose a robust, formal definition for adaptation:

Definition 4. Let K be the prefix-free descriptive complexity. We say that the system at the state M_n is ϵ-*adapted* to the E if:

$$K(E|S(M_0, E, n)) \leq \epsilon. \tag{3}$$

The inequality states that the minimal amount of information that is needed to describe E from a complete description of M_n is ϵ or less. This information is provided in form of a program p that produces E from the system at the time n. We will define such programs p as the *adapted behaviour* of the system. Uniqueness for p is not required.

The proposed structure for adapted systems is robust since $K(E|S(M_0, E, n))$ is equal or less than the numbers of characters needed to describe any computable method of describing E from the state of the system at the time n, either be a computable theory for adaptation or a computable model for an organism that tries to predict E. Follows that any computable characterization of adaptation that can be described within ϵ number of bits meets the definition of ϵ-*adapted* given suitable choice of E, the *adaptation condition* for any given environment.

As a simple example, we can think of an organism that must find the food located at the coordinates (x, j) on a grid in order to survive. If the information of an organism is codified by a computable structure M (such as DNA), and there is a set of finitely specified, unambiguous rules that govern how this information is used (such as the ones specified by biochemistry and biological theories) codified by a program p, then we say that the organism finds the food if $p(M) = (j, k)$. If $|\langle p \rangle| \leq \epsilon$, then the we say that the organism is adapted according to a behaviour that can be described

within ϵ characters. The proposed model for adaptation is not limited to such simple interactions. For a start, we can suppose that the organism *sees* a grid, denoted by g, of size $n \times m$ with food at the coordinates (j, k). The environment can be codified as a function E such that $E(g) = (j, k)$ and ϵ-adapted implies that the organism defined by the genetic code M, which is interpreted by a theory or behaviour written on ϵ bits, is capable of finding the food upon seeing g. Similarly, more complex computational structures and interactions imply ϵ-adaptation.

Now, describing an evolutionary system that (eventually) produces an ϵ-*adapted* system is trivial via an enumeration machine (the program that produces all the natural numbers in order), as it will eventually produce E itself; moreover, we want for the output of our process to remain adapted. Therefore we propose an stronger condition called *convergence*:

Definition 5. Given the description of a computable dynamical system $S(M_0, E, t)$ where $t \in \mathbb{N}$ is the variable of time, M_0 is an initial state and E is an environment, we say that the system S *converges* towards E with degree ϵ if there exist δ such that $t \geq \delta$ implies $K(E|S(M_0, E, t)) \leq \epsilon$.

For a fixed initial state M_0 and environment E, is easy to see that the descriptive complexity of a state of the system depends mostly on t: we can describe a program that, given full descriptions of S, E, M_0 and t finds $S(M_0, E, t)$, therefore

$$K(S(M_0, E, t)) \leq K(S) + K(E) + \tag{4}$$
$$K(M_0) + K(t) + O(1),$$

where the constant term is the length of the program described. In order words, as the time t grows, *time is the main driver of the descriptive complexity within the system.*

Non-Randomness of Decidable Convergence Times

One of the most important issues for science is the prediction of the future behaviour of dynamical systems. The prediction that we focus on is that of the first state of convergence (definition 5): Will a system converge and how long it will take? In this section we show what the first limit that decidability imposes to the complexity of the first convergent state is. A consequences of this is the existence of undecidable adapted states.

Formally, for the convergence of a system S with degree ϵ to be decidable there must exist an algorithm D_ϵ such that $D_\epsilon(S, M_0, E, \delta) = 1$ if the system is convergent at the time δ and 0 otherwise. Moreover, we can describe a machine P such that given full descriptions of D_ϵ, S and M_0 it runs D_ϵ with inputs S and M_0 while running over all the possible times t, returning the first t for which the system converges. Note that $\delta = P(\langle D_\epsilon \rangle \langle S \rangle \langle M_0 \rangle \langle E \rangle)$, hence we have a short description of δ and therefore δ *cannot be random*: if $S(M_0, E, t)$ is a convergent system then

$$K(\delta) \leq K(D_\epsilon) + K(S) + K(E) + K(M_0) + O(1), \tag{5}$$

where δ is the first time at which convergence is reached. Note that all the variables are known at the initial state of the system. This result can be resumed by the following lemma:

Lemma 6. *Let S be a system convergent at the time δ. If δ is considerably more descriptively complex than the system and the environment, i.e. for every reasonably large natural number d we have that*

$$K(\delta) > K(S) + K(E) + K(M_0) + d,$$

then δ cannot be found by an algorithm described within d number of characters.

We call such times *random convergence times* and the state of the system M_δ a *random state*. It is important to notice that the descriptive complexity of a random state must be also high:

Lemma 7. *Let S be a convergent system with a complex state $S(M_0, E, \delta)$. For every reasonably large d we have that*

$$K(S(M_0, E, \delta)) > K(S) + K(E) + K(M_0) + d.$$

In other words, if δ has high descriptive complexity, then there does not exist a reasonable algorithm that finds it even if we have a complete description of the system and its environment. It follows that the descriptive complexity of a computable convergent state cannot be much greater than the descriptive complexity of the system itself.

What a *reasonably large d* is has been handled so far with ambiguity as it represents the descriptive complexity of any computable method D_ϵ. We might intend to find convergent times, which intuitively cannot be arbitrarily large. It is easy to '*cheat*' on the inequality 5 by including in the description of the program D_ϵ the full description of the convergence time δ, which is why we ask for *reasonable* descriptions.

Another question left to answer is whether complex convergent times do exist for a given limit d, considering that the limits imposed by the inequality 5 loosen up in direct relation to the descriptive complexity of S, E and M_0.

The next result answers both questions by proving the existence of complex convergent times for a broad characterization of the size of d:

Lemma 8 (Existence of Random Convergence Times). *Let F be a total computable function. For any ϵ, there exist a system $S(M_0, E, t)$ such that the convergence times are $F(S, M_0, E)$-random.*

Let us focus on what the previous lemma is saying: F can be any computable function. It can be a polynomial or exponential function with respect to the length of a given descriptions for M_0 and E. It can also be any computable theory that we might propose for setting an upper limit to the size of an algorithm that finds convergence times given descriptions of the system behaviour, environment and initial state. In other words, for a class of dynamical systems, finding convergence times, therefore convergent states, is not decidable even with complete information of the system and its initial state.

Randomness of Convergence in Dynamic Environments.

So far we have limited the discussion to fixed environments. However, as observed in the physical world, the environment itself can change over time. We call such environments *dynamic environments*. In this section we extend the previous results to cover environments that change depending on time as well as on the initial state of the system. We also propose a weaker convergence condition called *weak convergence* and propose a necessary (but not sufficient) condition for the computability of convergent times called *descriptive differentiability*.

We can think of an environment E as a dynamic computable system, a moving target that also changes with time and depends on the initial state M_0. In order for the system to be convergent, we propose the same criterion: there must exist δ such that $n \geq \delta$ implies

$$K(E(M_0, n)|S(M_0, E(M_0, n), n)) \leq \epsilon. \tag{6}$$

A system with a dynamic environment also meets the inequality 5 and lemmas 6 and 8 since we can describe a machine that run both S and E for the same time t.

Now with dynamic environments, E is a moving target and therefore is convenient to consider an *adaptation period* for the new states of E:

Definition 9. We say that S *converges weakly* to E if there exist an infinity of times δ_i such that

$$K(E(M_0, \delta_i)|S(M_0, E(M_0, \delta_i), \delta_i)) \leq \epsilon. \tag{7}$$

As direct consequence of the inequality 5 and lemma 8 we have the following lemma:

Lemma 10. *Let $S(M_0, E(M_0, t), t)$ be a weakly converging system. Any decision algorithm $D_\epsilon(S, M_0, E, \delta_i)$ can only decide the first non-random time.*

As noted above, these results do not change when dynamic environments are considered. In fact, we can think of static environments as a special case of dynamic environments. However, with different targets of adaptability and convergence, it makes sense to generalize beyond the first convergence time. Also, it should be noticed that specifying a convergence index adds additional information that a decision algorithm can potentially use.

Lemma 11. *Let $S(M_0, E(M_0, t), t)$ be a weakly converging system with an infinity of random times such that $k > j$ implies that $K(\delta_k) = K(\delta_j) + \Delta K_\delta(j, k)$, where ΔK_δ*

is a (not necessarily computable) function with range on the positive integers. If the function $\Delta K_\delta(i, i + m)$ is unbounded with respect to i then any decision algorithm $D_\epsilon(S, M_0, E, i)$, where i is the i-th convergence time, can only decide a finite number of i's.

One direct consequence of the previous lemma is that if a sequence of times $\delta_1, \delta_2, ..., \delta_i, ...$ is decidable then for every m there exist a constant $c_{\delta,m}$ such that

$$\Delta K_\delta(i, i + 1) = K(\delta_{i+m}) - K(\delta_i) \leq c_{\delta,m}.$$

which we can be generalized as:

Definition 12. Let $\delta_1, \delta_2, ..., \delta_i, ...$ be a strictly growing sequence of natural numbers. We define the *descriptive derivative* of the natural mapping $\delta : i \mapsto \delta_i$ as

$$\Delta K_\delta(m) = \min\{c : |K(\delta_{i+m}) - K(\delta_i)| \leq c, i \in \mathbb{N}\}. \quad (8)$$

As a direct consequence of lemma 11, the existence of a descriptive derivative is a necessary condition for the computability of δ; thus not meeting this property is sufficient, but not necessary, for undecidability. Therefore the existence of a descriptive derivative is a stronger condition which we will call *non-descriptively differentiable*.

Definition 13. We say that a sequence of times is $\delta_1, \delta_2, ..., \delta_i, ...$ is *non-descriptively differentiable* if $\Delta K_\delta(m)$ is not a total function.

Irreducibility of Descriptive Time Complexity

At the previous section, it was established that time was the main factor in the descriptive complexity of the states within the evolution of a system. This result is expanded by the time complexity stability theorem (14). This theorem establishes that, within an algorithmic descriptive complexity framework, similarly complex initial states must evolve into similarly complex future states over similarly complex time frames, effectively erasing the difference between the complexity of the state of the system and the complexity of the corresponding time and establishing absolute limits to the reducibility of future states.

Let $F(t) = T(S(M_0, E, t))$ be the *real execution time* of the system at time t. By using our time counting machine U^H it is easy to see that $F(t)$ is computable and, by uniqueness of successor state, F increases strictly with t, and hence it is injective. Consequently, F has a computational inverse F^{-1} over its image. Therefore, we have that (up to a small constant) $K(F(t)) \leq K(F) + K(t)$ and $K(t) \leq K(F^{-1}) + K(F(t))$. Follows that, $K(t) = K(F(t)) + O(c)$, where c is an integer independent of t (but that can depend on S). In other words, for a fixed system S, the execution time and the system time are *equally complex up to a constant*. From here on I will not differentiate between the complexity of both times. A generalization of the previous equation is given by the following theorem:

Theorem 14 (Time Complexity Stability). *Let S and S' be two computable systems and t and t' the first time where each system reaches the states M_t and $M'_{t'}$ respectively, then exist c such that $|K(M_t) - K(t)| \leq c$ and $|K(M_t) - K(M'_{t'})| \leq c$.*

Beyond Halting States: Open-Ended Evolution

Inequality 5 states that being able to predict or recognize adaptation imposes a limit to the descriptive complexity of the first adapted state. A particular case is the halting state, as shown at the proof of lemma 8. By lemma 3 this result holds for any complexity measure. In this section we extend the lemma to continuously evolving systems, showing that computability of adapted times limits the complexity of adapted states beyond the first, imposing a limit to open-ended evolution for three complexity measures: sophistication, coarse sophistication and busy beaver logical depth.

For a system in constant evolution converging to a dynamic environment, the lemma 11 imposes a limit to the growth of descriptive complexity of a system with computable adapted states: *if the growth of the descriptive complexity of a sequence of convergent times is unbounded in the sense of definition 13 then all but a finite number of times are undecidable.* The converse would be convenient, however it is not always true. Moreover, the next series of result shows that imposing such limit would impede strong OEE:

Theorem 15. *Let S be a non cyclical computable system with initial state M_0, E a dynamic environment and $\delta_1, ..., \delta_i, ...$ a sequence of times such that for each δ_i there exist a total function p_i such that $p_i(M_{\delta_i}) = E(\delta_i)$. If the function $p : i \mapsto p_i$ is computable, then the function $\delta : i \mapsto \delta_i$ is computable.*

The last result can be applied naturally to weakly convergent systems (9): the way each adapted state approaches to E is unpredictable, in other words, its *behaviour* changes over different stages unpredictably. Formally:

Corollary 16. *Let $S(M_0, E, t)$ be a weakly converging system with adapted states $M_{\delta_1}, ..., M_{\delta_i}, ...$ and $p_1, ..., p_i, ...$ their respective adapted behaviour. If the mapping $\delta : i \mapsto \delta_i$ is non-descriptively differentiable then the function $p : i \mapsto p_i$ is not computable.*

While asking for totality might look like an arbitrary limitation at first glance, the reader should recall that in weakly convergent systems the program p_i represents an organism, a theory or other computable system that uses M_{δ_i}'s information to predict the behaviour of $E(\delta_i)$, and if this prediction does not process its environment in a sensible time frame then it is hard to argue that it represents an *adapted system* or an *useful theory*.

The intuition behind classifying descriptively differentiable adapted time sequences as *less complex* is better explained by borrowing ideas developed by Bennett and Koppel within the framework of logical depth (Bennett, 1988)

and sophistication (Koppel, 1988), respectively. Their argument states that random strings are as simple as very regular strings, given that there is no complex underlying structure in their minimal descriptions. The intuition that random objects contain no useful information leads us to the same conclusion. And thanks to theorem 14, the states must retain a high degree of randomness for random times.

Logical depth works under the assumption that complex or *deep* natural numbers take a long time to compute from near minimal descriptions. Conversely, random or incompressible strings are shallow since their minimal descriptions must contain the full description *verbatim*. For this work we will use a variation of logical depth called busy beaver logical depth, denoted by $depth_{bb}(x)$.

Sophistication is a measure of *useful information* within a string proposed by Koppel. The idea behind is to divide the description of a string x in two parts: the program that represents the *underlying structure* of the object and the input, which is the random or *structureless* component of the object. This function is denoted by $soph_c(x)$, where c is a natural number representing the significance level.

Now, the images of a mapping $\delta : i \mapsto \delta_i$ already have the form $\delta(i)$, where δ and i represent the structure and the random component respectively. Random strings should hold strongly this structure up to a logarithmic error, which is proven in the next lemma.

Lemma 17. *Let $\delta_1, ..., \delta_i, ...$ be a sequence of different natural numbers and r a natural number. If the function $\delta : i \mapsto \delta_i$ is computable then there exists an infinite subsequence where the sophistication is bounded up to a logarithm of a logarithmic term of their indexes.*

Now, small changes in the significance level of sophistication can have a large impact on the sophistication of a given string. Another possible issue is that the constant proposed at lemma 17 could appear to be large at first (but it becomes comparatively smaller as i grows). A *robust* variation of sophistication called coarse sophistication (Antunes and Fortnow, 2003), incorporates the significance level as a penalty. The definition presented here differs slightly from theirs in order to maintain congruence with the chosen prefix-free universal machine and to avoid negative values. This measure is denoted by $csoph(x)$.

With a similar argument as the one used to prove lemma 17, it is easy to show that coarse sophistication is similarly bounded up to a logarithm of a logarithmic term.

Lemma 18. *Let $\delta_1, ..., \delta_i, ...$ be a sequence of different natural numbers and r a natural number. If the function $\delta : i \mapsto \delta_i$ is computable then there exist an infinite subsequence where the coarse sophistication is bounded up to a logarithm of a logarithmic term.*

Another proposed measure of complexity is Bennett's logical depth, for which the next result follows from a theorem found by Antunes and Fortnow (Antunes and Fortnow,

2003) and lemma 18.

Corollary 19. *Let $\delta_1, ..., \delta_i, ...$ be a sequence of different natural numbers and r a natural number. If the function $\delta : i \mapsto \delta_i$ is computable then there exist an infinite subsequence where the busy beaver logical depth is bounded up to a logarithm of a logarithmic term of their indexes.*

Let us focus on the consequence of lemmas 17 and 18 and corollary 19. Given the relationship established between descriptive time complexity and the corresponding state of a system (theorem 14), these last results imply that either the complexity of the adapted states of a system (using any of the three complexity measures) grows very slowly for an infinity subsequence of times (becoming increasingly common up to probability of 1 (Calude and Stay, 2006)) or the subsequence of adapted times is undecidable. Formally:

Theorem 20. *If $S(M_0, E, t)$ is a weakly converging system with adaptation times $\delta_1, ..., \delta_i, ...$ then there exist a constant c such that, if csoph, $depth_{bb}$ or $soph_c$ show strong OEE that grows faster than $O(\log \log i)$ then the mapping $\delta : i \mapsto \delta_i$ is not computable.*

Technically theorem 20 does not impose undecidability to strong OEE. However, the growing rate that decidability imposes is extremely slow. If we disregard this increasingly insignificant growing rate, we can say that strong open-ended evolution implies undecidability of the adapted states.

Furthermore, by theorem 16, the behaviour and interpretation of the system evolves in an unpredictable way, establishing one path for *emergence*: *a set of rules that cannot be reduced to an initial set of rules*. Recall that for a given weakly converging dynamical system, the sequence of programs p_i represents the behaviour or interpretation of each adapted state M_i. If a system exhibits strong OEE with respect to the complexity measures $soph_c$, $csoph$ or $depth_{bb}$, by corollary 16 and theorem 20 the sequence of behaviours is uncomputable and therefore, irreducible to any function of the form $p : i \mapsto p_i$, even when possessing complete descriptions for the behaviour of the system, its environment and its initial state.

Conclusions

We have presented a formal and general mathematical model for adaptation within the framework of computable dynamical systems. This model exhibits universal properties for all computable dynamical systems, of which Turing machines are a subset.

Among other results, we have given formal definitions of open-ended evolution (OEE) and strong open-ended evolution. We also showed that decidability imposes universal limits to the growth of complexity of computable systems as measured by sophistication, coarse sophistication and busy beaver logical depth. Furthermore, as a direct implication of corollary 16 and theorem 20, undecidability of adapted states and unpredictability of the behaviour of the system

at each state is a requirement for a system to exhibit strong open-ended evolution (up to a $O(\log \log t)$ term) with respect to the complexity measures known as sophistication, coarse sophistication and busy beaver logical depth, establishing a rigorous proof that undecidability and irreducibility of future behaviour is a requirement for the growth of complexity among the class of computable dynamical systems.

References

Adami, C. and Brown, C. T. (1994). Evolutionary learning in the 2D artificial life system avida. In *Proc. Artificial Life IV*, pages 377–? MIT Press.

Antunes, L. and Fortnow, L. (2003). Sophistication revisited. In *ICALP: Annual International Colloquium on Automata, Languages and Programming*.

Auerbach, J. E. and Bongard, J. C. (2014). Environmental influence on the evolution of morphological complexity in machines. *PLoS Computational Biology*, 10(1).

Bedau (1998). Four puzzles about life. *ARTLIFE: Artificial Life*, 4.

Bedau, McCaskill, Packard, Rasmussen, Adami, Green, Ikegami, Kaneko, and Ray (2000). Open problems in artificial life. *ARTLIFE: Artificial Life*, 6.

Bennett, C. H. (1988). Logical depth and physical complexity. In Herken, R., editor, *The Universal Turing Machine: A Half-Century Survey*, pages 227–257. Oxford University Press.

Bournez, O., Graça, D. S., Pouly, A., and Zhong, N. (2013). *The Nature of Computation. Logic, Algorithms, Applications: 9th Conference on Computability in Europe, CiE 2013, Milan, Italy, July 1-5, 2013. Proceedings*, chapter Computability and Computational Complexity of the Evolution of Nonlinear Dynamical Systems, pages 12–21. Springer Berlin Heidelberg, Berlin, Heidelberg.

Calude, C. and Jugensen, H. (2005). Is complexity a source of incompleteness? *ADVAM: Advances in Applied Mathematics*, 35.

Calude, C. S. and Stay, M. (2006). Most programs stop quickly or never halt. *CoRR*, abs/cs/0610153.

Chaitin, G. J. (1982). Algorithmic information theory. In *Encyclopaedia of Statistical Sciences*, volume 1, pages 38–41. Wiley.

Delvenne, J.-C., Kurka, P., and Blondel, V. D. (2006). Decidability and universality in symbolic dynamical systems. *Fundam. Inform*, 74(4):463–490.

Kolmogorov, A. (1965). Three approaches to the quantitative definition of information. *Problems Inform. Transmission*, 1:1–7.

Koppel, M. (1988). Structure. In Herken, R., editor, *The Universal Turing Machine: A Half-Century Survey*, pages 435–452. Oxford University Press.

Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In Bullock, S., Noble, J., Watson, R. A., and Bedau, M. A., editors, *ALIFE*, pages 329–336. MIT Press.

Li, M. and Vitányi, P. (1997). *An introduction to Kolmogorov complexity and its applications*. Springer, 2nd edition.

Lindgren, K. (1992). Evolutionary phenomena in simple dynamics. In Langton, C. G., Taylor, C., Farmer, J. D., and Rasmussen, S., editors, *Artificial Life II*, pages 295–312. Addison-Wesley, Redwood City, CA.

Meiss, J. (2007). Dynamical systems. *Scholarpedia*, 2(2):1629. revision #121407.

Moore, C. (1991). Generalized shifts: Unpredictability and undecidability in dynamical systems. *Nonlinearity*, 4(2):199–230.

Ruiz-Mirazo, K., Peretó, J., and Moreno, A. (2002). A universal definition of life: Autonomy and open-ended evolution. *Origins of life and evolution of the biosphere*, 34(3):323–346.

Soros, L. B. and Stanley, K. O. (2014). Identifying necessary conditions for open-ended evolution through the artificial life world of chromaria. In *Fourteenth International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14)*. MIT Press.

Standish, R. K. (2003). Open-ended artificial evolution. *International Journal of Computational Intelligence and Applications*, 3(2):167–175.

Turing, A. M. (1936). On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42:230–265.

Wolfram, S. (2002). *A New Kind of Science*. Wolfram Media Inc.

Zenil, H., Gershenson, C., Marshall, J. A. R., and Rosenblueth, D. A. (2012). Life as thermodynamic evidence of algorithmic structure in natural environments. *Entropy*, 14(11).

How the Strictness of the Minimal Criterion Impacts Open-Ended Evolution

L. B. Soros[1], Nick Cheney[2], and Kenneth O. Stanley[1]

[1]Department of Computer Science, University of Central Florida, Orlando, FL, USA

[2]Department of Biological Statistics and Computational Biology, Cornell University, Ithaca, NY, USA

lsoros@cs.ucf.edu, nac93@cornell.edu, kstanley@cs.ucf.edu

Abstract

Because the kind of open-ended complexity explosion seen on Earth remains beyond the observed dynamics of current artificial life worlds, it has become critical to isolate and investigate specific factors that may contribute to open-endedness. This paper focuses on one such factor that has previously received little attention in research on open-endedness: the *minimal criterion* (MC) for reproduction. Originally proposed as an enhancement to novelty search, the MC is in effect a different abstraction of evolution than the more conventional competition-focused fitness-based paradigm, instead focusing on the minimal task that must be completed for an organism to be allowed to produce offspring. The MC is interesting for studying open-endedness because in principle its *strictness* (i.e. how hard it is to satisfy) can be varied on a continuum to observe its effects. While in many artificial life worlds the MC strictness is implicit and therefore difficult to vary systematically, in the previously-introduced Chromaria world, the MC is designed to be set explicitly by the experimenter, making possible the systematic study of different levels of MC strictness in this paper. The main result, supported by visual, quantitative, and qualitative observations, is that the strictness of the MC can profoundly affect open-ended dynamics, ultimately deciding between complete stagnation (both with extreme strictness or complete relaxation) and orderly divergence. This result offers a lesson of particular importance to worlds whose MCs are not explicit by exposing an area of sensitivity within open-ended systems that is easy to overlook because of its implicit nature.

Introduction

Artificial life (alife) offers a unique opportunity to replay the tape of life (Gould, 1989) and thereby empirically test hypotheses about phenomena observed in nature. In particular, alife worlds offer an ideal platform for exploring and controlling for mechanisms of interest in the domain of open-ended evolution (OEE) (Channon and Damper, 2000; Lehman and Stanley, 2015; Miconi and Channon, 2005; Ofria and Wilke, 2004; Ray, 1992; Soros and Stanley, 2014; Spector et al., 2007; Yaeger, 1994). Though a universally satisfying definition of OEE has been elusive (Bedau et al., 1998; Channon, 2003, 2006; Juric, 1994; Maley, 1999), open-ended systems should *at minimum* not stagnate or converge. Biological evolution is widely considered an open-

ended process, exhibiting 3.8 billion years of diversification and increasing complexity that continues today (Miconi, 2008). A major challenge to investigating this phenomenon scientifically is of course that experiments on computers must be tractable within much shorter timespans. Given that experimental models necessarily entail such unnatural constraints on time (in addition to space), the need to induce a rate of evolution that achieves natural-seeming dynamics is paramount. However, little is yet known on how to press the delicate levers of evolution to control its speed and open-endedness.

One such lever is the *minimal criterion* (MC) for reproduction, which in effect means the minimal function an organism must perform for its lineage to continue. As an example, on Earth the MC is in effect to survive and physically create a copy of oneself. However, the MC on Earth is really only one of a vast spectrum of possibilities. In e.g. the recent Chromaria alife world (Soros and Stanley, 2014) the MC is simply to plant oneself somewhere that has colors similar to one's own morphology (which then will cause an offspring to be created without any need for an evolved reproductive apparatus). This general concept of MC began to appear in evolutionary computation literature in an algorithm called *minimal criterion novelty search* (MCNS) (Lehman and Stanley, 2010) as a low-level alternative abstraction to fitness-centric evolution. Its first appearance in OEE was in Chromaria.

The attractions of the MC for the purpose of developing a more fundamental understanding of OEE are threefold. First, it allows fitness, which is usually explicitly formalized as a score on a continuum, to become an implicit side effect of a deeper evolutionary principle. Second, by abstracting away the metric of fitness, it enables domains radically different from Earth to be tested in the context of OEE while still maintaining a parallel with natural evolution. Third, the MC becomes just the kind of lever that can be calibrated to alter the overall dynamics of long-term evolution. That is, it is possible to alter its *strictness*. On Earth, this notion of strictness is implicit in the difficulty of surviving to the point of physically making a copy of oneself, but in other conceiv-

able worlds, the difficulty of satisfying the MC can be made *explicit*, allowing the experimenter literally to tune it precisely and observe the consequence. Chromaria offers such an opportunity through its color-matching MC, the strictness of which can be tuned straightforwardly.

Experiments in this paper will reveal that the strictness of the MC in OEE has a similar effect as selection pressure in traditional evolutionary algorithms – the more strict, the more convergent evolution tends to become. At the other end of the spectrum, the MC can allow the world to degrade into chaos when not strict enough. However, the strictness of the MC interestingly does *not* correspond to the proportion of the population that is allowed to reproduce; in fact the MC can be very strict even as many still succeed in satisfying it (and thereby reproducing). At the same time, when set too strictly it becomes a force for convergence and hence an obstacle to OEE. As results in this study will show, the MC is instrumental in striking the right balance between order and chaos – a sweet spot for open-endedness made accessible to investigation by the explicit MC in Chromaria.

The implications of the results speak both to how to steer OEE towards desired dynamics and also to the delicacy of the balance that exists in open-ended systems such as nature where the MC is implicit. Stray too far from this golden balance, and the world can either stagnate or explode into meaningless chaos – a prescription worth considering for any future attempts to achieve OEE.

Background

To understand the significance of MC-based evolution, it is important first to understand the difference between *reproduction* (the passive generation of an offspring, usually as a reward for good performance) and *self-replication* (the active assembly of a copy of oneself). On Earth these processes are conflated because biological self-replication is the sole means of reproduction. However, in alife worlds, an evolutionary algorithm can allow an individual to reproduce even if the individual evolves no mechanism for self-replication. From an algorithmic point of view in such systems, traversing genotype space is all that really matters. Abstracted in this way, the MC can be unfastened from the familiar imperative of self-replication because the MC can in principle be to achieve something *other* than self-replication, which would still then lead to reproduction.

The idea of a MC distinct from self-replication first arose in the context of novelty search (Lehman and Stanley, 2011), an evolutionary algorithm that ignores fitness and instead searches only for behavioral novelty. While pure novelty search proved effective in closed domains with limited possible behavioral trajectories (such as for robots in an enclosed maze), Lehman and Stanley (2010) observed that the method gets lost in spaces that encompass vast regions of degenerate behaviors (such as when robots can wander outside the maze). To address this problem, they proposed adding a

MC that everyone in the population must satisfy by displaying a minimal level of competence (such as staying within the maze) to be even considered for further selection based on novelty. In this way, filtering selection through a MC allows the population to stay focused on the interesting part of the search space.

However, as emphasized by Lehman and Stanley (2010), the MC offers more than just practical utility. More significantly, it is an abstraction of a *fundamental aspect of evolution* that is not captured conceptually by traditional fitness-centric models. The shift in focus from performance-based selection in nature to surviving long enough literally to manufacture a rough copy of oneself transforms the view of evolution from emphasizing optimization instead into a picture of a kind of tinkerer that continually discovers new ways to do the same thing (i.e. satisfy the MC), reminiscent of the concepts of adaptive radiation (Schluter, 2000) and neutral networks (Wagner, 2011). This perspective merits further investigation because it provides a potentially fruitful alternative view of natural evolution largely orthogonal to the usual competition-centric and fitness-inspired interpretations, thereby opening up new avenues for investigation.

Of course, this alternate abstraction need not subsume the fitness-based view. Rather, it provides another angle for investigating evolutionary mechanics as demonstrated by experiments with the recent Chromaria alife world. The original paper introducing Chromaria (Soros and Stanley, 2014) hypothesizes that OEE requires a non-trivial MC and that self-replication is just one of many possibilities for such non-triviality. Chromaria becomes a test for this hypothesis by driving evolution with an explicit MC (planting in a region of similar color to oneself) that is *not* self-replication. Moreover, because the threshold for satisfying this MC (i.e. the degree of similarity required to plant) can be varied, Chromaria provided for the first time a world in which the strictness of the MC can be isolated and directly manipulated to see its effects. Presumably, the difficulty of satisfying the MC would have a profound effect upon the consequent evolutionary dynamics. Yet no studies thus far have explored how to adjust the difficulty of satisfying the MC to strike just the right balance between order and chaos in open-ended alife worlds. The consequent lack of understanding is problematic because, as the remainder of this section will demonstrate, the reproductive mechanisms in many existing alife worlds incorporate a form of MC even if it is not defined as such explicitly.

For example, in Avida (Ofria and Wilke, 2004), individuals are encoded as software programs and the MC, as on Earth, is to self-replicate. However, Avida additionally rewards individuals that evolve to perform certain computational tasks by allowing them to loop through their evolved programs (and thereby potentially reproduce) more frequently than others can. When an individual reproduces in this way, its offspring replaces a preexisting individual
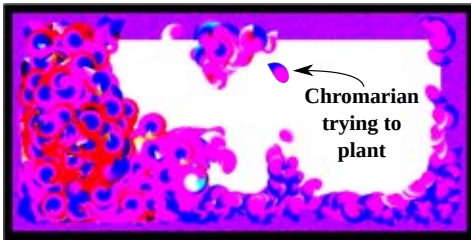
Figure 1: **Chromaria.** The initial Chromarian is born at the center of the world and then must find an appropriate place to plant. Subsequent individuals are born wherever their parents planted. The color-rich borders initially provide the only viable options, but more emerge as Chromarians continue to thrive in the environment.

in the population. Thus the MC of self-replication is made more difficult by introducing the threat of effectively random replacement.

Polyworld (Yaeger, 1994) also implements an Earth-like MC by removing individuals from the population if they fail to survive long enough to mate. In contrast with Avida, Polyworld does not award fitness bonuses. However, Polyworld creatures are given a primitive fighting behavior that can be used to kill other creatures and thereby generate food. In this way, selection incentivizes the evolution of adversarial behaviors, which in turn increases the difficulty of meeting the survival-based MC.

The reproductive mechanisms implemented in Avida and Polyworld are typical of those in other alife worlds, which frequently include similar nature-inspired mechanisms that effectively increase the difficulty of satisfying the MC. However, alife is not bound by nature's constraints and thus admits a variety of alternate MCs that can be more explicitly quantified and manipulated. As a result, alife opens up the opportunity to learn via experimentation the largely unexplored implications of varying MC strictness. The next section describes the alife world of Chromaria, which serves as the domain in this paper for experiments varying the MC strictness.

Chromaria setup

Chromaria[1] (Soros and Stanley, 2014) is an alife world explicitly designed for testing theories about the necessary conditions for OEE (figure 1). In deliberate contrast with many other alife worlds, Chromaria is designed without an explicit notion of competition or relative fitness. Instead, Chromarians qualify for reproduction (via an evolutionary algorithm) by satisfying the unique MC of navigating the world and finding an appropriate place to plant themselves.

Each Chromarian's morphology is a two-dimensional image composed of RGB pixels. To facilitate intelligent nav-

---

[1]Source code is available at http://github.com/lsoros/chromaria.

igation of the world, each Chromarian is equipped with a rectangular visual field consisting of 100 RGB sensors centered at the forefront of the Chromarian's body. Half of these sensors fall underneath the body and the rest extend in front of the creature. The exact resolution of the visual field depends on the creature's morphology; as body length and width increase, the distance between neighboring sensors grows. The visual field is complemented by a heading-sensitive *compass* consisting of 8 pie slice sensors. All sensor values are scaled and then input to an evolved multimodal neural controller (Figure 2). The output layer of this network has four effector nodes corresponding to the Chromarian's rotation (L and R), speed (S), and inclination to plant itself (P). If the P node's activation level exceeds a constant threshold, the Chromarian is immobilized and the function described in Figure 3 calculates a *matching value* quantifying how closely the creature's morphology matches the colors already on the ground at the requested planting location. The attempt succeeds (and the MC is thereby considered met) if the matching value exceeds a configurable *threshold* (τ), which in effect thereby controls the strictness of the MC. Successful planting thus requires evolving a synergistic combination of morphology and behavior.
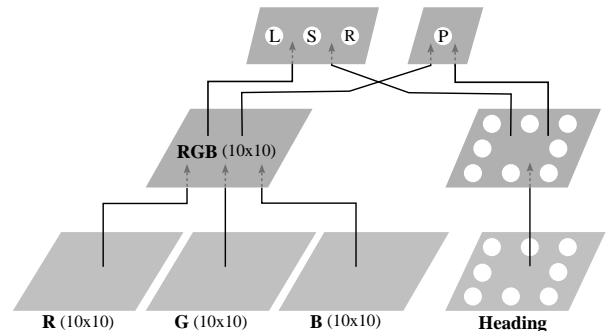
Figure 2: **Behavioral controller.** All sensor values are scaled and then input to an evolved multimodal neural controller. Each plane represents an array of sensors or neurons. Arrows between planes in this schematic denote *sets* of connections between one plane and another. The input layer contains three individual color fields, and an additional set of heading inputs. The four output nodes control movement and planting behaviors (which enable the Chromarian to satisfy the MC). The maximum number of connections in this network (evolved by HyperNEAT) is 30,448.

In the unconventional MC-driven main loop in Chromaria, the Chromarians that have successfully planted most recently are kept in a *parent queue* capped at 100 individuals. The parent queue initially contains only one initial seed Chromarian, which is found by preliminary search (Soros and Stanley, 2014) and thereby guaranteed to successfully
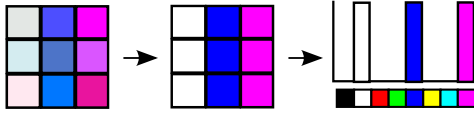
Figure 3: **Boolean MC function in Chromaria.** Each pixel of both the Chromarian's morphology and sensor field (an area of ground overlapping the front of the Chromarian) is placed into one of eight bins: black, white, red, green, blue, yellow, cyan, or magenta. Here, a simple morphology is shown to the left of its binned equivalent and a histogram of the bins. The bins are defined by halving the ranges [0,255] that the R, G, and B component values can take. For instance, any pixel with $R \in [0, \frac{255}{2}]$ (more non-red than red), $G \in [0, \frac{255}{2}]$ (more non-green than green), and $B \in [0, \frac{255}{2}]$ (more non-blue than blue) falls into the black bin because black has values R,G,B = 0,0,0. Once every pixel is binned in this way, color ratios are calculated for each bin by dividing the bin size by the total pixel count. Ratios are recorded for both the morphology and sensor field. The differences between these ratios for each color are summed to get a *matching value*. If this value is less than a configurable threshold τ (effectively the threshold for satisfying the MC), the function is satisfied.

Figure 4: **Morphology-encoding CPPN and decoding process.** The CPPN on the left is decoded by iteratively activating the network with pairs of polar coordinates r and θ as input and using the resulting outputs to draw the morphology pixel by pixel. Upon activation, the CPPN returns an r_{max} for each value of θ, which determines how far the perimeter of the Chromarian's body extends at that angle. The CPPN is then activated again for every r on the interior of this border to get the corresponding RGB values. In this way the CPPN determines both the shape (via the r_{max} output) and internal color (via the RGB outputs) of the Chromarian. These characteristics ultimately determine where the Chromarian can successfully plant and thereby satisfy the MC.

plant. The initial seed individual is born at the center of the world and navigates to plant on the color-rich border. Subsequent offspring are born on top of their parents and then must similarly move away from their spawn point to find an appropriate place to plant. To prevent individuals who attempt trivially to plant without moving from reproducing, planting attempts made during the first 25 ticks of an individual's life are automatically invalidated. If an individual either makes an invalid planting attempt or fails to plant altogether, the Chromarian is removed from the world and does not generate any children. However, if a Chromarian succeeds in moving to an appropriate location and then requests to plant, its body remains frozen and thereby supercedes all other pixels at that location. Its offspring is then generated (because anyone who satisfies the MC is given an offspring) based on the reproductive mechanics of HyperNEAT (Stanley et al., 2009). Accordingly, the genotype used in this system is a compositional pattern producing network (CPPN; Stanley 2007), an indirect encoding that generates patterns with regularities seen in nature such as symmetry, repetition, and repetition with variation. Separate CPPNs are used to encode the morphology (Figure 4) and neural controller. Parameters for evolution are included with the source code.

Note that reproductive dynamics in Chromaria are unlike those in many other alife worlds. To both allow the steady state evolutionary algorithm to explore multiple lineages simultaneously and to guarantee that *all* Chromarians who successfully plant eventually get to reproduce, offspring who satisfy the MC are placed at the end of the parent queue and only reproduce once they reach the front of the queue. (The oldest parent in the queue is removed to make room for the new offspring if the queue is at maximum capacity.) After reproducing, if it is not the oldest, the Chromarian at the front of the queue is sent to the back again. Thus the individual active in the world is not always the child of the individual that was active just before it. Though potentially unintuitive, this process forces the system to allow every preexisting member of the population to reproduce before a newcomer. As a result, the explicit competition so central to many alife worlds is intentionally absent from Chromaria. Note that if offspring of enough parents in the queue fail to plant, some parents may receive more than one opportunity to reproduce when they advance again to the front of the line.

Chromaria is chosen as the domain for experiments in this paper precisely because it allows the MC to be studied in isolation from competition and other selective pressures (everyone who satisfies the MC is guaranteed an offspring). Additionally, the difficulty of satisfying the MC is easily adjusted by simply setting a different τ for the planting function described in Figure 3. An interesting question that will be clarified by these experiments is what features of open-endedness can be achieved when more conventional competitive mechanisms are absent. However, the hope is that the results will inform the design of systems that *do* involve more intricate evolutionary pressures.

Experiment

One aim of this study is to develop an intuition for how the MC affects the balance of order and chaos in potentially open-ended evolutionary systems. For this reason, experimental runs are carried out through 350,000 reproductions, each lasting 5 to 7 days (wall clock time). Fifty runs were performed in total, each starting from the same initial seed and varying only τ. Twenty baseline runs were performed with a **moderate** τ of 87.5%, which is the same matching threshold implemented in previous experiments by Soros and Stanley (2014). Additionally, the MC was made more **strict** by increasing τ to 95% and made more **forgiving** by decreasing it to 75%. Ten runs were performed with each of these thresholds.

A related aim of this study is to investigate one of the hypothesized necessary conditions for OEE proposed by Soros and Stanley (2014). Though artificial life provides a promising platform for answering the question of exactly what is necessary for OEE, surprisingly few empirical studies have been performed to test the validity of the few frameworks proposed thus far (i.e. by Waddington (1969) and Taylor (2004)). According to the hypothesis that the MC is *necessary* for OEE (Soros and Stanley, 2014), evolving systems should become degenerate if every individual is allowed to reproduce. This hypothesis in effect addresses the extreme case wherein the MC is so relaxed that there is **no MC (the control)**. To this end, an additional set of ten control runs are performed wherein every individual generates an offspring regardless of what it does during its lifetime. Note that this control setup is slightly more complicated than simply setting τ to 0% because planting attempts during the first 25 ticks of an individual's life are normally invalidated, but this restriction is removed in the control case. Individuals that *never* ask to plant also generate offspring (after timing out) in the control case, though only the bodies of Chromarians who actively ask to plant remain visible in the world after their lifetime ends.

Results

Chromaria's visual design allows evolutionary dynamics to be observed in an intuitive, engaging manner. Harnessing this capability, sequences of screenshots up to 350,000 reproductions demonstrate typical effects of varying τ to produce different MC strictness levels (Figure 5). The story told by these sequences illustrates the profound effect of MC strictness on OEE. At one end of the spectrum, when the MC is strict, the world only slightly changes over hundreds of thousands of reproductions. Interestingly, the extreme opposite end of the spectrum, when there is no MC at all, while initially chaotic, also ends up descending into stagnation because the entire breeding population eventually loses the ability even to request to plant. It is only at the middle ranges that consistent and coherent change (both in terms of morphology and behavior) is readily observable.

Of course, screenshots offer only a static portrait of life in Chromaria and cannot capture the diversity and complexity of individual Chromarians' behaviors. For this reason, videos (and additional screenshots) are available at http://eplex.cs.ucf.edu/chromaria-resources/. Observing these dynamic worlds reveals that the most interesting behaviors are expressed when the environment contains a variety of simple RGB signals. When the MC is strict, little deviation from the simple dichromatic initial seed is tolerated, and as a result the world becomes filled with strong magenta, blue, and white signals. Individuals evolve behaviors such as turning left when a magenta/blue edge appears and then planting when a patch of whiteness (such as at the frontier created by new planters) is encountered. When the MC is moderate, environmental complexity increases as a result of Chromarians with more complex morphologies successfully planting. However, the color-based signals in the environment become harder to differentiate and individuals begin to rely on heading (which provides a more reliable signal) to affect planting behaviors. This phenomenon is readily observed in the later stages of the runs with moderate MC; though some individuals visibly change direction based on environmental signals, others ignore the RGB inputs and simply move in smooth arcs and then plant once a desired heading has been reached. Runs with the forgiving MC exhibit the widest diversity of behaviors, reflecting the clear yet diverse RGB signals that emerge in the environment as individuals successfully plant. Finally, when the MC is completely absent, individuals tend to follow simple trajectories with no inherent purpose. Eventually, planting behaviors end up disappearing completely in this case, and senselessly crashing into walls becomes a popular behavior. Recall that the second goal of this experiment is to test the hypothesis that a MC is necessary for OEE. If this hypothesis is true, then evolution should stagnate when the MC is absent. In fact such stagnation (as depicted in Figure 5) *does* occur in every control run.

The qualitative view is strengthened by complementary quantitative analysis. Of course, the quantification of open-endedness is a contentious topic, reflecting the general lack of consensus regarding the phenomenon's salient features. Activity statistics (Bedau et al., 1998; Channon, 2003), which measure the persistence of advantageous genotypes over the course of evolution, present one possibility for quantification. However, alternate metrics are devised for the experiments in this paper so that the control runs (which completely eliminate genotypic advantage by giving every individual an offspring) can be included in comparisons.

The first metric, which is an application of the graph theoretic notion of *connected components* (Hopcroft and Tarjan, 1973), interprets the RGB phenotype space as a three-dimensional cube-shaped graph (Figure 6). Every 250 reproductions, each individual in the parent queue is placed into one of 216 bins based on the R, G, and B ratios in its mor-
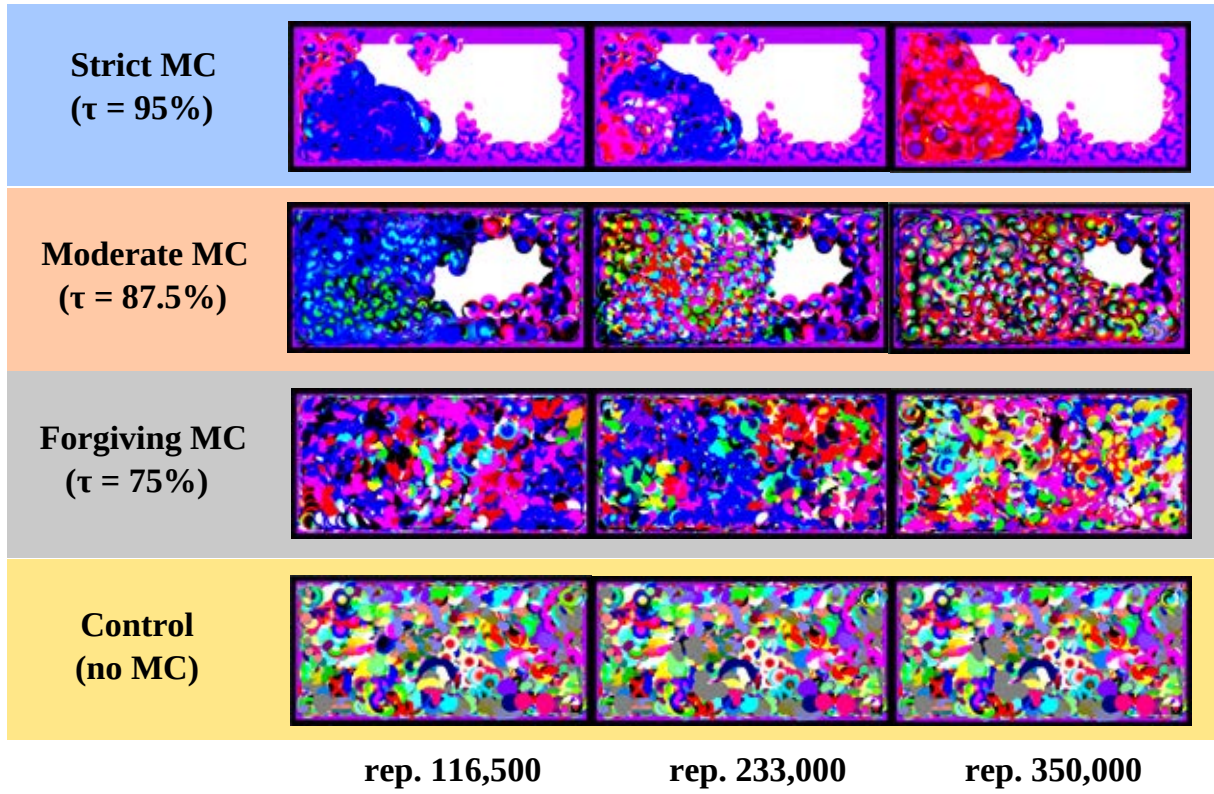
Figure 5: **Representative worlds over time.** Worlds with different MC strictness are depicted after the specified numbers of reproductions have occurred. The variable τ quantifies the difficulty of the MC (detailed in the Experiment section). When the MC is strict, evolution exhibits little progression and the breeding population lacks diversity (an undesirable result for OEE). As the MC is relaxed, diversity increases but stability eventually plunges into chaos (eventually leading back to stagnation). Videos demonstrating the complexity of behavior in additional runs are available at http://eplex.cs.ucf.edu/chromaria-resources/.

phology. The individual bins can then be treated as nodes in a graph, with connections existing between adjacent nodes. Evolutionary divergence (or lack thereof) can then be approximated by the average number of *disjoint components* ($\overline{dc}$) in the RGB cube at any given time in the run. Interestingly, the minimal value of 1 therefore jointly indicates that evolution is *either* maximally ordered (when few bins are filled) or maximally chaotic (when all bins are filled). Between these extremes, $\overline{dc}$ increases as evolution diverges. However, this metric (which indicates divergence only every 250 reproductions) does not tell the full story because it does not reveal how much of the space was eventually explored over the *entire* run. For this reason, the average total number of bins filled over the course of an entire run ($\overline{bf}$) is also recorded for each MC level.

Figure 7 depicts $\overline{dc}$ over time for each run. As expected, when the MC is strict ($\tau = 95\%$) the average number of disjoint components and number of bins filled per run are both low ($\overline{dc} = 1.34$, $\sigma = 0.59$; $\overline{bf} = 51/216$), indicating limited evolutionary divergence. Interestingly, when the MC is relaxed to the more moderate baseline value ($\tau = 87.5\%$), $\overline{dc}$ increases only slightly to 2.24 ($\sigma = 0.42$), but $\overline{bf}$ sharply

Figure 6: **Discretized RGB space.** Every Chromarian in the parent queue is placed into 1 of 216 bins based on the R, G, and B ratios in its morphology. The number of connected components in the color space approximates the amount of correlation in the parent queue. When all bins are filled (as is the case when the system is completely chaotic) as shown above, the number of connected components is 1. It is also 1 when the system is completely converged.
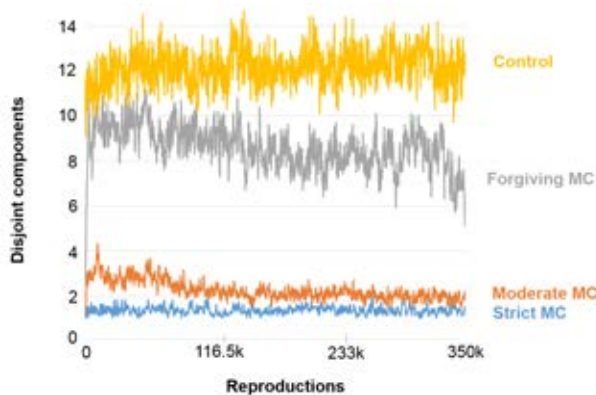
Figure 7: **Average number of disjoint components in the parent queue over time.** A minimal value of 1 indicates that evolution is either maximally ordered (when few bins are filled) or maximally chaotic (when all bins are filled). However, *high* values may also indicate a tendency towards chaos. For this experiment, samples are taken every 250 reproductions. Student's t-tests with p-values $p < 0.05$ indicate significant differences between strict versus moderate runs for 97.21% of parent queue samples, for 100% of the samples between moderate versus forgiving runs, and for 98.64% of samples between forgiving and control (no MC) runs.
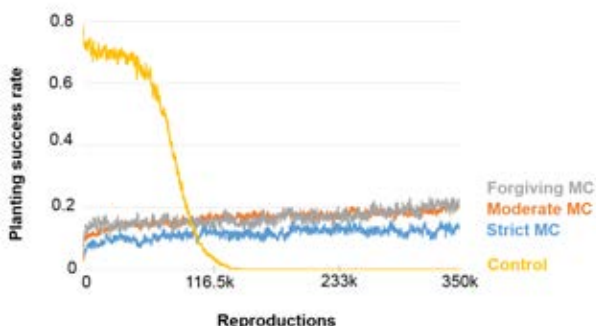
Figure 8: **Average planting success rates over time.** The number of successful planters increases steadily and eventually approximately doubles over the course of the non-control runs, indicating that evolution reliably discovers nontrivial behaviors. However, when there is no MC (the control) evolution degenerates so rapidly into triviality that evolution halts altogether. This result confirms the hypothesis that a minimal criterion is necessary for OEE.

increases to 178.5/216 ($\sigma = 17.56$), signifying a similar degree of order as for the strict MC, but much faster traversal of phenotype space over time. As the MC becomes easier to satisfy, the runs display an increased tendency towards divergence and chaos. In the forgiving case ($\tau=75\%$), $\overline{dc}$ increases dramatically to 8.59 ($\sigma = 0.97$) and eventually fills

all 216 bins. The $\overline{dc}$ measure remains high (12.16, $\sigma = 0.79$) and all bins are still filled when the MC is completely absent, but recall that the world nevertheless actually eventually stops changing entirely without the MC because of the eventual loss of the ability to plant.

The catastrophic effects of removing the MC are clearly reflected also in the immediate and inevitable decline in planting success rates (Figure 8). In contrast, when there is an MC of *any* strictness, planting success rates steadily *increase* over time, suggesting a nontrivial evolutionary process leading to gradual behavioral improvements.

Discussion

Combining the number of connected components with the overall number of bins filled paints a quantitative picture of evolution's trajectory. When the MC is too strict, evolution proceeds so slowly as to become stagnant. Unsurprisingly, there is little variation between runs in terms of color space coverage because search does not progress far beyond the initial magenta and blue subspace. The MC in this case encourages genetic homogeneity and leads to an effective equilibrium. At the other extreme, when the minimal criterion is absent (the control case), the system initially exists so far from equilibrium that it verges on chaos. While such a system may avoid attractors and thus maximize coverage of the search space, it may also lack the qualities of cohesion and correlation that differentiate evolution from random search. Most problematic, though, is the proliferation of degenerate behaviors because of the lack of pressure to satisfy the MC. The lack of new planters even in the control when τ is 0 (evidenced by eventual planting success rates of 0) indicates that *nobody is requesting to plant* even though any such request would be granted regardless of the colors in the world. This result may at first seem surprising, but in fact strongly supports the hypothesis that maintaining a minimal level of behavioral complexity is necessary for OEE.

These experiments illustrate the complex manner in which the strictness of the explicit MC impacts evolution in Chromaria. However, the results are important more broadly for their potential to generalize to any open-ended alife world that limits which individuals are allowed to reproduce. In other words, modifying the intrinsic strictness of the *implicit* MC in other worlds may have similarly dramatic implications for their degree of open-endedness. This issue is nontrivial because the MC prunes the space being explored by evolution. If there were no MC on Earth, for example, every organism would reproduce regardless of its viability. As a result, degeneracy would thrive whereas otherwise it could not. If the MC were too difficult (in this case, anything requiring functionality beyond that of a single cell), foundational building blocks would be discarded. Thus setting the wrong MC can have disastrous implications for a system's ability to uncover the maximal amount of interestingness in a search space. While a designer may be able to

compensate for an overly strict MC by increasing run time if the run time is capped, it may be desirable to relax the MC through whatever means available to increase coverage of the search space in the allotted time.

An emphasis on optimization in evolutionary computation has influenced the primacy of *competition* in the design of alife worlds. Unlike competition-based selection, the MC admits innovations that offer no discernible benefit for an individual (which would otherwise have been discarded by optimization-oriented evolution). For this reason, the MC allows evolution to maximize its potential as a creative force.

However, competition is not the only overly restrictive mechanism. Any selective force that arbitrarily imposes a strict limit on reproduction (i.e. by only allowing a certain percentage of the population to generate offspring) will suffer the same inhibited creative output, including on Earth. Many mechanisms implemented in artificial life systems can be reduced to a MC-based interpretation in this way. By abstracting away the system-specific details and working at the most general level, we can develop a model of open-ended evolution with maximum applicability.

Conclusion

Life strikes a delicate balance between order and chaos. In alife worlds, this balance is moderated at least in part by an often-implicitly-implemented minimal criterion (MC). As results in this paper have shown, tuning this mechanism incorrectly can have disastrous implications for a system's open-endedness. However, if configured correctly, the MC can create a dynamic environment in which evolved life flourishes.

Acknowledgements

References

Bedau, M. A., Snyder, E., and Packard, N. H. (1998). A classification of longterm evolutionary dynamics. In *Proc. of Artificial Life VI*, pages 189–198, Cambridge, MA. MIT Press.

Channon, A. (2003). Improving and still passing the ALife test: Component-normalised activity statistics classify evolution in geb as unbounded. In *Proc. of Artificial Life VIII*, pages 173–181, Cambridge, MA. MIT Press.

Channon, A. (2006). Unbounded evolutionary dynamics in a system of agents that actively process and transform their environment. *Genetic Programming and Evolvable Machines*, 7(3):253–281.

Channon, A. D. and Damper, R. I. (2000). Towards the evolutionary emergence of increasingly complex advantageous behaviours. *International Journal of Systems Science*, 31(7):843–860.

Gould, S. J. (1989). *Wonderful Life: The Burgess Shale and the Nature of History*. W. W. Norton and Company.

Hopcroft, J. and Tarjan, R. (1973). Algorithm 447: Efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378.

Juric, M. (1994). An anti-adaptationist approach to genetic algorithms. In *Proc. of First IEEE Conf. on Evolutionary Computation*, volume 2, pages 619–623. IEEE.

Lehman, J. and Stanley, K. O. (2010). Revising the evolutionary computation abstraction: minimal criteria novelty search. In *Proc. of the 12th annual conf. on Genetic and evolutionary computation*, pages 103–110. ACM.

Lehman, J. and Stanley, K. O. (2011). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223.

Lehman, J. and Stanley, K. O. (2015). Investigating biological assumptions through radical reimplementation. *Artificial Life*, 21(1):21–46.

Maley, C. C. (1999). Four steps toward open-ended evolution. In *Proc. of the Genetic and Evolutionary Computation Conf.*, volume 2, pages 1336–1343. Morgan Kaufmann.

Miconi, T. (2008). *The road to everywhere: Evolution, complexity and progress in natural and artificial systems*. PhD thesis, University of Birmingham.

Miconi, T. and Channon, A. (2005). A virtual creatures model for studies in artificial evolution. In *The 2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 565–572. IEEE.

Ofria, C. and Wilke, C. O. (2004). Avida: A software platform for research in computational evolutionary biology. *Artificial life*, 10(2):191–229.

Ray, T. S. (1992). An approach to the synthesis of life. In *Proc. of Artificial Life II*, pages 371–408.

Schluter, D. (2000). *The Ecology of Adaptive Radiation*. Oxford University Press.

Soros, L. B. and Stanley, K. O. (2014). Identifying minimal conditions for open-ended evolution through the artificial life world of chromaria. In *Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems*, pages 793–800, Cambridge, MA. MIT Press.

Spector, L., Klein, J., and Feinstein, M. (2007). Division blocks and the open-ended evolution of development, form, and behavior. In *Proc. of the 9th annual conf. on Genetic and evolutionary computation*, pages 316–323. ACM.

Stanley, K. O. (2007). Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines Special Issue on Developmental Systems*, 8(2):131–162.

Stanley, K. O., D'Ambrosio, D. B., and Gauci, J. (2009). A hypercube-based indirect encoding for evolving large-scale neural networks. *Artificial Life*, 15(2):185–212.

Taylor, T. (2004). Niche construction and the evolution of complexity. In *Artificial Life IX: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Artificial Life*, pages 375–380, Cambridge, MA. MIT Press.

Waddington, C. H. (1969). Paradigm for an evolutionary process. In Waddington, C., editor, *Towards a Theoretical Biology, Volume 2*. Edinburgh University Press, Edinburgh, Scotland.

Wagner, A. (2011). *The Origins of Evolutionary Innovations: A Theory of Transformative Change in Living Systems*. Oxford University Press.

Yaeger, L. (1994). PolyWorld: Life in a new context. *Proc. Artificial Life*, 3:263–263.

Evolution of Heterogeneous Cellular Automata in Fluctuating Environments

David Medernach[1,*], Simon Carrignon[2,3], René Doursat[4], Taras Kowaliw, Jeannie Fitzgerald[1] and Conor Ryan[1]

[1]Biocomputing and Developmental Systems (BDS), University of Limerick, Ireland
[2]Barcelona Supercomputing Center (BSC), Catalonia, Spain
[3]ICREA-Complex Systems Lab, Universitat Pompeu Fabra, Barcelona, Catalonia, Spain
[4]Informatics Research Centre (IRC), Manchester Metropolitan University, UK
[*]david.medernach@ul.ie

Abstract

The importance of environmental fluctuations in the evolution of living organisms by natural selection has been widely noted by biologists and linked to many important characteristics of life such as modularity, plasticity, genotype size, mutation rate, learning, or epigenetic adaptations. In artificial-life simulations, however, environmental fluctuations are usually seen as a nuisance rather than an essential characteristic of evolution. HetCA is a heterogeneous cellular automata characterized by its ability to generate open-ended long-term evolution and "evolutionary progress". In this paper, we propose to measure the impact of different types of environmental fluctuations in HetCA. Our results indicate that environmental changes induce mechanisms analogous to epigenetic adaptation or multilevel selection. This is particularly prevalent in two of the tested fluctuation schemes, which involve a round-robin inhibition of certain cell types, where phenotypic selection seems to occur.

Introduction

In natural evolution, environmental changes may include cyclic events such as seasonal changes and the daily alternation of light and darkness, occasional changes such as the appearance of new predators and the potential for new food sources, or more radical modifications such as environmental stresses induced by climate transitions.

Since the work of Levins (1968) on evolution in changing environments and more recent attempts to integrate "epigenetic inheritance systems" (EIS) (Heard and Martienssen, 2014), developmental or "evo-devo" processes (Müller, 2007) and "niche construction" effects (Laland et al., 2016), emphasis has been put on the importance and impact of *changes* on the course of the evolution of living organisms by natural selection. Through such works, environmental fluctuations have been linked to many central properties and mechanisms of evolution, among which the most discussed examples are modularity, plasticity (West-Eberhard, 2005), genome size, mutation rates, and evolvability.

For Jablonka et al. (2014), changing environments unmask variations in the capacity of individuals to make adjustments to new conditions, therefore promoting plasticity and multilevel selection. These authors contend that "For a lineage in a constantly changing environment, switching among several alternative heritable states was probably an advantage. While cells in one state survived in one set of conditions, those in other states did better in different circumstances." (*ibid.*, p. 318). In the same line of thought, continually varying or cyclic conditions might also explain the origin of EIS, as "epigenetic mutations" are more reversible and occur more frequently than genetic ones. To illustrate this notion, Lachmann and Jablonka (1996) modeled the effects of oscillating variations, such as seasonal or daily cycles, on phenotypical inheritance. Their model predicts correctly that when the environmental cycle is longer than the reproductive cycle, while remaining relatively short otherwise, heritable variations produced by non-DNA inheritance systems are likely to be observed.

In parallel to biological research, a number of studies in artificial life, especially evolutionary robotics (Floreano and Urzelai, 2000), have also investigated environmental variations, some of them explicitly defining the environment as a driving evolutionary force (Bredeche and Montanier, 2012). Others, such as Lipson et al. (2002), showed a correlation between the modularity and the rate of change of external resources, while Yu (2007) observed that populations exploit neutrality to cope with environmental fluctuations and can evolve a type of evolvability under two alternating objective functions. Both of these simulation works relied on genetic programming (GP) and explicit fitness functions.

We wish to study the effects of such fluctuations in a model closer to the living world without using an explicit objective function, and determine if these fluctuations promote phenotypic selection over genotypic selection. To that goal, we propose an open-ended experimental setup allowing us to systematically and quantitatively measure the influence of cyclic environmental fluctuations on the course of the evolution of cellular automata (CA). We show that such fluctuations lead to the emergence of processes similar to those exhibited by EIS.

The paper is organized as follows. First the general mechanisms of our Heterogeneous Cellular Automata (HetCA) model are explained. Then, the implementation of environ-

mental fluctuations in HetCA is described. Next, we specify the computational setup used to study environmental fluctuations. This is followed by a report on the experimental results and a discussion of their implications. Finally, we propose a qualitative analysis and a conclusion.

The HetCA Model

HetCA (Medernach et al., 2013) is based on classical two-dimensional CA with several additional features: cells follow a heterogeneous transition function, i.e. one which depends on their location, inspired by linear genetic programming (LGP); they can also fall into special *decay* and *quiescent* states; and there is a notion of genetic transfer of transition functions (i.e. genotypes) between adjacent cells. Decay and quiescent cells do not possess a genotype; all other cells do, and are called *living*. There are 5 different living states: quiescent cells can acquire a genotype from any nearby living cell and therefore become living in turn; decay cells cannot, but become quiescent after a number of consecutive iterations comprised between 375 and 1,875 after decay (Fig. 1). Living cells always automatically turn into decay after 7 consecutive iterations spent in any one or several living states (tracked by an "age" counter). Further details about the HetCA model can be found in (Medernach et al., 2013).

We showed that HetCA could exhibit long-term phenotypic dynamics, a high variance over very long runs, greater behavioral diversity than classical CA, and "evolutionary progress" (Shanahan, 2012) on three criteria: robustness, size and density of the genotype (Medernach et al., 2015).

Finally, while there is a lasting debate over the units of selection in evolutionary biology since the origins of the field, including genotype selection, phenotype selection, epigenetic selection, behavioral selection, multilevel selection, group selection, and so on (Lloyd, 2012; Okasha, 2006), several of them are potentially included in HetCA. There is genotypic selection of the transition rules, but also phenotypic selection of cell groups able to replicate patterns, such as the ones found in the Game of Life. This point is important when one is interested in environmental fluctuations because, as mentioned in the introduction, we anticipate that the existence of frequent environmental fluctuations will promote phenotypic selection over genotypic selection.

Experimental Setup

As in the previous version of HetCA (Medernach et al., 2013), the genotype of an individual consists of its transition rules encoded in a custom CA-LGP program using the function set listed in Table 1. Such a program maps the space of neighborhood states to a new cell state, while providing an evolvable representation framework based on an alphabet of elementary functions. Individual genotypes are modified by micro-mutations (change in one component of a statement)
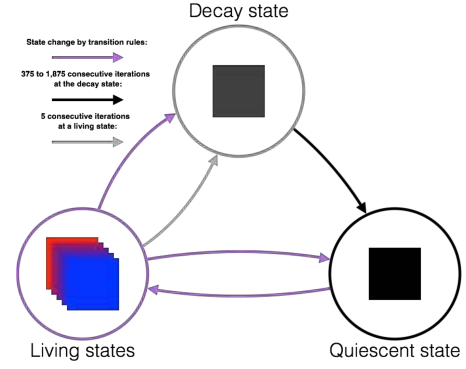
Figure 1: A cell's life cycle between states.

Table 1: Function set.

operator	action on inputs (x, y)
abs	$\lvert x \rvert$
plus	$x + y$
delta	1, if $\lvert x - y \rvert < 1/10^4$; 0 o.w.
dist	$\lvert x - y \rvert$
inv	$1 - x$
inv2	safeDiv$(1, x)$
magPlus	$\lvert x + y \rvert$
max	$\max\{x, y\}$
min	$\min\{x, y\}$
safeDiv	x/y if $\lvert y \rvert > 1/10^4$; 1 o.w.
safePow	x^y, if defined; 1 o.w.
thresh	1, if $x > y$; 0 o.w.
times	xy
zero	1, if $\lvert x \rvert < 1/10^4$; 0 o.w.

and macro-mutations (addition or removal of an entire statement) of the corresponding CA-LGP programs.

Previously, the new genotype g of a living cell c during genetic transfer was chosen randomly among candidate genotypes according to a uniform distribution. To be candidate, a genotype had to come from a living cell in c's immediate neighborhood (von Neumann). In the present study, to introduce environmental variations we vary the likelihood of propagation of a genotype according to its cell state $s(c)$. In this new setup, the probability $P(c)$ of a candidate genotype to be selected becomes $P(c) = K(s(c))/\sum_{i=1}^{n} K(s(c_i))$, where $K(s)$ is the state distribution and n the number of neighboring candidate genotypes, i.e. 4. Thus, an environment E is characterized by the propagation probabilities of the 5 possible living states: $E = \{K(s_1), ..., K(s_5)\}$. To mimic environmental fluctuations we initialize the simulation with $K(s_k) = 1$ for all $k \in [1, 5]$, then regularly modify those values every f iterations starting from iteration 3,000 of the CA. We introduce three types of environmental fluctuations (Table 2):

Short-cycle fluctuations (ScF) consist of alternating between two opposite environments, $\{0, 0, 1, 1, 1\}$ and $\{1, 1, 1, 0, 0\}$, every f iterations of the CA. We set $f = 100$ to remain within the range of frequency described by Lipson et al. (2002) and Yu (2007). Here we consider that a successful reproductive cycle for a cell involves passing through the

Table 2: Stable and fluctuating environments.

Name	Short Name	Cycles	Transitions	Environment list: propagation probabilities $E = \{K(s_1), ..., K(s_5)\}$ of the living types
Stable environment	(SE)	NA	NA	$\{1, 1, 1, 1, 1\}$
Short-cycle fluctuations	(ScF)	100	1	$\{0, 0, 1, 1, 1\}, \{1, 1, 1, 0, 0\}$
Light fluctuations	(LF)	5,000	1	$\{1, 1, 1, 1, 0\}, \{1, 1, 1, 0, 1\}, \{1, 1, 0, 1, 1\}, \{1, 0, 1, 1, 1\}, \{0, 1, 1, 1, 1\}, \{1, 1, 1, 1, 1\}$
Strong fluctuations	(SF)	5,000	1	$\{0, 0, 1, 1, 1\}, \{1, 1, 1, 0, 0\}, \{0, 1, 0, 1, 1\}, \{1, 0, 1, 1, 0\}, \{0, 1, 1, 0, 1\}, \{1, 1, 0, 1, 0\},$ $\{1, 0, 1, 0, 1\}, \{0, 1, 1, 1, 0\}, \{1, 0, 0, 1, 1\}, \{1, 1, 0, 0, 1\}, \{1, 1, 1, 1, 1\}$

Table 3: HetCA parameters.

Parameter	Value
Number of living states	5
Successive living iterations before decay	7
Number of iterations during decay	375 to 1,875
Direct transition to decay	enabled
Size of the grid	500×500
Grid boundaries	toroidal grid
Transition Rule (TR)	CA-LGP
Maximum TR size	50 statmts
Genotype copy neighborhood	v. Neum. (4)
Transition rule neighborhood	Moore (8)

quiescent state. This should take between 2 iterations (alternating between quiescent and living) and 7 iterations (after which a living cell decays and can no longer receive a genotype for a long period of time).

Light fluctuations (LF) consist of alternating between 6 different environments every $f = 5,000$ iterations. The first 5 environments each prohibit a different living state from spreading its genotype; the last one gives an equal chance to all living states.

Strong fluctuations (SF) consist of alternating between 11 environments every $f = 5,000$ iterations. The first 10 environments each prohibit a different pair of living states from spreading their genotypes; the last one gives an equal chance to all pairs.

The rationale behind ScF is their analogy with circadian rhythms in certain bacteria. The idea is to mimic the highly regular cycles during which these organisms have enough time to reproduce repeatedly. LF, by contrast, are more similar to seasonal fluctuations, while SF resemble ecological crises. However, owing to the variety of both biological temporal rhythms and reproductive cycles, the relevance of these analogies remains limited.

Simulations

For each one of the three types of environmental fluctuations and the stable (non-fluctuating) environment (SE) we performed 50 simulations. This produced a total of $4 \times 50 = 200$ runs. Each cell of the CA was initialized in a random state, then each cell in one of the 5 living states was initialized with an individual randomly generated genotype. Each run lasted 500,000 iterations under the parameters listed in Table 3.

Genotype Size

We used the number of program statements n_{prog} as a measure of genotype size and computed the average size of all current genotypes of a run every 2,500 iterations. We then reported the average and standard error of the mean (SEM) among all 50 runs sharing the same settings.

Phenotype Comparison

If environmental changes led to the emergence of phenotypic selections (similar to the EIS) using easily reversible phenotypic mutations (Jablonka et al., 2014), then phenotypes from different individuals of the same lineage observed while environmental conditions are similar should stay relatively close, even though individuals from their lineage evolved in other environmental conditions between these measures. By contrast, if the adaptation to each environmental change was done exclusively through the selection of classical, irreversible genotypic mutations, these phenotypes should be quite different, despite the potential evolutionary convergence. We developed a metric to measure phenotypic proximity between two iterations of the CA. To do this we simply used the distributions of living cells over the living states. Thus the phenotypic difference σ between two iterations t_1 and t_2 was calculated as follows:

$$\sigma(t_1, t_2) = \sum_{k=1}^{5} \left| \frac{N(s_k, t_1)}{N(t_1)} - \frac{N(s_k, t_2)}{N(t_2)} \right| \qquad (1)$$

where $N(s_k, t)$ is the number of cells in living state s_k at iteration t and $N(t) = \sum_{k=1}^{5} N(s_k, t)$ is the total number of living cells at t.

Every 5,000 iterations of the CA we performed two phenotypic comparisons between the current iteration $t_1 = t$ and an iteration in the past, $t_2 = t - \Delta t$. In one scenario, the temporal distance Δt was a multiple of the periodicity f, so that we compared two similar environments: $E(t_1) = E(t_2)$. We chose $\Delta t = 60,000$ in the SE, ScF and LF cases and 55,000 in the SF case. In another scenario, we introduced an additional single-period shift such that we compared two dissimilar environments: $E(t_1) \neq E(t_2)$ but $E(t_1) = E(t_2 + f)$. Here Δt was respectively equal to 60,100, 65,000 and 60,000 in the ScF, LF and SF cases.

Diversity

We used the "true diversity index" of order two (Jost, 2006) to measure the phenotypic and genotypic diversity at every

iteration t of the CA:

$$^2D_p(t) = 1/\sum_{k=1}^{5} \frac{N^2(s_k,t)}{N^2(t)}, \quad ^2D_g(t) = 1/\sum_{g=1}^{G(t)} \frac{N^2(g,t)}{N^2(t)},$$
(2)

where $G(t)$ is the number of distinct genotypes at iteration t and $N(g,t)$ is the number of cells sharing genotype g at iteration t. Note that $N(t) = \sum_{k=1}^{5} N(s_k,t) = \sum_{g=1}^{G(t)} N(g,t)$.

Homogeneous Test

We also collected the *most common genotype* (MCG), i.e. the most frequently occurring one, starting at iteration 2,500,[1] then 102,500 and again about every 100,000 steps until iteration 500,000, thus creating 6 sampling points. All MCGs collected in that way were exported and tested alone in homogeneous conditions, i.e. where all cells were initialized with that genotype and no mutations could occur. Since each collected MCG from any one of the four fluctuating environments was cross-tested again in all four environments, we performed a total of $4 \times 50 \times 6 \times 4 = 4800$ runs. The maximum duration of these runs was set to 60,000 iterations. Sometimes the genotype was not adapted to the environment and all living cells went extinct (i.e. turned into the decay or quiescent state) before the end of the simulation. We considered a homogeneity test to be successful if living cells did not all go extinct before 60,000 iterations. In the Results section, we report the success rate of those simulations, along with statistics on the last iterations of the failed runs.

Phenotypic Disturbance in the Homogeneous Test

In HetCA, to survive on the long term genotypes must regularly "release" cells by transforming them into quiescent cells without genotypes. This generates patterns and cycles easy to observe in homogeneous simulations where a single genotype is tested (Fig. 2a). It is also observable, although with greater difficulty, in standard heterogeneous HetCA simulations (Fig. 2b). This is why, to characterize phenotypes we found it useful to measure these cycles as well as their irregularities. At every iteration $t > 8$ of the homogeneous genotype test, we compared the sequence of states $s(c,t)$ and $s(c,t-1)$ of each cell c to its anterior sequences during the previous 8 iterations of the CA. We assessed whether this sequence of two states was repeated and with what periodicity $p = \min\{p \in [2,7]\}$, i.e. such that $(s(c,t),s(c,t-1)) = (s(c,t-p),s(c,t-p-1))$. We used a sequence of two states because if the genotype of a cell adopts a stable strategy, i.e. repeats a sequence of states, this sequence must contain a minimum of two states in order to ensure the survival of the genotype—the quiescent state and one of the living states. We chose to limit the comparison to

---

[1]It was shown by Medernach et al. (2015) that during the initial iterations of HetCA the MCGs were unlikely to exhibit any viable survival strategy.

(a)   (b)
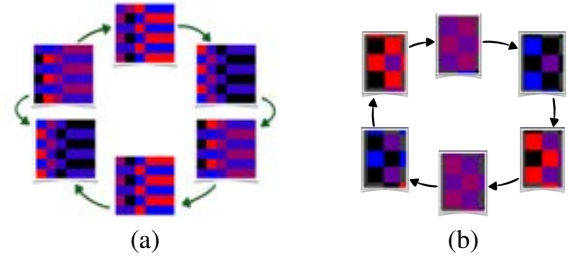
Figure 2: Examples of 6-step survival strategies: (a) Genotype extracted from a HetCA simulation in a stable environment (SE) with random homogeneous initialization. (b) Genotype produced by short-cycle fluctuations (ScF).

the 8 previous iterations in order to reduce the computational cost and because the limit of 7 consecutive live iterations before decay involves, for a successful regular phenotype, a maximum periodicity of 7 iterations for the quiescent state. We performed this measure only if there was at least one living state and no decay among the last two states. For each iteration t of the CA, we reported the phenotypic disturbance

$$P(t) = \sum_{p=1}^{7} \left| \frac{N(p,t)}{N(t)} - \frac{N(p,t-1)}{N(t-1)} \right|$$
(3)

where $N(p,t)$ is the total number of cell that had periodicity p at iteration t. This measure is rough but interesting because, unlike phenotypic differences, it is not directly based on states and therefore is less likely to be correlated to a state's probability of propagation.

Results

Genotype Size and Genotype Mutations

In the evolution of genotype size under the 4 types of environmental fluctuations (Fig. 3), we notice that the imposed size limit of 50 program statements tends to blur the differences between the different scenarios since most simulations converge to this limit. Yet, ScF clearly restricts the size of the genotypes more severely than SE, LF and SF, while these other conditions do not appear to influence genotype size. This size reduction by ScF could be a way to increase the impact of genotypic mutations on the phenotype. This is because, even though the number of statements potentially affected by mutations in LGP increases proportionally with genotype size, hence could have a larger effect on the phenotype, there can also be a "buffer effect" brought by information redundancy in longer genotypes, which would in fact stabilize the phenotype. Hence, mutations in more compact genomes might end up being more impactful[2].

Observing the amount of mutations separating the current MCG from individuals created during initialization (Fig. 4),

---

[2]This assessment was supported by an analysis of the effective length of the most common individuals (not reported here).
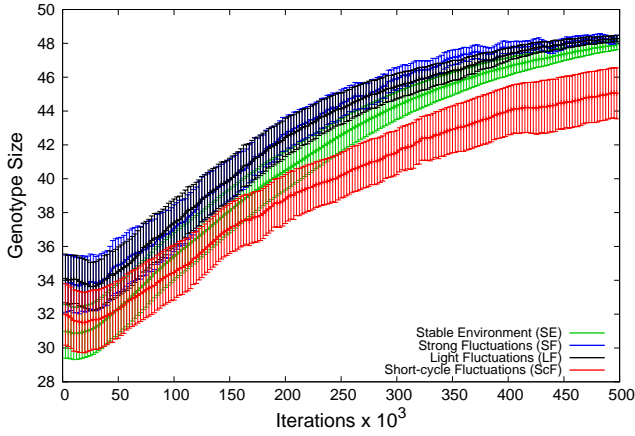
Figure 3: Evolution of genotype sizes (average $\pm$ SEM) in number of LGP statements.
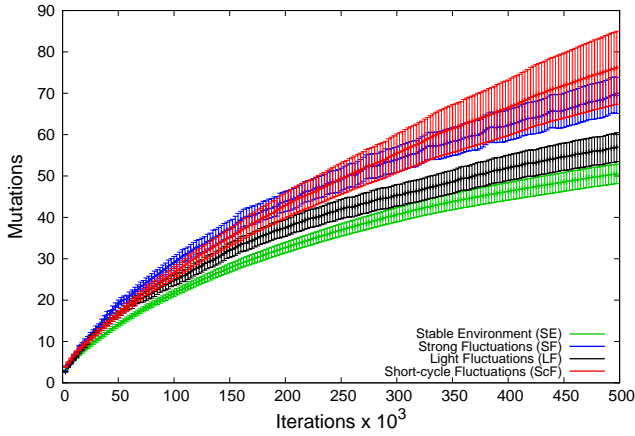
Figure 4: Evolution of number of ancestral mutations (average $\pm$ SEM) involved in producing the current MCG.

Figure 5: Evolution of phenotypic comparison function σ (average $\pm$ SEM) between dissimilar environments.
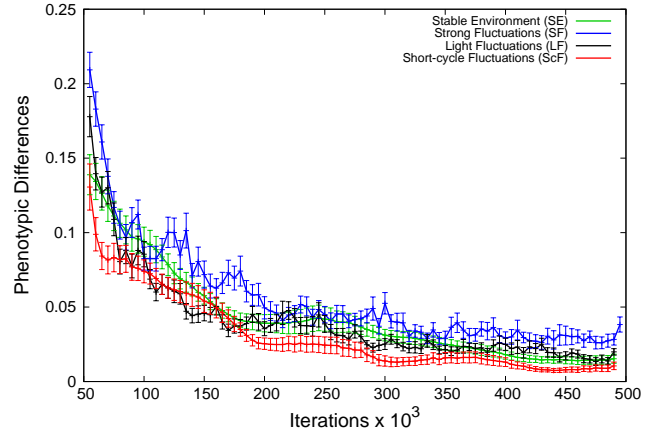
Figure 6: Evolution of phenotypic comparison function σ (average $\pm$ SEM) between similar environments.

it is not surprising to see that more mutations are selected for when environmental fluctuations are introduced. Their number also seem to depend more on the strength of these fluctuations (i.e. the contrast between two successive propagation probability patterns E) than on their periodicity. We also note that the proximity of SF and ScF indicates that their differences in size are not explained by differences in the number of selected mutations.

Phenotypic Comparison

In the phenotypic comparison $\sigma(t_1, t_2)$ between dissimilar environments, i.e. at t_1 and $t_2 = t_1 - \Delta t$ such that $E(t_1) \neq E(t_2)$ but $E(t_1) = E(t_2 + f)$ (Fig. 5), we observe that the impact of environmental fluctuations decreases quickly for ScF while it remains very high for other types of environmental fluctuations. The phenotypic difference of ScF also remains most of the time lower than the phenotypic differences of the SE. This suggests the selection of a single phenotype, robust in both environments. Looking at σ be-

tween similar environments, i.e. such that $E(t_1) = E(t_2)$ (Fig. 6), we note that phenotypic differences in LF and SF are much lower than in Fig. 5.

Phenotypic and Genotypic Diversity

Figures 7 and 8 depict the average phenotypic and genotypic diversities, ^2D_p and ^2D_g. The generally low phenotypic diversity of ScF suggests the existence in this configuration of a dominant phenotype, which remains rather stable over time, whereas the relatively high phenotypic diversity of LF and SF combined with their relatively low genotypic diversity might suggest the existence of strong phenotypic selection, hence some form of plasticity.

Success Rates of the Homogeneous Test

Success rates of genotypes in different homogeneous simulations are reported in Fig. 9 using a normal approximation with a 95% confidence interval. The fact that SE offers the lowest challenge is not surprising. Similarly, the fact that SF
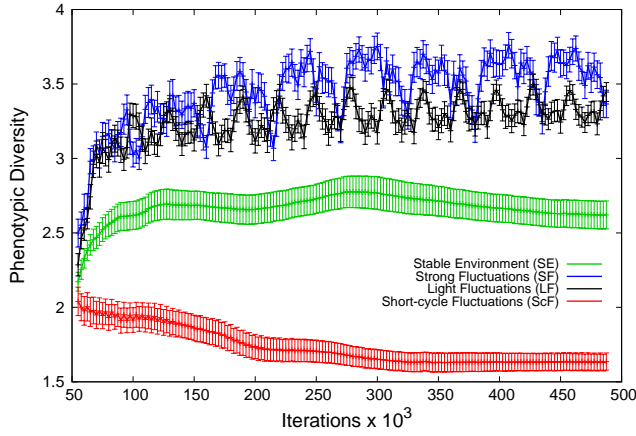
Figure 7: Evolution of phenotypic diversity (average $\pm$ SEM).

Figure 8: Evolution of genotypic diversity (average $\pm$ SEM).

is the least conducive to success is also expected. A comparison of the levels of difficulty between LF and ScF is less clear, however, since ScF performs significantly better in its own settings while on the contrary all other tested fluctuations are slightly more efficient in LF. It is also noteworthy that individuals from LF and SF seem relatively robust in various environmental configurations, while those from ScF seem fragile outside the environmental conditions in which they evolved. Moreover, among genotypes collected from iteration 102,500, these same individuals are the only ones that do not reach a 100% survival ratio in a stable environment.

Ending Iteration

Displaying the last iterations reached by living cells of homogeneous runs, with genotypes collected at iteration 100,000 and 500,000 (Fig. 10), we see that the ScF genotype failures are concentrated around iteration 15,000 in LF homogeneous tests and 25,000 in SF homogeneous tests. This corresponds for these two configurations to the first environment for which $K(s_3) = 0$, whereas $K(s_3) = 1$ for all distributions E in ScF. But we also see that some of the genotypes from ScF fail during the early iterations of the homogeneous test regardless of the environmental fluctuations, including SE and ScF. This might imply that the ecosystem resulting from the evolutionary history of the individuals plays a key role in their ability to survive in ScF.

Analysis

Environmental Transitions

Figure 11 visualizes the typical transitions between environments in different homogeneous runs. To compute these transitions we averaged the phenotypic disturbance P over iterations $[t-40, t+40]$ for all $t \geq 5,000$ and $t \in T_r$, where T_r is the sequence of iterations of run r at which a transition between environments effectively occurred, i.e. for which $E(t) \neq E(t+1)$. Figure 11d shows the average transitions
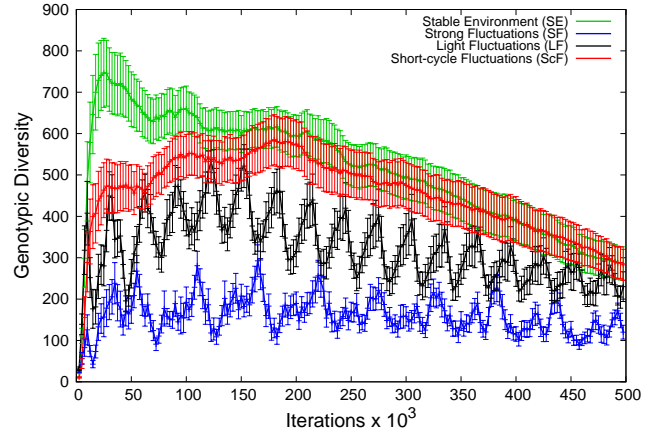
of ScF genotypes collected at the 6 aforementioned time steps {2,500, 102,500, ..., 500,000} and subjected to a homogeneous ScF test. It shows that phenotypes corresponding to genotypes collected later in the evolutionary process are less sensitive to environmental fluctuations. Conversely, as reported in Fig. 11b, the phenotypes of genotypes from SF keep the same high sensitivity regardless of the iteration at which they were collected. Finally, Fig. 11a and 11c compare the average transitions in homogeneous ScF and SF tests with genotypes collected at iteration 500,000 from the four different configurations. Again, it can be observed there that the phenotype of ScF is much more stable than the others in its original environment, but is at the same time very sensitive to transitions in SF.

Phenotypic Diversity

The phenotypic diversity measured in Fig. 7 can also be observed by visual inspection of the CA as displayed in the screenshots of Fig. 12. First, LF and SF are visibly different from ScF. These two groups diverge significantly in texture and also clearly differ from SE. Individuals from ScF seem to produce stable and robust phenotypes in any environment encountered within a ScF scheme. Their adaptations appear to be essentially created by genotypic mutations. They are also very dependent on their original ecosystem, sometimes distinctively so (Fig. 13), and as a consequence they are not robust in other types of fluctuations, where the effect of mutations is probably enhanced by the reduced size of the genotypes, as mentioned previously. By contrast, individuals evolved within LF or SF have high phenotypic diversity, and it seems likely that phenotypic selection occurs despite their lower genotypic diversity.

Conclusions

The phenotypic selection observed in this series of simulations evokes the multilevel selection model described by Jablonka et al. (2014). Among the three tested envi-
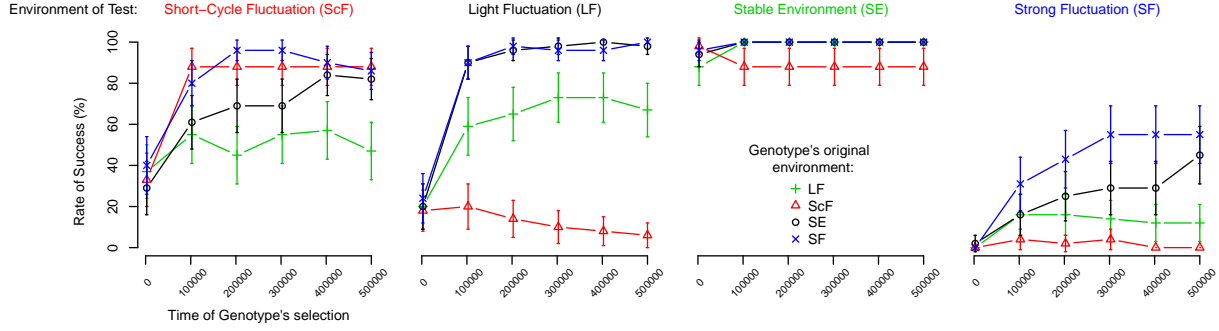
Figure 9: Evolution of success rate of genotypes (average ± SEM) in homogeneous tests.
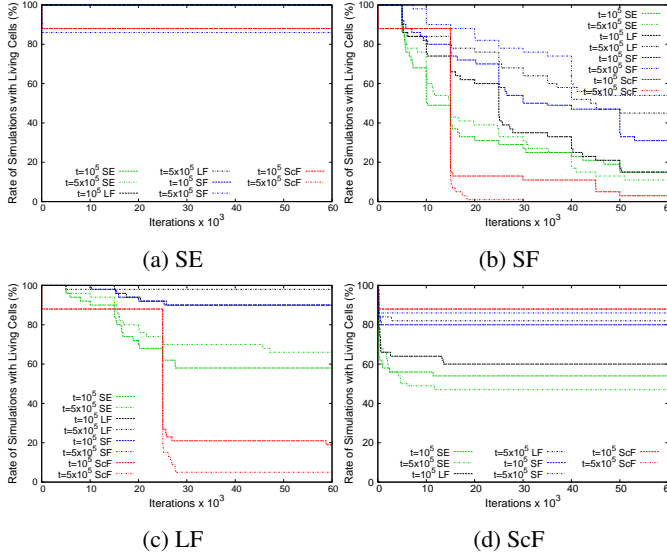
Figure 10: Last iterations reached by living cells of homogeneous runs of genotypes from iterations 100,000 and 500,000.

Figure 11: Phenotypic disturbance: Average transition between environments in different types of homogeneous tests.

ronmental fluctuations, the LF and SF simulations are displaying the greatest similarity with this model. In ScF, the best evolutionary strategy seems to involve small genotypes, which could be favored for their capacity to maximize the phenotypic impact of mutations. Furthermore, the inability of most ScF individuals to survive outside of the ecosystem resulting from their evolutionary history is reminiscent of the impossibility of saving species solely by preserving their DNA, as claimed in Jablonka et al. (2014): "You would have to reconstruct the community, and often these communities are very old, with historical memories that are stored in their epigenetic and behavioral systems. These are part of their 'identity,' part of their stability. You cannot freeze these memories: they have to be maintained and transmitted through use, so you cannot reconstruct the communities from their component parts." (*ibid.*, p. 363). However, these experiments alone could not determine whether the main distinguishing feature between LF and SF on the one hand

and ScF on the other hand was the duration of the environmental cycles or the number of environmental types. Further investigation is needed.

Acknowledgments

References

Bredeche, N. and Montanier, J.-M. (2012). Environment-driven open-ended evolution with a population of autonomous robots. In *Evolving Phys. Systems Workshop*.

Floreano, D. and Urzelai, J. (2000). Evolutionary robots with on-line self-organization and behavioral fitness. *Neural Networks*, 13(4):431–443.

SE   SF

LF   ScF
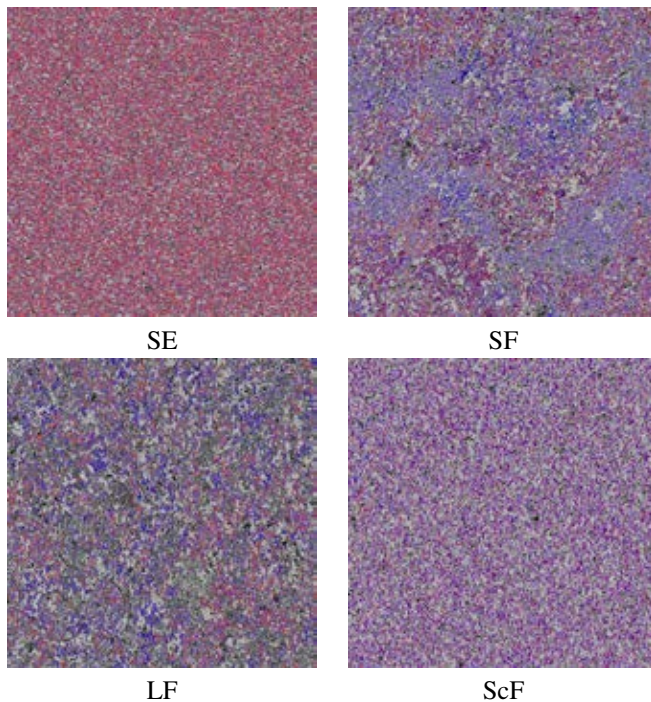
Figure 12: Screenshots of the CA. Grid state distribution (phenotype) at iteration 495,000 in the four different configurations. Each cell state is represented by a different color. Black and grey represent cells in the *decay* and *quiescent* states, respectively. Shades of blue, red and purple represent the living states.

$t = 100,000$ $t = 200,000$

$t = 400,000$ $t = 500,000$

Figure 13: Original ScF simulation: a distinctive "wavy" phenotype, very stable over time, produces genotypes that fail in the early iterations of the homogeneous test.

Heard, E. and Martienssen, R. A. (2014). Transgenerational epigenetic inheritance: myths and mechanisms. *Cell*, 157(1):95–109.

Jablonka, E., Lamb, M. J., and Zeligowski, A. (2014). *Evolution in Four Dimensions, revised edition: Genetic, Epigenetic, Behavioral, and Symbolic Variation in the History of Life*. MIT press.

Jost, L. (2006). Entropy and diversity. *Oikos*, 113(2):363–375.

Lachmann, M. and Jablonka, E. (1996). The inheritance of phenotypes: an adaptation to fluctuating environments. *Journal of theoretical biology*, 181(1):1–9.

Laland, K., Matthews, B., and Feldman, M. W. (2016). An introduction to niche construction theory. *Evolutionary Ecology*, 30(2):191–202.

Levins, R. (1968). *Evolution in changing environments: some theoretical explorations*. Princeton Univ Press.

Lipson, H., Pollack, J. B., and Suh, N. P. (2002). On the origin of modular variation. *Evolution*, 56(8):1549–1556.

Lloyd, E. (2012). Units and levels of selection. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Stanford University, spring 2012 edition.

Medernach, D., Fitzgerald, J., Carrignon, S., and Rya, C. (2015). Evolutionary progress in heterogenous cellular automata (hetca). In *Proceedings of the European Conference on Artificial Life 2015 (ECAL 2015)*, volume 13, pages 512–519.

Medernach, D., Kowaliw, T., Ryan, C., and Doursat, R. (2013). Long-term evolutionary dynamics in heterogeneous cellular automata. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 231–238. ACM.

Müller, G. B. (2007). Evo–devo: extending the evolutionary synthesis. *Nature Reviews Genetics*, 8(12):943–949.

Okasha, S. (2006). *Evolution and the levels of selection*, volume 16. Clarendon Press Oxford.

Shanahan, T. (2012). Evolutionary progress: conceptual issues. *eLS*.

West-Eberhard, M. J. (2005). Developmental plasticity and the origin of species differences. *PNAS*, 102(suppl 1):6543–6549.

Yu, T. (2007). Program evolvability under environmental variations and neutrality. In *Proceedings of the 9th annual conference companion on Genetic and evolutionary computation*, pages 2973–2978. ACM.

Morphology

On the Difficulty of Co-Optimizing Morphology and Control in Evolved Virtual Creatures

Nick Cheney[1], Josh Bongard[2], Vytas SunSpiral[3] and Hod Lipson[4]

[1]Department of Biological Statistics and Computational Biology, Cornell University

[2]Department of Computer Science, University of Vermont

[3]Intelligent Systems Division, NASA Ames, SGT Inc.

[4]Department of Mechanical Engineering, Columbia University

nac93@cornell.edu

Abstract

The field of evolved virtual creatures has been suspiciously stagnant in terms of complexification of evolved agents since its inception over two decades ago. Many researchers have proposed algorithmic improvements, but none have taken hold and greatly propelled the scalability of early works. This paper suggests a more fundamental problem with co-evolving both the morphology and control of virtual creatures simultaneously – one cemented in the theory of embodied cognition. We reproduce and explore in greater detail a previous finding in the literature: premature convergence of the morphology (compared to the convergence point of optimizing controllers), and discuss how this finding fits as a symptom of the proposed problem. We hope that this improved understanding of the fundamental problem domain will open the door for further scalability of evolved agents, and note that early findings from our future work point in that direction.

Introduction

In 1994, Karl Sims' seminal work on "Evolving Virtual Creatures" (Sims, 1994b) created a field of study by that name. This work featured simulated creatures that were able to optimize both their physical layout and their behavioral control strategies for such tasks as terrestrial locomotion, swimming, phototaxis, and competition (Sims, 1994a).

The potential applications of virtual creatures extends beyond their initial contribution to computer graphics and animation, serving as a testbed for the co-optimization of brain-body systems in robotics. With the challenges of continually modifying the morphology of physical robots during the optimization process, the field of Evolutionary Robotics often turns to virtual creatures to optimize morphologies (and their associated controllers) before physical robots are manufactured from the optimized designs (Lund et al., 1997; Funes and Pollack, 1998; Lipson and Pollack, 2000; Nolfi and Floreano, 2002; Doursat et al., 2012; Bongard, 2013).

However, in the two decade lifetime of this field, there have been notable struggles in optimizing creatures, with a very limited ability to extend beyond Sims' initial works (Geijtenbeek and Pronost, 2012) – despite significant increases in computing power. Many researchers have suggested hypotheses for the cause of this standstill, such as de-

ficiencies in the search algorithms (Hornby, 2006; Lehman and Stanley, 2011; Mouret and Clune, 2015) or genetic encodings (Hornby et al., 2001; Bongard and Pfeifer, 2003). It has also been suggested that the environments/tasks chosen are not complex (or morphologically dependent) enough to necessitate optimization of both the morphology and controller (Auerbach and Bongard, 2014; Cheney et al., 2015). But since we have yet to clearly surpass Sims' work, each of these hypotheses must be approached with some skepticism.

This work takes note of the particular difficulty in optimizing morphology (Joachimczak et al., 2016) and sets out with the intent of proposing a new hypothesis for the field's current roadblock. Our hypothesis, unlike many before it, does not rely on more powerful or astute search algorithms to laboriously make our way through the rugged and harsh search space which make optimization of virtual creatures so difficult. Rather, we intend to use our understanding of the behavior of virtual creatures, specifically the theory of embodied cognition, to suggest a fundamental issue in the way that we frame the problem of optimization of virtual creatures – which in turn causes the search landscapes to present such an unpleasant terrain.

The theory of embodied cognition suggests that a fundamental part of the cognitive control process of an individual is being situated (Wilson, 2002). It suggests that the dynamic interactions between a reactive agent and the environment, through sensory-motor feedback loops, are an important driver of behavior (Brooks, 1991), as opposed to cognitivism – the hypothesis that the central functions of mind can be accounted for in terms of the manipulation of symbols according to explicit rules (Anderson, 2003).

This line of reasoning puts an extra emphasis on the morphology of an individual, as it acts as the lens and modulator for all physical communication between that individual's internal controller and the outside environment (Pfeifer and Bongard, 2006). This work outlines the specific hypothesis that the body's importance, afforded to it by its role as the connection between internal desires for action and the external consequences of them (as well as external events and the internal sensory observations of them), is understood. With-

out a well established and properly functioning communication channel, the sensory information and motor commands of an individual are ineffective.

From this supposition, we can create a testable hypothesis about the value of the established morphological communication channel. Specifically, control optimization on an existing morphology can be more effective than morphological optimization on a fixed controller – as the latter does not maintain an established communication framework from the controller to the environment (through the morphology). This results in a system which effectively causes large, unintended variations in the behavior of the controller, as its physical interface is constantly being scrambled while optimization seeks to improve the physical shape of the body.

In comparing each of these hypothetical situations to the current state of evolved virtual creatures, we will conclude by discussing a possible connection between this theory of embodied cognition and the lack of effective optimization. Our hope is that such evidence will shed additional light on (at least one of) the problem(s) facing our field, and arm us with the information to help tackle it in future works.

Background

The literature on failed attempts to co-optimize the morphology and control of virtual creatures is sparse. This may be due in part to the bias against publishing negative results (both in submission and acceptance of such findings) (Fanelli, 2011). However, informal conversation with members of the field acknowledge the lack of progress. We note the difficulty of optimizing morphologies in our own virtual creatures (Lipson and Pollack, 2000; Bongard and Pfeifer, 2003; Cheney et al., 2014, 2015) (and unpublished works), but find ourselves grasping for an understanding of why this may be the case.

One clear and concise description of this very problem is expressed in Joachimczak et al. (2016), where they note:

It can also be observed how during the first 100 generations of the evolutionary run, morphological changes occurred very frequently. At generation 125, the overall morphology of the best individual already resembles the best final individual found in the generation 1386 (although its fitness is only 5.07, compared to the 11.15 of the latter). The following generations bring multiple small changes to the morphology of adult form and almost no changes to the larval form. Both stages, however, undergo continuous modifications of their controllers, and it is these alterations that contribute the most to the improvements in fitness. This pattern was also observed in other evolutionary runs: **the final morphology would emerge in the first few hundreds of generations and the remainder of the run would be spent on small tweaks to the bodies and optimization of controllers**. *(emphasis added)*

This notion of premature convergence of morphology is not a stand alone case. At times this premature convergence can be incorrectly interpreted as a positive trait, noted as diversity of results (despite the lack of explicit diversity maintenance), as in Cheney et al. (2013).

In the remainder of this work we set out to reproduce the symptoms described in Joachimczak et al. (2016), where morphology converges prior to control. We seek to further examine and characterize this phenomenon, and describe a theoretical framework which may help to explain its cause.

Methods

Similarly to Joachimczak et al. (2016), we employ soft robots as our instantiation of evolved virtual creatures. We use 3D voxel-based soft robots, following from Cheney et al. (2013), but replace their discrete muscle types and synchronized contractions with voxels which allow individualized phase offsets, consistent with the controllers used in Joachimczak et al. (2016). This allows for behaviors such as propagating waves, which were not possible in Cheney et al. (2013) (but were achieved in Joachimczak et al. (2016) and Cheney et al. (2014)). A global frequency of oscillations is also optimized.

Dual-Network CPPN

We genetically encode the soft robot phenotypes as a network, inspired by the CPPN-NEAT (Stanley, 2007), the algorithm employed by both Cheney et al. (2013) and Joachimczak et al. (2016) (though the later cleverly employs the CPPN alongside development, rather than as an alternative to it). However, this work differs from those two by optimizing two separate networks, one containing only the outputs associated with the physical structure and material placement ("morphology") of the creatures, while the second network produces only the outputs used to determine the actuation of the muscle voxels ("control"). This allows us to very clearly make variations to either the morphology or the controller, without affecting the genotype of the other[1].

To translate the CPPN genotype to a soft robot phenotype, for each individual voxel in our $7 \times 7 \times 7$ discretized design space, the "presence" output of the morphology network is queried. If the output value (which all span the range $[-1, 1]$) is positive, a voxel is placed there and the "material type" output is queried. If the "material type" output is positive as well, then a the voxel is an active "muscle" cell, otherwise, that voxel is a passive "tissue" cell.

For each active muscle cell, the control network is queried, and the floating point value of the "phaseOffset" output (again from $[-1, 1]$) is assigned as the relative phase offset of that muscle cell (where 0 is exactly in phase with a global clock, -1 and 1 are synchronized a full phase ahead or behind it, and -0.5 and 0.5 are perfectly out of sync with

---

[1]both source code and resulting data are available upon request

it). Finally, the frequency of this global "clock" oscillator is set using the mean value of the "frequency" output across all voxels (including those not currently expressed in the phenotype). In order to easily allow the full range of possible frequencies to be expressed after averaging, a mean value of -0.5 or lower corresponds to the minimal frequency of 5Hz, while a mean value of 0.5 or higher corresponds to the maximal frequency of 10Hz (with linear scaling between them), despite the continued $[-1, 1]$ range of each individual "frequency" output node. The optimization of the global oscillation speed is intended to allow the muscle actuations to resonate with the natural frequency of a given morphology.

We should note that this encoding does allow for morphological changes to affect the expressed control (as the addition or removal of muscle cells will allow more or less of the underlying phase offset pattern to be expressed in the phenotype). Due to the integrated and embodied nature of control, we believe that such an effect would happen with various definitions of "morphology" and "control" – such as a robot with 6 legs expressing a different number of joint control outputs than a 4 legged robot in the rigid body paradigm. This concept of morphology determining the expression of control may be less about this specific implementation and instead a more general consequence of embodied cognition in a situated creature (Pfeifer and Bongard, 2006).

Physics Simulation in VoxCad

Consistent with Cheney et al. (2013), we employ the open-source soft-body simulator VoxCad (Hiller and Lipson, 2014) as the physics engine which determines the fitness of each creature's phenotype. In order to normalize the number of actuations per muscle cell across creatures with different actuation frequencies, each individual is evaluated for exactly 20 actuation cycles (following a passive initialization period in which it is allowed to settle on the ground in a relaxed pose – intended to discourage passive falling strategies rather than active locomotion behaviors). This means that two creatures with different actuation frequencies will be simulated for different lengths of time. Following the termination of the simulation, the displacement of the creature's center-of-mass along the positive x axis is returned to the evolutionary algorithm. All other parameters regarding VoxCad simulation are taken from Cheney et al. (2013).

Evolutionary Algorithm

The optimization of these soft robots takes the form of an evolutionary algorithm. The genotype is a directed acyclic graph, represented in memory as a tree to allow an implementation similar to that of genetic programming. Following from CPPNs (Stanley, 2007), each node in the graph sums its weighted inputs and feeds them through a series of nodes with geometric activation functions (here: sigmoid, sine, absolute value, negative absolute value, square, square root, or negative square root) to arrive at each of its output

value(s). The inputs to this network are Cartesian (x, y, z) and polar (r) coordinates of the voxel in question, along with a bias node. The outputs are interpreted as described above.

Variation and selection follow a $(\mu/\rho + \lambda)$ scheme of $(50/25 + 25)$. Variations may be: the addition/removal of a node to a network, addition/removal of an edge between existing nodes, mutation of the weight associated with an edge, or mutation of a node's activation function. Each of these variations occurred with equal probability, and each variation occurs to only one network of the phenotype, each with equal probability. Crossover was not considered in this work. Variations to the genotype were only considered valid if they resulted in a phenotypic change in the resulting soft robot. Variations were also disallowed if they resulted in creatures who occupied less than 10% of the available voxels, or employed less than 5% of the available voxels as actuated muscle cells. Selection was rank-based with elitism.

Statistical Reporting

All experimental data below represent the mean values of 30 independent runs lasting for 5000 generations each. P-values are calculated using a Mann-Whitney rank-sum test, as we cannot assume normality of fitness values. Confidence intervals were plotted using bootstrapping of 10,000 samples at the 95% confidence level. Significance values are marked with the following convention: *ns* for $p > 0.05$, * for $p \leq 0.05$, ** for $p \leq 0.01$, and *** for $p \leq 0.001$.

Results

First and foremost, we set out to replicate and examine the results founds in Joachimczak et al. (2016), where "the final morphology would emerge in the first few hundreds of generations and the remainder of the run would be spent on small tweaks to the bodies and optimization of controllers."

By visually inspecting the resulting creatures we find that this implementation appears able to reproduce the phenomenon. Fig. 1 shows the optimization over time of the 10 best performing trials. Notice how conserved the morphologies appear to be over time, with the gross morphology generally emerging at or before the 100 generation mark (middle column). While only the top 10 trials are shown for sake of space, this theme applies generally to all the runs.

It is also interesting to note that the top two final-fitness-achieving runs were the only two to undergo a morphological change between generations 1000 and 5000 (the last two columns). This suggests that creatures to which search immediately converges upon are not optimal, and that better performing solutions may not be that far away in phenotypic space (inferred from the similarities between the top two rows at generations 1000 and 5000), yet such creatures appear to be difficult for this search process to find (inferred by the lack of occurrence before generation 1000 in the top two runs, and at all in the next 8 runs). The idea that each run converges to a local, rather than global, optimum is also
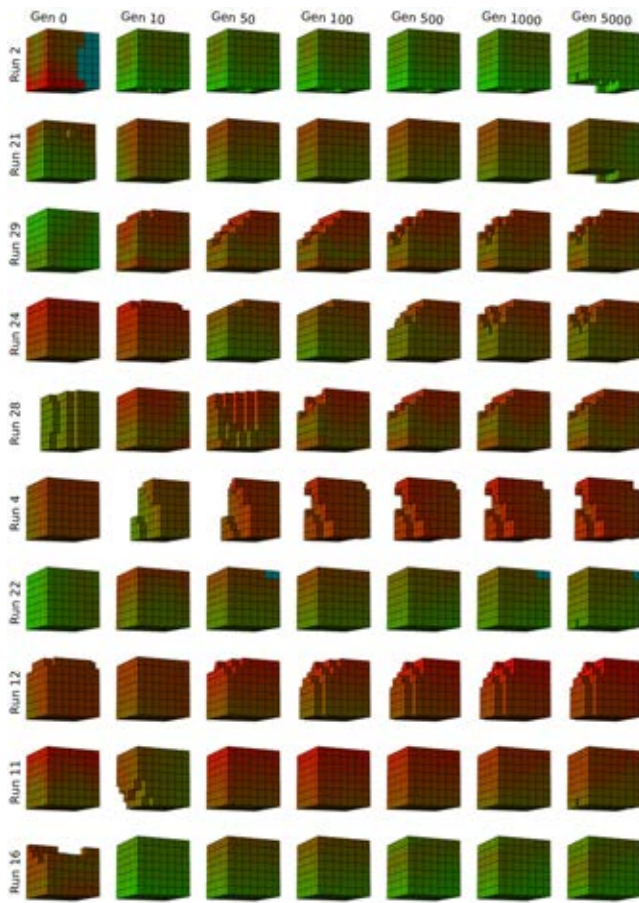
Figure 1: Evolved morphologies at various stages in optimization (voxel color from red to green indicates phase offset of controllers). Each row represents one of the top 10 run (out of 30, order by final fitness). Each column represents a point in time during optimization. Note that morphologies generally lock in before gen 100, often on simple forms.

evident by the fact that the set of final creatures differ from one another, rather than converging to the same form.

This visualization serves as an initial indication that the effect of early convergence is apparent in our setup, as it was in Joachimczak et al. (2016). However, it does not demonstrate that the effect of stagnation is more prominently featured by morphology than controllers, or characterize just how detrimental such an effect may be. These two questions are both approached quantitatively in Figs. 2 and 3.

To quantify how early the morphology converges and how detrimental this may be towards the optimization of virtual creatures, we artificially freeze the morphology after a given amount of time, and only allow control variations to occur after this point. If the resulting fitness value does not show a significant change following a morphology freeze at a given time (compared to optimizing both the morphology and control for the entire optimization process), we can be confident
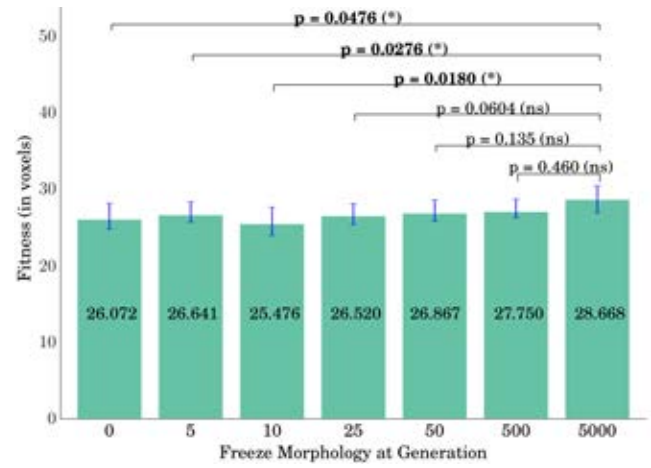
Figure 2: Fitness impact of freezing morphology at various points in optimization. Both morphology and control are optimized up to the freezing point. After it, only control variations are considered for the reminder of the trial. The p-values (and significance markers) reported compare the resulting fitness to that achieved with co-optimization of both morphology and control for the full 5000 generations. Note that morphologies optimized for 25 or more generations show no significant fitness difference, compared to those optimized for all 5000 (noted above in bold).
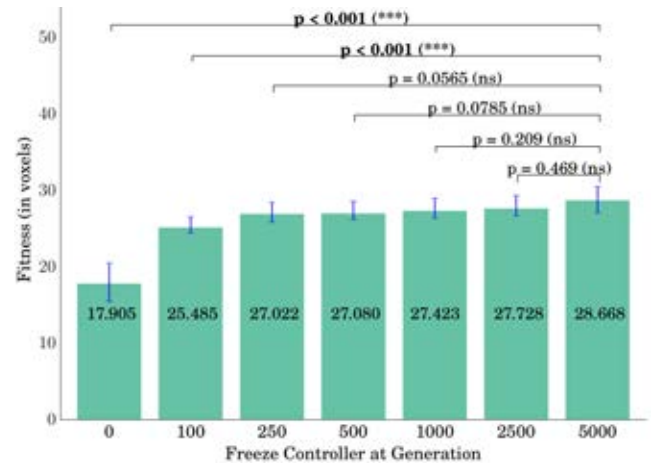
Figure 3: Fitness impact of freezing control at various points in optimization. Note that controllers with less than 250 generations of optimization (but full morphological optimization) show no significant difference with those optimized for all 5000 gens, suggesting that control mutations continue to provide fitness benefits further into optimization than the morphology variations, which cease to be beneficial to final fitness values much earlier (cf. figure 2, generation 25 – please note the different x-axis compared to that Figure).

that the morphology did not significantly contribute to fitness improvements after that point in optimization time.

Fig. 2 shows the fitness impact of morphology freezes at various times during optimization compared to co-optimization of morphology and control for the entire 5000 generations. We see that full optimization does not show a significant fitness improvement compared to morphologies optimized for 25 generations or more (at the 95% confidence level, as $p \geq 0.0604$ for all freezing points ≥ 25 gens). This means that the morphological variations after generation 25 do not significantly contribute to the fitness of the resulting creatures, suggesting that morphology converges to (near) final forms by generation 25. The visual inspection of these creatures in Fig. 1 does not contradict such a suggestion.

This does not mean that optimization as a whole is converged at this point. Improvements from control optimization occurring after the final gross morphology is fixed are noted in Joachimczak et al. (2016). We also see this effect here, with the fitness resulting from control optimization after morphology freezing (26.520) significantly outperforming ($p < 0.001$) the fitness at the time of freezing (21.157).

Fig. 3 shows the impact of the converse treatment, in which the creature's controller is frozen at a given point in time and only morphological variations are allowed thereafter. This treatment shows that significant differences in resulting fitness values occur for at least 100 generations (at the 95% confidence level, as $p < 0.001$ for freezing points ≤ 100), but not more than 250 generations ($p \geq 0.0565$ for freezing points ≥ 250). The lack of significant difference past 250 generations also points to early convergence of controllers to (near) final levels early in optimization.

However, the significant drop in fitness from control freezing (at times past those when morphological change stops contributing to final fitness values) suggests that this example of virtual creature evolution creates earlier convergence for morphologies than it does for controllers.

This picture is further reinforced when we examine the time of convergence to a final morphology and controller in each run. On average, the convergence to the final (best of run) morphology occurs at generation 558. In comparison the mutation which leads to the best-ever controller occurs significantly later ($p < 0.001$) at generation 2926. Widening our view from only the final successful variations, and considering all individuals who were the top fitness performers at some point during optimization, we see the same story, with controller mutations leading to top performers continuing significantly later than those created by morphological mutations (mean of gen 750 vs. gen 158, $p < 0.001$). The next section will discuss a potential cause for such an effect.

Discussion

The above results suggest that, in this instance of virtual creatures co-evolving morphology and control, we run into a problem of premature convergence, which is especially pro-

nounced with regard to the morphology of the creature. Premature convergence alone could point to issues in any number of aspects of optimization (diversity maintenance, genetic encoding, etc.). However the difference between optimization effectiveness of morphology and control draws our attention towards the theory of embodied cognition.

Let's revisit the concept of the morphology as the interface between the control architecture and its effect on the environment. This suggests that modifications to an agent's morphology will not only change the shape of its body, but also change the way in which its control architecture affects the environment, since the commands sent by that controller will now be interpreted differently – as it affects the actuators of a different body layout. Thus mutations to the morphology of a creature will have the effect of also "scrambling" its controller (causing variation in it) as well.

Contrary to the chain reaction effect of morphological mutations, variations which occur to the controller do not affect any part of the morphology's relationship with the outside environment. While the control signals which the body is receiving may change, these new commands are still executed in the same framework and "language" as previous commands were. The organization and path of information from controller through morphology to environment causes variations in the morphology to propagate upstream (i.e. affecting the controller/morphology interface in addition to the morphology/environment interface), while variations to the controller do not propagate downstream (affecting the control/morphology interface, but not the morphology/environment interactions).

This feature of embodied cognition has the effect of creating larger (and arguably more unpredictable) behavioral changes to similar sized variations to the "morphology" genome than the "controller" genome. This effect would lead to a more rugged fitness landscape in the space of morphologies (for a given controller) than exists in the fitness landscape of controllers (for a given morphology). We would then predict that a more rugged landscape would lead to more local optima and less efficient optimization with quicker convergence to sub-optimal solutions than in less rugged landscape (Kauffman, 1993). This is consistent with what we have experienced thus far with the optimization of morphology converging prior to control.

Potential Causes and Limitations

There are undoubtedly features of this experimental setup which may cause us to overstate (or understate) the importance of embodied cognition compared to other instances. Firstly, this setup employs soft robots, which are notoriously compliant and adaptive to a wider variety of environmental conditions than their rigid body counterparts (Trivedi et al., 2008). Given that adaptability of this robot-environment interface (in our case to unexpected perturbations in control signals), it's possible that soft robots dampen this effect. In

the extreme, one may conjecture that the soft robot paradigm is so compliant that almost any morphology can adequately move along flat ground. If this is the case, then it would not be surprising that freezing the morphology on an arbitrary shape has little effect on the resulting fitness value. As soft robots are relatively new to the literature, this may explain why this effect has been unnoticed previously.

In order to further explore this facet, we produced an alternative fitness function which explicitly selects for shape (adding a term to minimize the number of actuated voxels or "energy"). In the extreme this would produce creatures with minimal muscle cells, though since actuated cells directly contribute to locomotion ability, a complex trade-off creates an incentive for specialized energy-efficient morphologies. Another way incentivize to specialized morphologies would be to evaluate the robot in a more complex (and morphologically dependent) task environment than flat ground.

Performing the same "freezing" tests on creatures evolved under the alternative fitness criteria, we see that that freezing morphology continues to show a non-significant effect on fitness at times when control freezing produces a significant fitness drop (e.g. gen 50). Fig 4 visually shows the continued convergence to final gross morphologies (with morphologies at gen 50 generally mirroring those found at gen 5000), as well as the added morphological dependence of the task – as the morphologies demonstrated here visually appear more complex than the more fully occupied shapes in Fig. 1.

In this treatment, we also see the final controllers appearing significantly later (gen 2968) than the final moprhologies (gen 419, $p < 0.001$). This is also seen in the average best-so-far individuals, with those produced by control mutations continuing to appear significantly later on average (gen 709) than those produced by morphological variations (gen 119, $p < 0.001$). This data suggests that while the original task was not as "morphologically dependent" as others, the findings still hold in a scenario which puts more of an emphasis on morphological optimization.

A second aspect which may contribute to this effect is the size of the search space. These runs use robots of size $7 \times 7 \times 7$. As each of these voxels can have one of three states (empty, actuated, or passive) which results in $3^{343} = 4.5 * 10^{163}$ distinct morphological phenotypes. It's possible that the difficulty in searching the morphology space is due in part to its size. This could explain why this effect was not seen sooner (as previous work in evolutionary robots heavily favors legged morphologies with low degrees of freedom). This phenotype is indirectly encoded, but generative in different ways than previous work evolving morphology (Sims, 1994b; Lehman and Stanley, 2011).

In attempting to reproduce the work from Joachimczak et al. (2016), we optimize phase offset and frequency for an oscillating actuation as the control parameters. These values are encoded by floating point numbers, and thereby create a continuous (theoretically infinite) search space for control.

(a) Generation 50:

(b) Generation 5000:
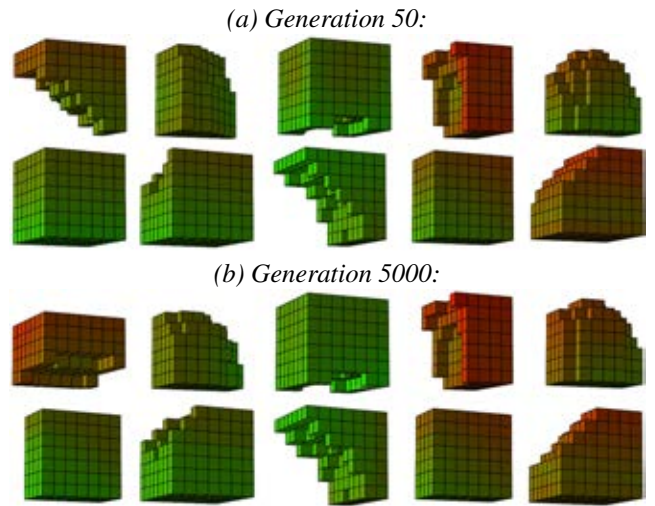
Figure 4: Stagnation shown in the top 10 morphologies under the distance/energy fitness treatment. Note the similarity in gross morphologies from gen 50 (top) to gen 5000 (bottom). The top performing creature shows the largest change between these points, with the new morphology arriving from a mutation at gen 53. Also note the variance and complexity in forms, compared to Fig. 1, suggesting the added morphological dependence of this fitness function.

The concept of discrete physical cells creating a morphology and real valued control parameters (such as neuronal synapse weights) fits biologically – but the differing search spaces give us pause from an optimization perspective.

To create a similar scenario where the size of the controller search space was smaller than that of the morphology, we borrow the two distinct "muscle type" system from Cheney et al. (2013). This allows just two offset control states (implemented by rounding the continuous phase offset values) to create a search space of size $2^{343} = 1.8 * 10^{103}$ (smaller than the morphology space). In this set of trials, we see the above effect disappear, and morphology no longer appears to be more difficult to optimize than "control". Here, the final morphological innovation of each run, on average, occurs at generation 665, while control innovation continues only to gen 795 – an insignificant difference ($p = 0.149$). Similarly, the point at which freezing morphology causes a non-significant difference in resulting fitness values no longer occurs before that of controller freezing.

However in this scenario, the line between "morphology" and "control" becomes very blurry. In practice, a two-oscillator-actuation system can be viewed as the placement of cells of these two types (a "morphological" concept) more so than the fitting of phase offset parameters to a predefined placement of muscles (a "control" concept). Thus one could easily argue that the two discrete-phase-offset system from Cheney et al. (2013) should be considered to be en-

tirely morphological optimization, with little to no control to be optimized (as is argued in that paper), and thus immune from our embodied cognition argument.

This is representative of a larger "problem" of this CPPN oscillating actuation setup: that there may not be a clean distinction between "morphology" and "control" to be made, and such divides may be arbitrary labeling. In our example, one could argue that the output node denoting if a cell is actuated or passive should belong in the "morphology" CPPN, as it denoted the placement of different types of cells ("muscles" or "tissues"). But another person could argue equally well that this output belongs in the "control" CPPN, since it does not change the actual shape or stiffness of the creature, and only informs where actuations do or do not occur.

The point here is that virtual creatures are situated and embodied, and thus ideas like embodied cognition or morphological computation (Pfeifer and Gómez, 2009) suggest that there isn't a clear cut distinction or dualism between two separate pieces (the body and the brain), but rather a single integrated and embodied agent. Therefore we need to consider the tight coupling and interdependencies of the "morphology" and "control" and consider holistic effects whenever we attempt to modify a single part of the system.

Future Work

The results shown in this work are specific only to this instance and experimental setup. Thus, many more instances of this approach (artificially separating morphology from control and freezing each to measure their independent impact on fitness) would need to be attempted on different experimental setups to extrapolate from this single instance. This should ideally include different: morphological encodings (such as the generative block encodings used by Sims (1994b); control architectures (perhaps complexifying to neural nets rather than simplifying to discrete oscillations as we did in our follow up tests – or employing closed-loop control, which may help controllers to adapt to new morphologies); evolutionary algorithms (especially those with a strong emphasis on diversity); tasks (increasing environmental complexity); and/or scales (as increased scales of a cellular creature closer approximate a "continuous" morphology – which comes with various benefits and costs).

Regarding the distinction between "morphology" and "control", this work necessarily chooses a logical splitting point between the two: representing CPPN outputs that dictate placement of voxels as "morphology" and outputs that dictate voxel size changes as "control". But this distinction is far from black and white. Future work should explore various groupings of outputs into the categories of "morphology" and "control" (or any grouping names), and examine the effect that such distinctions produce on these results.

The central issue to this paper can be viewed as a problem stemming from the dynamic coupling of control on morphology, with different morphologies creating hills and valleys in the fitness landscape of controllers. As in any multi-modal landscape, diversity maintenance during search is crucial. This includes diversity coming from crossover (omitted here), or from any existing diversity maintenance method. However, informed by this paper, we would be wise to notice that since hills and valleys of this landscape may be caused *by* the morphology and *onto* the controller, diversity maintenance would do best to focus on protection of diversity within morphologies if it were to encourage the morphological variations (despite their adverse effects on control).

The most important future work would involve potential solutions to this problem. Initial results regarding future work already suggest that our understanding of embodied cognition, and the finding of especially poor mutation success for morphological variations, can inform improved search methods. Specifically, results employing a multi-timescale model, in which morphological mutations are given time to re-adapt their controllers to their new situated forms (and thus conform themselves to their new morphological "communication channels", thereby "unscrambling" the detrimental effects of the morphological mutation) before the value of these morphological variations are evaluated, shows an improved ability for optimization of virtual creatures compared to traditional methods. This is exactly the type of diversity maintenance that focuses on protecting innovations to the morphology specifically.

Ideally, further algorithmic improvements will occur from embracing the fundamental theory of embodied cognition, but the positive initial results noted here provide conformation that it's possible and that the understanding gained from this current work may contribute to future improvements.

Conclusion

We have examined a specific example of co-evolving morphology and control in virtual creatures. In this example, morphology prematurely converges: converging quicker than control, showing a lack of fitness benefits after as little as 25 of the 5000 generations, and with "optimal" final morphologies emerging significantly sooner than final controllers. We have suggested a theoretical basis, founded in the concept of embodied cognition, that could explain such a obstacle and is consistent with the results we present. While there is plenty of work still to be done to solidify this theory, we conclude by suggesting future work based from our newly proposed understanding, and note its striking potential in early initial results. We hope this work will help to explain the difficulty we face in scaling the complexity of evolved virtual creatures, and will help inspire (combined with other efforts) a solution to our current stagnation.

Acknowledgments

References

Anderson, M. L. (2003). Embodied cognition: A field guide. *Artificial intelligence*, 149(1):91–130.

Auerbach, J. E. and Bongard, J. C. (2014). Environmental influence on the evolution of morphological complexity in machines. *PLoS Comput Biol*, 10(1):e1003399.

Bongard, J. C. (2013). Evolutionary robotics. *Communications of the ACM*, 56(8):74–83.

Bongard, J. C. and Pfeifer, R. (2003). Evolving complete agents using artificial ontogeny. In *Morpho-functional Machines: The New Species*, pages 237–258. Springer.

Brooks, R. A. (1991). Intelligence without representation. *Artificial intelligence*, 47(1):139–159.

Cheney, N., Bongard, J., and Lipson, H. (2015). Evolving soft robots in tight spaces. In *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*, pages 935–942. ACM.

Cheney, N., Clune, J., and Lipson, H. (2014). Evolved electrophysiological soft robots. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 222–229.

Cheney, N., MacCurdy, R., Clune, J., and Lipson, H. (2013). Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 167–174. ACM.

Doursat, R., Sayama, H., and Michel, O. (2012). *Morphogenetic engineering: toward programmable complex systems*. Springer.

Fanelli, D. (2011). Negative results are disappearing from most disciplines and countries. *Scientometrics*, 90(3):891–904.

Funes, P. and Pollack, J. (1998). Evolutionary body building: Adaptive physical designs for robots. *Artificial Life*, 4(4):337–357.

Geijtenbeek, T. and Pronost, N. (2012). Interactive character animation using simulated physics: A state-of-the-art review. In *Computer Graphics Forum*, volume 31, pages 2492–2515. Wiley Online Library.

Hiller, J. and Lipson, H. (2014). Dynamic simulation of soft multimaterial 3d-printed objects. *Soft Robotics*, 1(1):88–101.

Hornby, G. S. (2006). Alps: the age-layered population structure for reducing the problem of premature convergence. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 815–822. ACM.

Hornby, G. S., Lipson, H., and Pollack, J. B. (2001). Evolution of generative design systems for modular physical robots. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 4146–4151. IEEE.

Joachimczak, M., Suzuki, R., and Arita, T. (in press: 2016). Artificial metamorphosis: Evolutionary design of transforming, soft-bodied robots. *Artifical Life*.

Kauffman, S. A. (1993). *The origins of order:Self organization and selection in evolution*. Oxford Universty Press.

Lehman, J. and Stanley, K. O. (2011). Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218. ACM.

Lipson, H. and Pollack, J. B. (2000). Automatic design and manufacture of robotic lifeforms. *Nature*, 406(6799):974–978.

Lund, H. H., Hallam, J., and Lee, W.-P. (1997). Evolving robot morphology. In *Evolutionary Computation, 1997., IEEE International Conference on*, pages 197–202. IEEE.

Mouret, J.-B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*.

Nolfi, S. and Floreano, D. (2002). Synthesis of autonomous robots through evolution. *Trends in cognitive sciences*, 6(1):31–37.

Pfeifer, R. and Bongard, J. (2006). *How the body shapes the way we think: a new view of intelligence*. MIT press.

Pfeifer, R. and Gómez, G. (2009). Morphological computation–connecting brain, body, and environment. In *Creating Brain-Like Intelligence*, pages 66–83.

Sims, K. (1994a). Evolving 3d morphology and behavior by competition. *Artificial life*, 1(4):353–372.

Sims, K. (1994b). Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 15–22. ACM.

Stanley, K. O. (2007). Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162.

Trivedi, D., Rahn, C. D., Kier, W. M., and Walker, I. D. (2008). Soft robotics: Biological inspiration, state of the art, and future research. *Applied Bionics and Biomechanics*, 5(3):99–117.

Wilson, M. (2002). Six views of embodied cognition. *Psychonomic bulletin & review*, 9(4):625–636.

233

Material properties affect evolution's ability to exploit morphological computation in growing soft-bodied creatures

Francesco Corucci[*1,3], Nick Cheney[2,3], Hod Lipson[4], Cecilia Laschi[1] and Josh C. Bongard[3]

[1]The BioRobotics Institute, Scuola Superiore Sant'Anna, Pisa, Italy
[2]Department of Biological Statistics and Computational Biology, Cornell University, Ithaca, NY, USA
[3]Morphology, Evolution & Cognition Lab, University of Vermont, Burlington, VT, USA
[4]Creative Machines Lab, Columbia University, NY, USA
*f.corucci@sssup.it

Abstract

The concept of morphological computation holds that the body of an agent can, under certain circumstances, exploit the interaction with the environment to achieve useful behavior, potentially reducing the computational burden of the brain/controller. The conditions under which such phenomenon arises are, however, unclear. We hypothesize that morphological computation will be facilitated by body plans with appropriate geometric, material, and growth properties, while it will be hindered by other body plans in which one or more of these three properties is not well suited to the task. We test this by evolving the geometries and growth processes of soft robots, with either manually-set softer or stiffer material properties. Results support our hypothesis: we find that for the task investigated, evolved softer robots achieve better performances with simpler growth processes than evolved stiffer ones. We hold that the softer robots succeed because they are better able to exploit morphological computation. This four-way interaction among geometry, growth, material properties and morphological computation is but one example phenomenon that can be investigated using the system here introduced, that could enable future studies on the evolution and development of generic soft-bodied creatures.

Figure 1: A soft (a-d) and stiff (e-h) robots evolved to grow towards two lateral light sources. Red voxels expand in response to environmental stimuli, blue ones shrink. While the soft robot only employs expanding voxels and effectively exploits morphological computation, passive dynamics, and the interaction with the environment to solve the task, the stiff one is prevented from doing so due to its unsuitable material properties, and had thus to evolve a more complex and active form of control in order to achieve the same result. See them in action at: `https://youtu.be/Cw2SwPNwcfM`

Introduction

Evolving complete and intelligent artificial creatures is one of the long-term goals of artificial life and evolutionary robotics researchers. More than two decades after the first pioneering attempts (Sims, 1994), we are still far from matching the complexity exhibited by even the simplest organisms. Nevertheless, many insights have been gained to date, and many limitations overcome.

Hand in hand with similar developments in robotics (Kim et al., 2013; Rus and Tolley, 2015), substantial steps forwards in the complexity and interestingness of evolved virtual creatures have been recently made, by allowing evolution to make use of soft materials (Hiller and Lipson, 2010, 2012; Joachimczak and Wróbel, 2012; Joachimczak et al., 2014; Cheney et al., 2013; Rieffel et al., 2014; Lessin and Risi, 2015). In addition to enhancing morphological and behavioral diversity, the use of soft materials allows morphologies that more closely mimic biological ones, thus enabling

the investigation of additional aspects of evolution and development that were previously beyond reach. Here we focus on two such aspects relevant to soft-bodied creatures.

The first regards morphological plasticity: the ability to change some aspects of the body during one's lifetime. Here we investigate environment-mediated morphological development — growth in response to environmental stimuli — referred to henceforth simply as growth. Although it has been shown that morphological growth can provide adaptive advantages for machines (Bongard, 2013), previous work only focussed on rigid-bodied agents and environment-insensitive growth processes. Yet there is evidence that biological development is influenced and driven by the environment. For example, plant roots follow gradients of nutrients in the soil, while human bones and tissues alter their properties in response to mechanical loading (Wolff, 1986). Moreover, when compared to rigid-bodied

creatures, soft ones more naturally allow for some forms of morphological plasticity, that are already within the reach of soft robotics technology as well. Despite that, probably due to the lack of a general understanding of why, when, and how these new capabilities should be exploited, these robots feature to date basic forms of morphological plasticity (Shepherd et al., 2011; Corucci et al., 2015b,c), or no plasticity at all (Calisti et al., 2015; Corucci et al., 2015a; Cacucciolo, Corucci et al., 2014).

The second aspect we explore is the influence of material properties on the evolution and development of adaptive behavior. The behavior of soft-bodied creatures is to a large extent determined by their material properties, yet in soft robotics these are often fixed a priori and for the whole lifetime of an agent, perhaps after a limited number of heuristic tests. Some recently proposed ideas suggest that those properties — and softness in particular — can have implications for the development of intelligent behavior (Pfeifer and Bongard, 2006), but few studies (Nakajima et al., 2013) and theoretical frameworks (Hauser et al., 2011, 2012) elucidate and quantify these implications, typically not embracing an evolutionary and developmental approach.

Here we investigate the evolution and development of soft robots. In this context we hypothesize that with the right combination of geometry, material properties, and growth process, a robot can exploit morphological computation (Paul, 2006; Hauser et al., 2014) better than one for which one or more of these aspects is not well adapted. We test this hypothesis by evolving the body plans and developmental trajectories of simulated soft robots for a phototaxis task. Two variations of such robots are evolved: softer and stiffer ones. We find that the former achieve better performances, despite their evolved growth processes being simpler, according to an information theoretic measure. By interpreting environment-mediated growth as a form of control, it is suggested that the softer robots are in fact exploiting more morphological computation than the stiffer ones. This hypothesis is but one of many that can be tested using the system here introduced. These include the investigation of general relationships among morphology, control, evolution, and development. Such studies could provide a deeper comprehension of biological systems, with potential implications for the development of more complex, autonomous and adaptive machines.

Methods

Simulated task and environment. In this work soft-bodied creatures are simulated in the VoxCad environment (Hiller and Lipson, 2014). A number of changes and new features have been introduced in the simulator in order to enable our experiments.

First, sources of environmental stimuli can now be added to the environment. These sources are characterized by a fixed 3D location, and robot's voxels can sense the distance from each of them.

Second, the base of each robot is fixed to the ground for the entire simulation. If a robot does not touch the ground at the beginning of the simulation, it is translated along the z axis before the simulation starts, until it does. This is done to put emphasis on growth and deformability, ruling out locomotion strategies to approach the light sources.

Third, differently from other works adopting VoxCad (Cheney et al., 2015, 2013; Methenitis et al., 2015), in this experiment there is no fast-twitch actuation mechanism (i.e. the fast control based on an oscillating global signal is disabled). A distributed growth mechanism has been implemented instead, acting at a slower time scale (more below).

The task is inspired by plants, and consists in performing stationary phototaxis: growing towards static sources present in the environment. While reaching a single source is not a particularly difficult task, simultaneously pointing toward multiple ones becomes more challenging, as it requires the ability to evolve modular, branching structures. The specific growth mechanism is detailed in the next sections, as well as the underlying developmental paradigm.

Developmental paradigm. Different approaches have been adopted in the literature in order to model developmental processes (Stanley and Miikkulainen, 2003). Those can be roughly classified based on the level of abstraction with respect to the biological phenomenon they try to capture. Among high-level abstractions we find grammar-based approaches (Rieffel et al., 2014; Hornby and Pollack, 2002) and CPPN-based ones (Stanley, 2007; Cheney et al., 2013; Auerbach and Bongard, 2014). Lower-level abstractions, broadly referred as *cell chemistry* approaches (Doursat et al., 2013; Joachimczak et al., 2014; Bongard and Pfeifer, 2003), model finer details of developmental processes, such as gene regulatory dynamics.

Despite many achievements in cell chemistry, a drawback of these approaches lies in their complexity. On the other hand, when adopting a high-level perspective, the risk exists of overlooking potentially useful aspects of development. As an example, CPPNs and grammatical encodings neglect both the unfolding over time of biological developmental processes as well as the interaction of the creature with the environment during those processes.

The approach proposed here is based on CPPNs, but empowers them by joining their ability to capture the formation of regular patterns (Stanley, 2006) with an environment-mediated developmental stage that unfolds over time. While the implications of such a choice deserve to be thoroughly investigated in future work, we note that this approach enables potentially interesting feedback loops during development: growth is guided by environmental stimuli, but modifies in turn the sensory information the creature will experience next. Also, as mutation effects may arise at different points during development, the ability to enact changes later
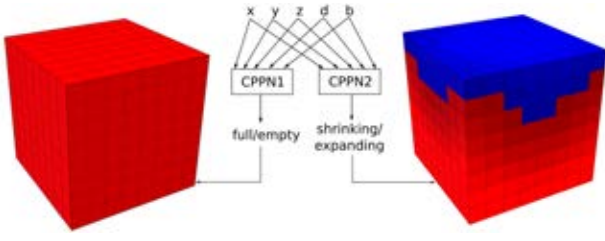
Figure 2: Different attributes of the robot can be "painted" by different CPPNs. CPPN1 dictates the geometry of the robot, while CPPN2 determines its growth properties. In the current system red voxels expand in response to environmental stimuli, while blue ones shrink.

in development may allow for smoother fitness gradients than all-or-nothing mutations (Hinton and Nowlan, 1987).

Time scales. The proposed developmental paradigm embeds all three time scales experienced by living systems: the evolutionary/phylogenetic time scale, the developmental/ontogenetic time scale, and the sensorimotor dynamics timescale (Pfeifer and Bongard, 2006). The sensorimotor timescale is here represented by the interaction of the body with the environment during growth (e.g. gravity, collisions, detected light levels, etc.).

The genotype. The encoding here adopted is based on CPPNs (Stanley, 2007). Designed to capture the formation of regular patterns in developmental systems without modeling development per se (Stanley, 2006), CPPNs are networks that convolve incoming spatial information to produce outputs that tend to exhibit symmetry, repetition, and repetition with variation.

In order to enforce a distinction between geometric and growth properties, here we adopt two different CPPNs (Fig. 2) that are queried for each voxel of a cubic workspace. They receive the same inputs: the 3D location of the voxel (x, y, z), the polar radius (d), and a constant bias (b).

The first CPPN, determining the creature's geometrical structure, has a single output $o \in [-1, 1]$ that dictates whether a voxel should be empty $(o < 0)$ or filled $(o \geq 0)$. Differently from similar setups (Cheney et al., 2013), evolution is provided with a single material in the experiments here reported, that is assigned to all non-empty voxels. A different stiffness can be specified for this material in different evolutionary runs, though all voxels in each run share a global and constant stiffness parameter.

The second CPPN determines the growth properties of each voxel. It also has a single output $\in [-1, 1]$ henceforth referred to as the *growth rate* $(g_{\rm rv})$, whose role is described in detail in the next section.

	Description
$D_{\rm v}^{(t)}$	Linear dimension of voxel v at time t
$s_{\rm v}^{(t)}$	Scaling factor of voxel v at time t
$D_{\rm vn}$	Nominal dimension of voxel v (set to 1)
$g_{\rm a}$	Growth amplitude $\in (0, 1)$ (equal $\forall$ voxel, set to 0.5)
N	Number of environmental sources
i	A specific source
$s_{\rm vi}^{(t)}$	Influence of source i on the scaling factor $s_{\rm v}^{(t)}$
$s_{\rm v\_min}$	Lower bound for $s_{\rm v}^{(t)}$ (set to 0.1)
$g_{\rm rv}$	Growth parameter of voxel v ($\in [-1, 1]$)
$\overline{d_{\rm i}}^{(t)}$	Distance of voxel v from source i, normalized by voxel size (the latter being set to 0.01)

Table 1: Description of parameters appearing in Eq. 1

Environment-mediated morphological development. We simulate environment-mediated growth by enabling voxels to change volume in response to environmental stimuli (i.e. distance from each light source). This choice is motivated by the fact that localized volumetric changes are easy to achieve with currently-available soft robots (e.g. exploiting pneumatic actuation). A localized change in stiffness is also within the reach of current technology (Majmudar et al., 2007), making this kind of plasticity the next candidate to be integrated into our system. Topological modifications are, on the other hand, more difficult to achieve in real soft robots, as they require adding or removing material. Nevertheless, attempts have been made in this direction as well (Brodbeck et al., 2012).

The growth process is governed by equations and parameters reported in Eq. 1 and in Tab. 1:

$\forall$ voxel v, at each time step t

$$D_{\rm v}^{(t)} = s_{\rm v}^{(t)} \cdot D_{\rm vn}$$
$$s_{\rm v}^{(t)} = \max \left[g_{\rm a} \cdot \tanh \left(\sum_{i=1}^{N} s_{\rm vi}^{(t)} \right) + 1, s_{\rm v\_min} \right] \quad (1)$$
$$s_{\rm vi}^{(t)} = g_{\rm rv} \cdot \overline{d_{\rm i}}^{(t)}$$

The *growth rate* parameter $g_{\rm rv}$ determines the quality and the extent of the localized volumetric change for each voxel. When $g_{\rm rv} > 0$ the voxel will expand when close to a source, when $g_{\rm rv} < 0$ it will shrink. When $g_{\rm rv}$ is exactly zero, the voxel is insensitive to environmental stimuli. The greater the magnitude of $g_{\rm rv}$ for a particular voxel, the more pronounced will be its volumetric variation, that is also modulated by the distance from the sources. Each voxel can experience a considerable modification due to development: having set

$g_a = 0.5$ entails a 50% linear contraction/expansion with respect to the nominal size, resulting in a $\sim 238\%$ variation in volume. The parameter s_{v_min} ensures that the voxel does not shrink below a given size (here the 10% of the nominal size), for stability of simulations. The quantity Δs_{v_max} (not reported in Eq. 1 for ease of reading) dictates the maximum allowed $\Delta s_v = |s_v^{(t)} - s_v^{(t-1)}|$ between two subsequent time steps, as follows:

$$\text{if } (\Delta s_v > \Delta s_{v\_max}) \text{ then:}$$
$$s_v^{(t)} = s_v^{(t-1)} + \text{sign}(\Delta s_v) \cdot \Delta s_{v\_max}$$

In addition to influencing the stability of the simulation, this parameter (set to 0.0005 in our experiments) regulates the speed of the growth process: the higher Δs_{v_max}, the more rapid the growth. The value for Δs_{v_max} was selected in such a way that development acts over a slower time scale with respect to the typical sensorimotor dynamics (such as those generating locomotion or grasping behavior), thus implementing the developmental time scale.

Development is based on distributed sensing and actuation: each voxel senses the distance from all the sources and acts accordingly. Nevertheless, coordinated behavior emerges, for at least two reasons: first, nearby voxels experience similar sensory stimuli, and second, CPPNs tend to produce patches of tissue with homogeneous or smoothly varying growth parameters.

Optimization. A multi-objective implementation of NEAT (Cheney et al., 2015) has been adopted. Before performing selection, pareto ranking is applied, according to three objectives: the order in which sorting is performed determines the relative importance of each of them. The objectives are listed below from the most to the least important:

1. Minimize the distance from each of the sources
2. Minimize the number of employed voxels
3. Minimize the age of each individual

The first objective selects for phototaxis. This is implemented by minimizing the sum $\sum_{i=1}^{N} d_{min\_i}$, where N is the number of sources and d_{min_i} is the minimum distance between the robot and the i-th source. The second objective selects for smaller robots. The first two objectives are antagonistic as it is easier, in general, for larger robots to be closer to the sources (even if they do not grow at all). The combination of the two objectives thus selects for robots that exploit the growth process and their deformability to accomplish the task. The third objective helps maintain diversity in the population (Schmidt and Lipson, 2011).

Morphological computation and control complexity. Morphological computation (Hauser et al., 2014) has been defined as "computation obtained through the interaction of physical forms" (Paul, 2006). When it comes to robotics and embodied cognition (Pfeifer and Bongard, 2006), the idea is that part of the computation needed to perform a task can take place (implicitly or explicitly) not only in the brain/controller, but also within the body itself, provided that it has suitable characteristics. It has been argued that this property can alleviate the computational burden of the brain, simplifying the controller and achieving a more balanced brain-body trade-off (Pfeifer and Bongard, 2006; Paul, 2006; Hauser et al., 2011) that could hold the key to more intelligent, effective and natural behaviors. Many examples have been described in the literature (Pfeifer and Bongard, 2006). It is often postulated that systems benefiting from morphological computation tend to exploit the interaction and dynamical coupling with the environment in a beneficial way, e.g. leveraging passive dynamics in place of active control.

We hypothesize that material properties can affect evolution's ability to exploit morphological computation. To test this, we here define morphology as the robot's shape and material properties, and its 'controller' as the distributed growth mechanism which achieves phototaxis. We defend this latter definition as, like control, growth here closes the sensation-action feedback loop (although over a slower time scale).

For our purposes, we can thus define morphological computation as a property that simplifies the growth controller by exploiting in a beneficial way the interplay between material, geometric, and growth properties through the dynamical interaction with the environment.

To measure the extent of morphological computation in a given robot, we define control complexity as follows:

$$H(g_c) = -\sum_{i=1}^{n} p_i \log_2 p_i \tag{2}$$

where:

$$\begin{aligned} g_c &= \{g_{rv} \; \forall \text{ voxel } v\} \\ p_i &= \int_{x_i}^{x_{i+1}} p(x) \, dx \\ x_i &= -1 + 0.02i \qquad i = 0 \dots 100 \end{aligned} \tag{3}$$

The real-valued random variable g_c is associated to the *growth rate* quantity (Eq. 1, Tab. 1), embracing all parameters that collectively shape the growth trajectory of a given robot. The quantity $H(g_c)$ is the Shannon entropy (Shannon, 1948) of such a variable, whose probability density function $p(x)$ is discretized using $n = 101$ uniform bins (Eq. 3).

The control complexity of g_c thus corresponds to the number of bits that are necessary to describe the pattern of growth parameters: the higher this number, the more complex the controller. Consider two robots (r_1, r_2) and their associated growth controllers (g_{c1}, g_{c2}). We will state that a difference $\Delta H = H(g_{c2}) - H(g_{c1}) > 0$ between the

two controllers indicates that g_{c2} is more complex than g_{c1}. Moreover, if the two robots happen to score the same fitness, we will argue that r_1 better exploits morphological computation, as it requires simpler control to produce an equally effective behavior. It should be made clear that we are not providing here a general information theoretic metric to capture morphological computation (Zahedi and Ay, 2013), but rather a proxy to measure its effect in our setting.

Experiments Populations composed of 30 individuals are allowed to evolve for a maximum of 1500 generations. The maximum evaluation time for each individual is 3.5s (simulation time, wall time is higher). The simulation is stopped earlier if the robot settles into a static conformation before the allocated time elapses. The growth process starts after the first 0.5 seconds, which usually allows the initial shape to settle into an equilibrium position.

A first set of experiments is performed to qualitatively assess the overall ability of the system to evolve effective robots. To this end, 10x10x10 and 8x8x8 robots have been evolved in several environments, differing in the number (from 1 to 4) and position of environmental sources (Fig. 3). Material stiffness is set to $E = 5\,MPa$, corresponding to a rather soft material (comparable to rubber). Five runs were performed for each configuration.

A second set of experiments is then performed with 6x6x6 robots, characterized by different stiffness values ($E_1 = 500\,MPa$, $E_2 = 5\,MPa$) evolving in the same environment (2 laterally placed sources). Twenty independent runs were performed for each treatment. Reported confidence intervals are computed with a bootstrapping method, while p-values are the result of the Mann-Whitney U test.

The code used to produce these results is publicly available at: `https://goo.gl/cA2luO`. A video showing some of the creatures in action is available at: **https://youtu.be/Cw2SwPNwcfM**

Results and Discussion

A sample of the fittest morphologies evolved in preliminary trials is reported in Fig. 3 and in the accompanying video. Symmetry and modularity can be observed, with the latter property evidenced by the emergence of relatively independent appendages. As these features are ubiquitous in natural systems, their presence here may suggest the potential for more competent and scalable virtual creatures. Moreover, these properties appear to be selected for by our task and environment, as this level of morphological regularity is not common in similar settings (Cheney et al., 2013, 2015; Methenitis et al., 2015).

It can be noted that the best individuals from these runs tend to only exploit expanding (red) tissue, and not shrinking (blue) voxels (Fig. 3). Evolved creatures appear able to leverage their passive deformability and interaction with the environment in order to solve the task, rather than requiring
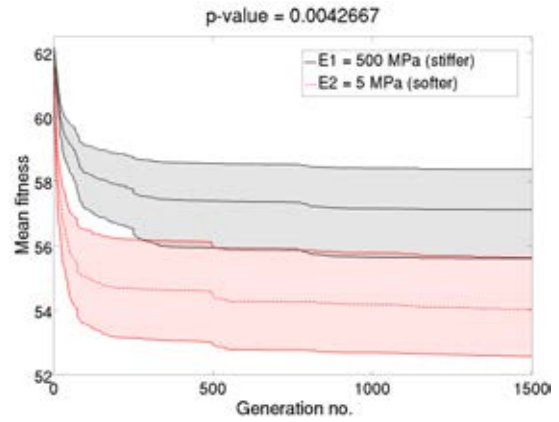
Figure 4: Average fitness over 20 independent runs. Softer robots have an evolutionary advantage over stiffer ones in this task/environment.

differential expansion and contraction to point towards light sources. For example, cantilevers spontaneously evolve (Fig 3b). Like human-built cantilevered structures, these robots distribute stresses across themselves with a minimum of support structure. Moreover, the curvature needed to point towards the two lateral sources spontaneously emerge from the passive interaction of the expanding body with the environment (and with gravity, in particular), rather than through internal actuation of the creature. This corresponds, intuitively, to the idea of morphological computation.

Given the scarce presence of highly-fit robots exploiting more complex forms of control — based on the combination of shrinking and expanding voxels — we hypothesize that it may be easier for evolution to discover solutions based on morphological computation rather than explicit control, provided that material properties allow it to do so. This would confer, in general, an evolutionary advantage to robots that have the "right" material properties for a given task/environment. This hypothesis is tested with the second set of experiments.

Geometry, materials, growth, and morphological computation. Fig. 4 reports the evolution of robots with stiffer (E_1) or softer (E_2) material properties, optimized to simultaneously approach two lateral light sources placed on opposite sides of the creature. Results show that softer robots have an evolutionary advantage over stiffer ones in this particular task/environment, which may be attributed to evolution's ability to better exploit morphological computation instead of developing more complex and active forms of control to regulate their shape. This can be qualitatively observed in the comparison of two highly-fit robots reported in Fig. 1 (see them in action in the accompanying video). With the softest material, evolution produced a passive cantilever, taking advantage of gravity to unfold the shape towards the two sources during growth. Under the effect of gravity, the
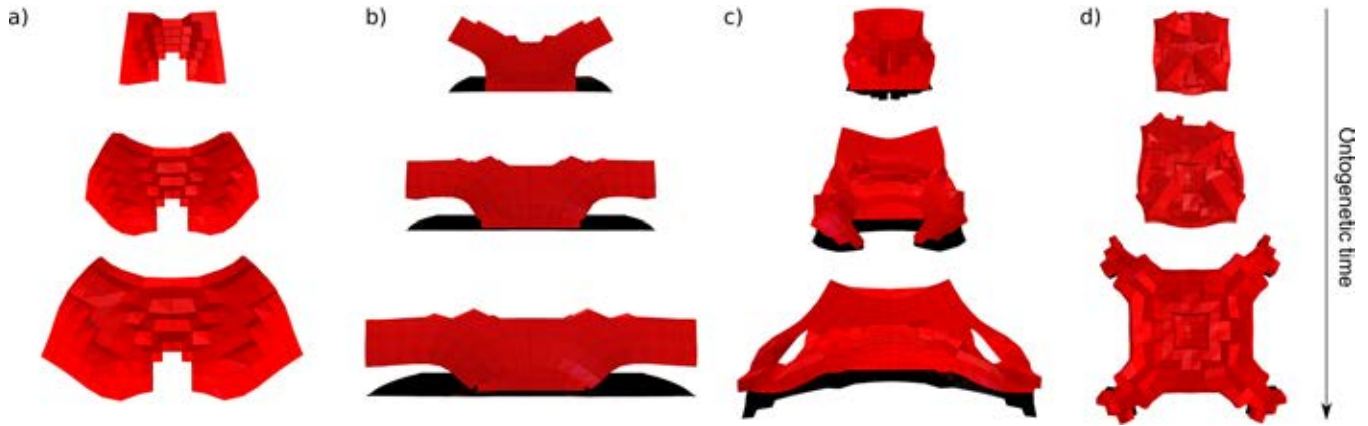
Figure 3: A small sample of the growing soft creatures, evolved in environments featuring: a-c) two lateral sources d) four sources placed at the corners. a,d) top view, b,c) front view. Most of the highly fit soft robots only exploit expanding tissue.

structure passively deforms and achieves an effective curvature, being able to sustain its own weight and point towards the sources at the same time. On the other hand, the stiffer robot fights gravity holding the two rigid arms horizontally, achieving the curvature needed to direct its appendages towards the sources through the antagonistic action of shrinking and expanding voxels in the central part of the body.

Further analyses suggest the generality of these observations in this task/environment. Figure 5 shows that stiffer robots use substantially more shrinking voxels in general. Their growth processes involve more active control in the sense that the appendages are pulled as well as pushed to achieve proximity to the environmental sources. This suggests that for stiffer robots, simpler strategies in which curvature is achieved passively in response to weight are either harder to find in the search space or are not viable at all.

Given the fitness benefit of expanding voxels, enlarging the volume of the robots and allowing them to approach the sources more closely, we would expect shrinking voxels only to be employed when necessary to control the direction of evolved appendages. Thus, the presence of more shrinking voxels in the stiffer robots suggest that they are unable to perform passive pointing from their material properties alone, as exemplified in the softer robots (Fig. 1).

The intuitive considerations regarding control complexity and morphological computation are also confirmed by an information theoretic analysis of the evolved robots: the average control complexity $H(g_c)$ (the global entropy across g_c) is significantly higher for the stiffer robots than for the softer ones (Fig. 6). In other words, stiffer robots employ more complex and active controllers. This again suggests that their morphologies are unable to perform the task to the same level without control.

In summation, softer robots better exploit morphological computation in this particular task/environment, achieving better performances (Fig. 4) with simpler controllers (Fig.
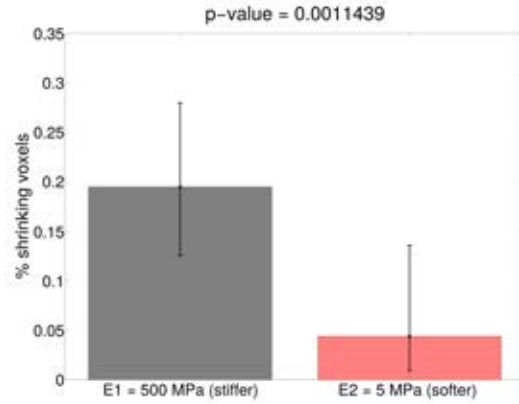
Figure 5: Stiffer robots tend to employ significantly more shrinking voxels than softer ones ($p < 0.002$), in the attempt to actively control the shape.

5, 6). This confers an evolutionary advantage to them over stiffer robots, if evolved alongside.

It should be noted that these results do not mean that softer is *always* better. It would be possible to define a task/environment that confers an advantage to stiffer robots (e.g. grow towards sources placed mid-air, where it is easier for stiffer robots to sustain their weight). What has been demonstrated is that for a specific task/environment, material properties can have a pronounced effect on evolution's ability to exploit morphological computation, i.e. to produce well adapted morphologies that exploit the interaction with the environment in a beneficial way.

Conclusions and Future Work

In this paper a novel system to study the evolution and development of soft-bodied creatures has been presented. The system is able to evolve robots that exhibit desirable morphological properties such as symmetry, modularity, and ex-
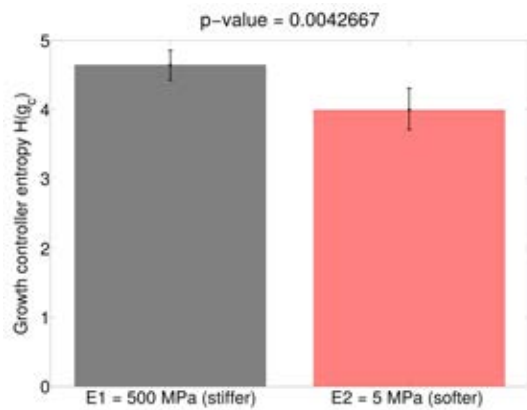
Figure 6: Stiffer robots exhibit more complex growth controllers than softer ones ($p < 0.005$), yet the latter achieve better performances (Fig. 4). It is argued that this difference is due to morphological computation, strictly connected to the material properties of robots in the two treatments.

ploitation of morphological computation. More specifically, it was shown that certain combinations of geometry, material properties, and environment-mediated growth make it more or less difficult for evolution to discover phenotypes that exploit morphological computation. Despite being a fundamental aspect of soft robotics, the interplay among these properties remains to date largely unexplored. Results also suggest that arbitrarily fixing even one of these dimensions can make it difficult for evolution to produce effective behaviors. Ideally, all of them should be put under evolutionary and/or developmental control, so that an optimal combination can be discovered. Future work will be directed towards exploring the potential evolutionary advantage of morphological plasticity, as well as possible benefits in terms of adaptivity and robustness. The environmental influence during development deserves special attention as well: its potential benefits for organisms as well as adaptive machines will be investigated. Another major topic that can now be studied is the general relationship between evolution, development and adaptive behavior. We believe that many interesting questions can be answered using the approach described here, and could help shed light on biological questions, while simultaneously contributing to engineering fields such as soft robotics.

Acknowledgments

References

Auerbach, J. E. and Bongard, J. C. (2014). Environmental influence on the evolution of morphological complexity in machines. *PLoS Comput Biol*, 10(1):e1003399.

Bongard, J. C. (2013). Evolutionary robotics. *Communications of the ACM*, 56(8):74–83.

Bongard, J. C. and Pfeifer, R. (2003). Evolving complete agents using artificial ontogeny. In *Morpho-functional Machines: The New Species*, pages 237–258. Springer.

Brodbeck, L., Wang, L., and Iida, F. (2012). Robotic body extension based on hot melt adhesives. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4322–4327. IEEE.

Cacucciolo, V.*, Corucci, F.*, Cianchetti, M., and Laschi, C. (2014). Evolving optimal swimming in different fluids: A study inspired by batoid fishes. In *Biomimetic and Biohybrid Systems*, pages 23–34. Springer.

Calisti, M., Corucci, F., Arienti, A., and Laschi, C. (2015). Dynamics of underwater legged locomotion: modeling and experiments on an octopus-inspired robot. *Bioinspiration & Biomimetics*, 10(4):046012.

Cheney, N., Bongard, J., and Lipson, H. (2015). Evolving soft robots in tight spaces. In *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*, pages 935–942. ACM.

Cheney, N., MacCurdy, R., Clune, J., and Lipson, H. (2013). Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. In *Proceedings of the 15th annual conference on Genetic and Evolutionary Computation*, pages 167–174. ACM.

Corucci, F., Calisti, M., Hauser, H., and Laschi, C. (2015a). Evolutionary discovery of self-stabilized dynamic gaits for a soft underwater legged robot. In *Proceedings of the 17th International Conference on the Advanced Robotics*, pages 337–344. IEEE.

Corucci, F., Calisti, M., Hauser, H., and Laschi, C. (2015b). Novelty-based evolutionary design of morphing underwater robots. In *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*, pages 145–152. ACM.

Corucci, F., Calisti, M., Hauser, H., and Laschi, C. (2015c). Shaping the body to shape the behavior: a more active role of the morphology in the brain-body trade-off. *Late Breaking Proceedings of the 13th European Conference on Artificial Life*, pages 7–8.

Doursat, R., Sayama, H., and Michel, O. (2013). A review of morphogenetic engineering. *Natural Computing*, 12(4):517–535.

Hauser, H., Füchslin, R. M., and Pfeifer, R. (2014). *Opinions and Outlooks on Morphological Computation*. Zürich.

Hauser, H., Ijspeert, A. J., Füchslin, R. M., Pfeifer, R., and Maass, W. (2011). Towards a theoretical foundation for morphological computation with compliant bodies. *Biological cybernetics*, 105(5-6):355–370.

Hauser, H., Ijspeert, A. J., Füchslin, R. M., Pfeifer, R., and Maass, W. (2012). The role of feedback in morphological computation with compliant bodies. *Biological cybernetics*, 106(10):595–613.

Hiller, J. and Lipson, H. (2012). Automatic design and manufacture of soft robots. *Robotics, IEEE Transactions on*, 28(2):457–466.

Hiller, J. and Lipson, H. (2014). Dynamic simulation of soft multimaterial 3d-printed objects. *Soft Robotics*, 1(1):88–101.

Hiller, J. D. and Lipson, H. (2010). Evolving amorphous robots. In *Proceedings of the 12th International Conference on the Synthesis and Simulation of Living Systems*, pages 717–724.

Hinton, G. E. and Nowlan, S. J. (1987). How learning can guide evolution. *Complex systems*, 1(3):495–502.

Hornby, G. S. and Pollack, J. B. (2002). Creating high-level components with a generative representation for body-brain evolution. *Artificial life*, 8(3):223–246.

Joachimczak, M., Suzuki, R., and Arita, T. (2014). Fine grained artificial development for body-controller co-evolution of soft-bodied animats. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 239–246.

Joachimczak, M. and Wróbel, B. (2012). Co-evolution of morphology and control of soft-bodied multicellular animats. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 561–568. ACM.

Kim, S., Laschi, C., and Trimmer, B. (2013). Soft robotics: a bioinspired evolution in robotics. *Trends in biotechnology*, 31(5):287–294.

Lessin, D. and Risi, S. (2015). Soft-body muscles for evolved virtual creatures: The next step on a biomimetic path to meaningful morphological complexity. In *European Conference on Artificial Life*.

Majmudar, T., Sperl, M., Luding, S., and Behringer, R. P. (2007). Jamming transition in granular systems. *Physical review letters*, 98(5):058001.

Methenitis, G., Hennes, D., Izzo, D., and Visser, A. (2015). Novelty search for soft robotic space exploration. In *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*, pages 193–200. ACM.

Nakajima, K., Hauser, H., Kang, R., Guglielmino, E., Caldwell, D. G., and Pfeifer, R. (2013). A soft body as a reservoir: case studies in a dynamic model of octopus-inspired soft robotic arm. *Front. Comput. Neurosci*, 7(10.3389).

Paul, C. (2006). Morphological computation: A basis for the analysis of morphology and control requirements. *Robotics and Autonomous Systems*, 54(8):619–630.

Pfeifer, R. and Bongard, J. (2006). *How the body shapes the way we think: a new view of intelligence*. MIT press.

Rieffel, J., Knox, D., Smith, S., and Trimmer, B. (2014). Growing and evolving soft robots. *Artificial life*, 20(1):143–162.

Rus, D. and Tolley, M. T. (2015). Design, fabrication and control of soft robots. *Nature*, 521(7553):467–475.

Schmidt, M. and Lipson, H. (2011). Age-fitness pareto optimization. In *Genetic Programming Theory and Practice VIII*, pages 129–146. Springer.

Shannon, C. (1948). A mathematical theory of communication, bell system technical journal 27: 379-423 and 623–656. *Mathematical Reviews (MathSciNet): MR10, 133e*.

Shepherd, R. F., Ilievski, F., Choi, W., Morin, S. A., Stokes, A. A., Mazzeo, A. D., Chen, X., Wang, M., and Whitesides, G. M. (2011). Multigait soft robot. *Proceedings of the National Academy of Sciences*, 108(51):20400–20403.

Sims, K. (1994). Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 15–22. ACM.

Stanley, K. O. (2006). Exploiting regularity without development. In *Proceedings of the AAAI Fall Symposium on Developmental Systems*, page 37. AAAI Press Menlo Park, CA.

Stanley, K. O. (2007). Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162.

Stanley, K. O. and Miikkulainen, R. (2003). A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130.

Wolff, J. (1986). *The law of bone remodelling*. Springer Science & Business Media.

Zahedi, K. and Ay, N. (2013). Quantifying morphological computation. *Entropy*, 15(5):1887–1915.

Evolution-in-Materio of a dynamical system with dynamical structures

Odd Rune Lykkebø[1], Gunnar Tufte[1]

[1]Norwegian University of Science and Technology
lykkebo@idi.ntnu.no, gunnart@idi.ntnu.no

Abstract

Evolution-in-Materio aims to exploit real-world physics of materials to achieve computation by a combination of external stimulus and interpretation of the state of materials through measurements and observations. In a majority of Evolution-in-Materio work the dynamics of the material is filtered out, or the problem is defined in a way that the sought solution is a point attractor. In this work we explore the dynamics of materials. Within the assumption that suited materials include rich behavior emerging from the underlying physical processes there should be observable behavior similar to Dynamical Systems with Dynamical Structures $((DS)^2)$. Such behavior result in systems with a possibility of inducing perturbations to their own dynamics. Further, the importance of the *observation level* used when observing and interpreting the state of the materials is discussed and related to dynamics in Evolution-in-Materio systems.

Introduction

Evolution-In-Materio (Miller et al., 2014) (EIM) can be seen as a method to explore unconventional computation, i.e. a computer operating outside of the traditional Turing/von Neumann (Turing, 1937; von Neumann, 1993) computational model and architecture, exploiting the power of evolution, i.e. Computer Controlled Evolution (CCE) to manipulate a physical system to search for regimes where the intrinsic properties of materials provide useful computation.

The concept of EIM is, as stated by Miller et al. (2014): "to exploit the intrinsic properties of materials, or "computational mediums", to do computation, where neither the structure nor computational properties of the material needs to be known in advance (Miller and Downing, 2002). In this way evolution is a bottom-up design process that can exploit natural physical processes to do useful computation."

Herein the bottom-up design concept is further investigated so as to gain a deeper insight toward exploiting physical systems such as materials for computation: "where neither the structure nor computational properties of the material needs to be known in advance". Both the structure and the intertwined underlying physics leading to computation are products of the bottom-up design approach taken.

When a bottom-up process such as evolution acts on physical systems, there may be intrinsic processes in the computational medium that influence the computational function as a result of underlying properties resulting in a non-static structure and thereby non-static functionality, i.e a two way coupling between dynamic structure and functionality. A system with the possibility of inducing perturbations to their own dynamics as a function of their system states, enables state space trajectory changes and topological reconfigurations of the state space (Omholt, 2013).

In this paper we show that such behaviours, present in living systems, can also be found in EIM systems making EIM a physical realisations of Dynamical Systems with Dynamical Structures $((DS)^2)$ (Spicher et al., 2004). In such systems state transition functions and the set of state variables can change over time.

Reaction diffusion systems as in Adamatzky (2009) work exploit massively parallelism of state updates in growing patterns where information processing take place. The reaction-diffusion computer is based on local interactions and change of spatial properties over time. Growth of patterns change the state and topological properties of the machine providing a similarity to the $((DS)^2)$ dynamical reconfigurations of the state space.

Further, it can be useful to compare the concept of EIM with morphological engineering (ME) Doursat et al. (2013). Central to ME is the concept of *agents* and the mechanisms involved in controlling them. EIM places less focus on the interaction of the individual agents, and more focus on exploitation of the emergent behaviours of the physical systems under study. Though local interaction between smaller parts of the physical matter (i.e. electro-chemical interaction between molecules or concentrations of chemicals) is crucial, EIM attempts to take a step away from the agent-focus and consider emergent properties a primary unit of study. However, one of the central aspects of ME, i.e. "endowing physical systems with information" Doursat et al. (2013) does capture EIM. 'Invisible' dynamics of physical systems can be manipulated without direct access to the individual agents' rule sets by giving the systems the ability to filter or

react to information and energy presented through physical interfaces.

In this article, an EIM system is observed in a $(DS)^2$ system view. The results show that such a view is applicable and can be used to gain further insight in the working of EIM systems. As a result EIM may share several properties with self-organizing systems. Herein artificial developmental systems (Kumar and Bentley, 2003) are used as a starting point for exploring and exploiting computational mediums where computation is a product of underlying dynamical physical processes.

An implication of "exploiting the intrinsic properties of materials" is that self-organizing processes and the resulting computation do not necessary comply with our mostly used computational paradigm of digital computers. EIM can be seen as a hybrid variant of Analog Computation (AC) (MacLennan, 2007). In AC, the fact that a mathematical function provides a model of the *observed* behaviour of a physical system is used 'conversely'; a physical system can be used to calculate a mathematical function since a physical system can be parameterized and adjusted to match a large number of mathematical functions. Configurable Analogue Processors (CAP) or Field Programmable Matter Arrays (FPMA) (Miller and Downing, 2002) are recent example of this paradigm. The hybrid approach of EIM include the computational matter, e.g. CAP or FPMA, in a mixed signal system using a digital computer to configure and communicate with the material.

This is an approach that enables the computational power of the material with the ease of programmability of digital computers (Broersma et al., 2012). In a hybrid approach observability is a core issue. Ensuring that the data from the material are observable and sound without using more computational power for the observation then the actual computation (Bremermann, 1962). A practical implication of observability is that there is a need for choosing what the smallest possible change is to be on the top-level. Top-level being the level that the observable computational result(s) emerge from the underlying physical processes.

Shannon (1941) provided an early theoretical model of analogue computers, the General Purpose Analog Computer (GPAC). Though application of the GPAC model on EIM seems un-natural since it relies on chaining together components (reminiscent of agents with particular tasks) and defining bigger expressions out of this. Neural networks is yet another potential model and link between the physical EIM system and a theoretical model– this link is further discussed in Broersma et al. (2012).

Figure 1 shows an EIM system using a digital computer to host an EA to configure a material for computation. The material operates in the analogue/physical domain and the computer responsible for input/output mapping and configuration will operate in the digital domain.

As stated, our focus is to explore EIM in a $(DS)^2$ view.
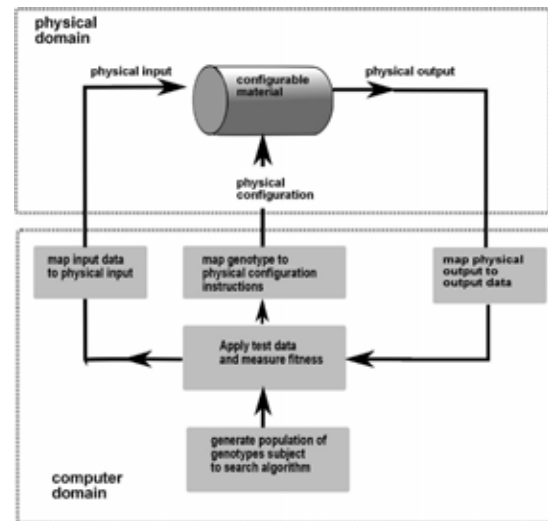
Figure 1: Principle of EIM using a hybrid approach. Taken from Miller et al. (2014).

In Figure 1 the configurable material is the self-organising system. The material inhibits dynamical properties and respond to perturbations from the input and configuration signals. To observe dynamical properties, the trajectory of the system is used as a quantifiable measurement of behaviour. In artificial development similar measurements have been used as a measurement of evolved complexity (Nichele and Tufte, 2013) and to show intertwined influence of structure and computation $((DS)^2)$ on artificial developmental systems (Tufte, 2009).

The systems herein are all observed at an electrical level. The underlying physical (and electrical) processes may change over time on the microscopic level, however our observations is on the 'digital level'. As such, the goal herein is to be able to exploit the power of EIM without a high cost in ensuring the correctness of the observation, i.e. a underlying rich physical system observed in the constrained digital domain.

Background

Evolution-in-materio extended to $(DS)^2$

Gordon Pask's pioneering work in EIM(Pask, 1959) is an interesting piece of work if viewed in a $(DS)^2$ setting. Pask observed (by eye) and evaluated (by ear) the growth of neural like structures using an electrochemical device made of a dish with electrodes covered by a metal salt solution. By adjusting the current between electrodes Pask was able to grow iron connections that responded to different frequencies. Pask's system show the principles of a $(DS)^2$ EIM system. The growth of the connections depend on current, when current passes the connection grow. By adjusting the current in different regions a structure emerges and are adjusted toward the evaluator's (here Pask) goal. The
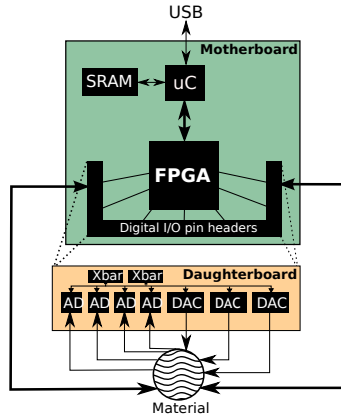
(a) Block diagram of the Mecobo hardware interface.

(b) Picture of the Mecobo motherboard with mixed signal daughter board.

(c) The material bay filled with salt crystals (red arrow).

Figure 2: Overview of the Mecobo hardware interface.

growth of the connections are governed by self-organization and perturbations. The system is constantly changing making the response to perturbations depending on the systems present state. In Pask's work this property of change by self-organization and perturbations was exploited toward the goal of growing "an artificial ear".

In the work of Clegg et al. (2014) a material of Carbon Nano Tubes (CNT) was exploited to solve the travelling salesperson problem by evolving statical configurations that manage to solve the problem. i.e. the material's dynamic properties was filtered out by allowing the system to reach a steady state, or point attractor. Thompson's experiments using a Field Programmable gate Array (FPGA) as a material (Thompson et al., 1999) evolved a static configuration defining the internal circuit architecture. The input signal (similar to Pask's) was two frequencies that changed the output value depending on the input frequency (a frequency discriminator). The static configuration in Thompson's FPGA experiments does not allow for dynamic structures. The input signal perturbed the system to an binary observable output. However, the underlying dynamics (internal state transitions) for the system was untested.

In Harding and Miller's EIM work utilizing Liquid Crystal Displays (LCDs), the material is viewed as a type of static device with an evolved configuration that enable the LCD to compute, e.g. a frequency discriminator (Harding and Miller, 2004). However, the LCD is not a static device. The behaviour change if it is disconnected and reconfigured. To regain previous behaviour a short re-evolution was required.

Systems that explicitly exploit dynamic structures such as slime moulds (Adamatzky, 2016) or the combination of CNTs and liquid crystals (Massey et al., 2015b) show a

clear connection between dynamic structure and computation similar to a $(DS)^2$ system.

As in Pask's work, the work on explicit dynamic structures and the shown change in response on configuration for the LCD it can be argued that a closer look at EIM systems as a $(DS)^2$ system are fruitful as to achieve more complex behaviour in such computational medium. More over, including more knowledge of the underlying bottom-up processes $((DS)^2)$ enables an increased understanding and insight in evolutionary exploitation of EIM systems toward complex computational tasks.

Materials

Recently the NASCENCE (NAnoSCale Engineering for Novel Computation using Evolution) project (Broersma et al., 2012) has provided a variety of material samples based on CNTs (Massey, 2013), mix of liquid crystals and CNTs (Massey et al., 2015b) and gold nano particles (Boses et al., 2015) for EIM research. Studies of Single walled carbon nano tubes(SWCNT) (Massey, 2013; Massey et al., 2015a) have shown that these CNTs have novel electrochemical properties and have the potential to do computation (Massey et al., 2015b).

The results of NASCENCE (e.g. (Mohid et al., 2014; Clegg et al., 2014; Massey, 2013)) have shown that computational results can be achieved without explicitly exploiting the dynamics of physical systems. Here the exploration of $(DS)^2$ expand our target behaviour of materials to a physical system with rich dynamic properties and complexity at many scales, which is a point frequently brought up in the complex systems literature. Often one divides a system into two broad categories relative to the observation level or scale, (Sayama, 2015) (Bar-Yam, 1997) (Fromm, 2004). Mi-

croscopic properties or behaviours are those that potentially give rise to emergent macroscopic properties. At different levels of observation, different macroscopic behaviours (such as emergence and self organization) exist, and thus the complexity of a system depends on the level of observation; as Simon (1962) writes, "How complex or simple a structure is depends critically upon the way in which we describe it."

A core idea in EIM is that it is should be possible to manipulate the microscopic behaviour of the material by providing input and 'configuration' energy (which potentially affects all scales of the system) such that the emerging or self-organizing behaviour is both observable and useful in terms of computation.

To further drive the point of the richness the materials used in the experiments are chosen to be very different. Single walled carbon nano tubes in a static physical configuration, i.e. electrical charge change cause dynamics, used and exploited for computation within the NACSENCE project show electrical observable response from underlying electrical networks exploited by evolution to emerge at the macroscopic level. As a second material common kitchen table salt, a material that is conductive when mixed with water. The crystalline form of salt is for the most part nonconductive since the ions are bound up in a crystal lattice structure, but by adding a tiny amount of water to the crystals conductivity is achieved. The salt structure is in contrast to the CNT material not static.

Method

The overall goal of EIM is to find methods and materials that can serve as complex computing systems. Methods should be capable of exploring and exploiting materials toward achieving useful computation, and materials should have inherent properties that enable this. The systems are physical systems, hence the material will operate in a real environment, and in particular, the environment will be part of the system. The main method is the bottom-up design approach of evolution. In a complexity setting evolution is argued to be a process that builds complexity (Holland, 2012). The building of complexity can also be considered as a case where the system learns to program itself, third of Brian Arthur's methods of complexity growth (Arthur, 1993). Further, the concept of growth of complexity fits well into a $(DS)^2$ setting; a system that evolve toward more complex dynamic behaviour by perturbations from the environment and the dynamics of the system itself (Nichele and Tufte, 2013).

An experimental approach is taken to investigate and explore the relation between EIM and $(DS)^2$. Exploiting evolution to configure materials with a behaviour that show induced perturbations to it's own dynamics. The experimental setting is in principle as shown in Figure 1. The experiments are designed to unveil the intertwined influence between the dynamics of underlying physics the input data and configu-
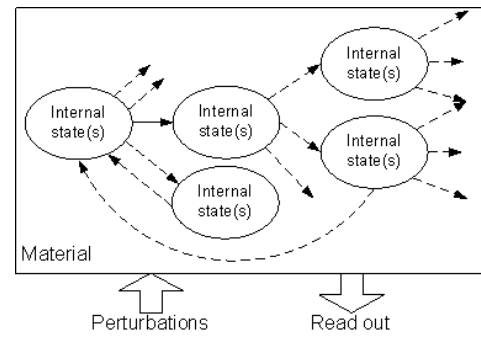
Figure 4: System view of the experiments. The material is considered as a $(DS)^2$ system capable of inducing perturbations to it's own dynamics. The external perturbations indicated as an input arrow include input data and configuration signals for the material. The output arrow indicates the external observation of the system state.

ration signals. To put the system in a $(DS)^2$ setting input and configuration signals are considered as external perturbations to the system. Figure 4 illustrates the experimental setting at a system level.

The material in the figure include a set of state transitions. The state transitions can be perturbed externally. Each state in the figure is observable through the read out signal. If the system inhibits $(DS)^2$ behaviour it should be possible to detect different trajectories (induced perturbations to it's own dynamics) whilst the external perturbation is unchanged, illustrated by dotted arrows. In the figure each state is as indicated one or several internal states (internal state(S)) as to illustrate the property of topological reconfigurations of the state space.

In literature on artificial (and biological) evolution (AE) many terms are used for breaking down the various parts of a search. In a physical and 'real' system such as the EIM-based ones it is not immediately clear where the boundaries go between these parts, such as *genotype*, *phenotype* and the *genotype-phenotype map*. The definition of these terms is context-dependant, and EIM mixes two contexts; the context of artificial evolution and the context of physical, real life systems. For the purposes of EIM it is sufficient when discussing these terms to simply note that when we discuss the *phenotype* of a system, we are talking about the observed entity interacting with the physical environment, e.g. the electrical current flowing through the material, and observed as state space trajectories, i.e. the observable read out in Figure 4. Since the material is a part of the environment by virtue of being a physical object, there is inevitable interaction between the material and the environment. The genotype can still be treated as is common in AE; the entity operated upon by genetic operators such as mutation and crossover.

The relation between the genotype and phenotype is the genotype-phenotype map, This map takes a genotype as in-
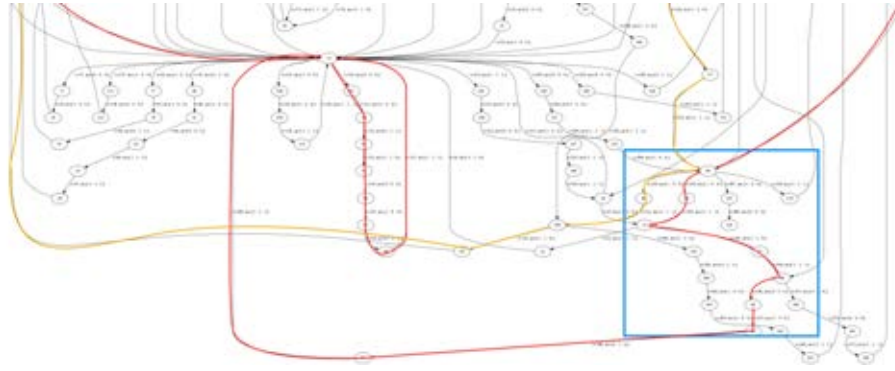
Figure 3: As an illustration of the experimental results a section of the full state space trajectory for an experimental run on salt/water solution with . A cut out in blue highlight state space trajectory changes (presented in Figure 5).

put and transforms it to a real-world entity with real-world physics and in particular includes the interface used to produce the desired manipulative phenomena, such as digital-to-analog converters. It is however worth noting that only what is observable to the EA (phenotypic behaviour or environmental effects of it's existance) can be used as input to a fitness function.

The experimental platform

The experimental results in this paper were achieved using the Mecobo platform (Lykkebo et al., 2014), a hardware/software implementation of an EIM system. The Mecobo platform is shown in Figure 2.

Figure 2(a) show the overall design. Configuration specification, i.e. genotypes, are loaded from a PC to Mecobo over the USB port. The micro controller communicates with the USB interface and with an FPGA on an internal bus. The FPGA can interface directly to materials or as in the figure use a daughter board to extend the signal range as shown in Figure2(b).

Mecobo is capable of controlling close to 100 individual configurable input/output signals (pins) that connects to the material. Each signal are described by parameters at a given point in time., e.g. recording pin from time 0, output frequency pin from time 0 to 10 or output pin voltage level 2.7V from time 0 etc., see (Lykkebo et al., 2014) for a detailed presentation of the Mecobo hardware and software.

The material samples are placed in a material bay, as seen in 2(c), pointed to with a red arrow, and connected to the Mecobo platform.

A standard genetic algorithm with tournament-based selection of size 3, a population size of 30 and a mutation probability of 0.2 was used. The mutation is drawn from a Gaussian distribution with $\sigma = 1$ and $\mu = 0$. The genome consists of 3 floating point numbers, from 0 to 1 which are scaled to integer *square wave* frequencies by 10^6 during the genotype-phenotype mapping process.

Each individual is run 5 times and stability (i.e. repeata-
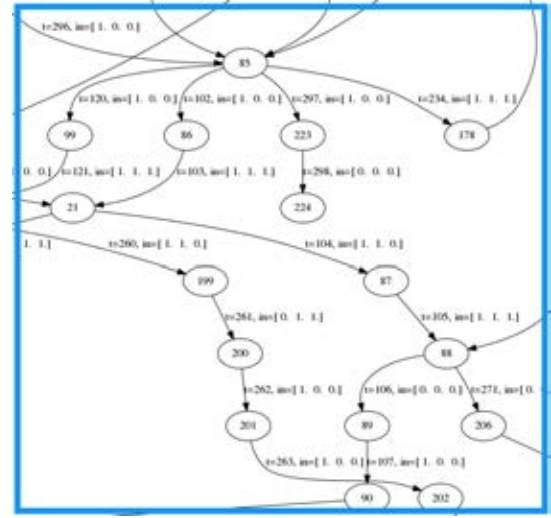
Figure 5: Zoomed region in blue in fig 3, showing the time-dependant behaviour of a driven evolved system in salt.

bility of the measured) states is part of the fitness. One run collects T state vectors $S(t)$ in a vector B_r, $r \in [0, 4]$.

The fitness function counts the number of unique states measured during one time period T, where unique means that for all pairs $(t_1, t_2) \in [0, T]$ $S(t_1) \neq S(t_2)$ and divides the number of unique states by T. Finally, we take the cosine between all pairs of B_r-vectors and multiply the fitness by this number, ensuring that systems who has very similar trajectories in state space get awarded a high fitness.

The material bay has room for 60 connectors, of which we used 50 to connect to the Mecobo platform. 3 pins were then selected as designated input (*to the material*) pins; spread out in the material, one pin was selected as a current sink and the remaining 46 pins were used as material output pins, whose *digital* values were recorded over a time period of 200 uS at 20KHz.

The output pins are connected directly to the FPGA input

246

buffers and further directly connected to FPGA-internal flip-flops with a triggering voltage of 1.7V. We define a *state* as a vector of size n $S(t) = (o_1(t), o_2(t), ..., o_n(t))$ where $o_i(t)$ is the value of flip flop i in the FPGA at time t.

We do not claim that the targeted system does useful computation. The purpose of this example is to demonstrate the existence of potentially complex behaviour in a driven physical system, which gives *potential* for useful computations. The chosen trajectory metric for the fitness evaluation is mainly chosen to be able to investigate for $(DS)^2$ behaviour. However the metric is in accordance with an abstract measurement of complexity as used by Langton (1991), Wolfram (1984) and for developmental systems (Kowaliw, 2008; Nichele et al., 2016).

Material Samples

The 'material cup', in fact a Multichannel Systems micro electrode array (MEA) model (60MEA100/10iR-Ti), pointed to in red in Figure 2(c), holds salt crystals formed by letting a solution 10mL of water with 1mL of kitchen table salt dry out, and before each evolutionary run these salt crystals are mixed with 50μL more of water to allow charge movement in the crystals. A second such MEA was filled with single walled carbon nano tubes (SWCNT) in a PMMA polymer solution. The same experiment is run on both. For control purposes, a fully conductive carbon plate was used.

Results and discussion

The Genetic Algorithm (GA) was used to provide data to be analyzed in a $(DS)^2$ setting. The resulting phenotypes from the evolutionary runs were analyzed by examining the state space traversal. In all results, $(DS)^2$ behavior was found. Figure 3 shows parts of a full state space traversal on dry salt crystals, visualized in graph form. Each node represents one state, and each arc between states one state transition. Figure 5 shows a zoomed version of the box marked by blue in figure 3. Figures 5 shows out-takes of the more interesting dynamic (time-dependent) behaviors in the run. Each edge is marked with the time-step (t) that the transition occurred on, along with the input as a tuple of 3 binary values (a,b,c). For a given input and a state one could expect that a transition should go to the same state, but as we can see for instance in state 85 of figure 5 this is not the case: at t=102, in=(1,0,0) there is a transition to state 86, whereas for t=120 the transition goes to a different state, 99. The red line in 3 traces out the full path of this branching behavior, turning orange at the second pass through state 85. This demonstrates a time-dependent behavior where the traversal of the state space depends on previously seen states.

The same method and set-up was used on the carbon nano tube(CNT) material sample. Time-dependent behavior whilst traversing the state space was present and found in all runs with the carbon nano tube material sample as well.
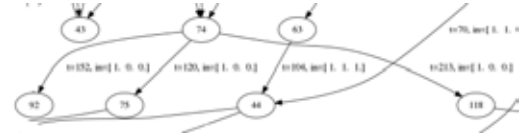
Figure 6: A state graph of a time-evolved carbon nano tubes driven state graph.

Figure 6 shows an out-take with dynamic (time-dependent) behaviors from the full state space traversal graph.

The GA typically achieves a fitness of 0.75 out of a 1.0 max within 50 generations, meaning that it finds a stable behavior that generates roughly $3/4$ unique states relative to the chosen observation level. This result is achieved in all cases tested. The number of transitions that show the time-dependent branching behavior discussed in relation to figure 6 is typically around 5. The chosen genome most often would map to frequencies around 100 KHz., which is higher than the Nyquist-rate relative to our sampling frequency of 50KHz, meaning that we cannot fully reconstruct the input signal from the sampling rate, however this simply underlines our previous points relating to where one sets the observation level– we are not concerned with signal reconstruction, but rather the systems ability to produce behavior in the $(DS)^2$ context.

This behavior is further shown in figure 7 for the salt crystals, and the plot is similar when using CNT as a material. The plot shows periodic behavior interspersed with spurious stable (in the sense that they meet our stability criteria previously defined), states. We stress that the nature of these states can have several reasons, (i.e. metastability in the flip-flops) and we cannot fully rule out the possibility of the sampling apparatus 'interfering' with the dynamics of i.e. the stimulus of the salt crystals, however runs with a fully conductive carbon plate as material gives us no such observed dynamics.

The vertical axis on this figure indicates the time as the voltage is applied to the salt crystals. On the left hand side we see the input to the system as it is captured by the same method that captures the rest of the state data– it is of a more regular nature (though some sampling artifacts).

The complexity of the input compared to the complexity of the output is currently under investigation, however for the purposes of this article the fact that the number of observable states (2^3) is much lower than the number of potential observable output states (2^{46}) demonstrates that there is potential for underlying $(DS)^2$ dynamics. Input is also verified as 'stable' in the same sense as output in that it is also based on a number of repeats of the same run and compared.

Conclusion

Analyzing the behavior of the evolved systems in a $(DS)^2$ setting show that both material samples explored show the

247

property of inducing perturbations to their own dynamics as a function of their system states. In the state space traversal graph this behavior is visualized and show that the underlying dynamics of the materials enables changes in the trajectory in the state space. For the CNT sample the change in trajectory is a product of electrical effects that emerges as observable $(DS)^2$ traversal of the state space. For the salt crystal/water sample topological reconfigurations are possible but not detectable directly. The chosen observation level is here also only based on sampled voltages, a result of currents and charge in the material.

In the experiment *dynamics* was explicitly targeted, in contrast to previous EIM work on similar CNT materials ((Clegg et al., 2014; Mohid et al., 2014)), as the results show the hybrid approach with a digital read out dynamic behavior are present. The presence and possibility to observe such behavior can expand the computational range of a relative simple EIM-set-up from problems requiring only feed forward networks, e.g. (Clegg et al., 2014), to computational tasks requiring memory.

The $(DS)^2$ behavior at our chosen observation level show that it is not given that the system will behave predictably in the sense that a given input and given state will produce the same output– the current state might look the same, but actually be a result of a different underlying dynamic process. This does not imply that it is useless to use this system as a basis for building computational systems or similar, but rather that care must be taken when choosing and applying stimulus to the system and also when we evaluate the output by using a fitness function in a artificial evolution-approach. We suggest allowing the computational system to 'unfold' over time, treating the apparent weakness of unpredictability more as a strength. A way of doing this would be to not consider the system using reductionism and divide it's functionality into smaller parts (i.e. individual gates), but rather consider the system as a whole as a 'basic component' and it's dynamic properties a way of achieving computation.

References

Adamatzky, A. (2009). *Encyclopedia of Complexity and Systems Science*, chapter Reaction-Diffusion Computing, pages 7548–7565. Springer New York, New York, NY.

Adamatzky, A. (2016). *Advances in Physarum Machines*. Springer International Publishing.

Arthur, W. B. (1993). On the Evolution of Complexity. Working Papers 93-11-070, Santa Fe Institute.

Bar-Yam, Y. (1997). *Dynamics of Complex Systems*. Studies in Nonlinearity. Westview Press.

Boses, S. K., Lawrence, C. P., Liu, Z., Makarenko, S., van Damme, R. M. J., Broersma, H. J., and van der Wiel,

Figure 7: Salt crystals state plot

W. G. (2015). Evolution of a designless nanoparticle network into reconfigurable Boolean logic. *Nature Nanotechnology*, 10(12):1048–1052.

Bremermann, H. J. (1962). *Self-Organizing Systems-1962*, chapter Optimization through Evolution and Recombination, pages 93–106. Spartan Books.

Broersma, H., Gomez, F., Miller, J. F., Petty, M., and Tufte, G. (2012). Nascence project: Nanoscale engineering for novel computation using evolution. *International Journal of Unconventional Computing*, 8(4):313–317.

Clegg, K., Miller, J. F., Massey, M. K., and Petty, M. (2014). Travelling Salesman Problem solved in materio by evolved carbon nanotube device. In *Parallel Problem Solving from Nature – PPSN XIII: 13th International Conference*, Lecture Notes in Computer Science, pages 692–701. Springer.

Doursat, R., Sayama, H., and Michel, O. (2013). A review of morphogenetic engineering. *Natural Computing*, 12(4):517–535.

Fromm, J. (2004). *The Emergence of Complexity*. Kassel University Press.

Harding, S. L. and Miller, J. F. (2004). A Tone Discriminator in Liquid Crystal. In *Congress on Evolutionary Computation(CEC2004)*, pages 1800–1807. IEEE.

Holland, J. H. (2012). *Signals and Boundaries: Building Blocks for Complex Adaptive Systems*. MIT Press.

Kowaliw, T. (2008). Measures of complexity for artificial embryogeny. In *GECCO '08: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 843–850. ACM.

Kumar, S. and Bentley, P. J., editors (2003). *On Growth, Form and Computers*. Elsevier Limited Oxford UK.

Langton, C. G. (1991). Computation at the edge of chaos: phase transitions and emergant computation. In Forrest, S., editor, *Emergent Computation*, pages 12–37. MIT Press.

Lykkebo, O. R., Harding, S., Tufte, G., and Miller, J. F. (2014). Mecobo: A hardware and software platform for in materio evolution. pages 267–279.

MacLennan, B. J. (2007). A Review of Analog Computing. Technical Report UT-CS-07-601. Technical report, University of Tennessee, Knoxville.

Massey, M., Volpati, D., Qaiser, F., Kotsialos, A., Pearson, C., Zeze, D., and Petty, M. (2015a). Alignment of liquid crystal/carbon nanotube dispersions for application in unconventional computing. *AIP Conference Proceedings*, 1648(1).

Massey, M. K. (2013). *Electrical Properties of Single-Walled Carbon Nanotube Networks Produced by Langmuir-Blodgett Deposition*. PhD thesis, Durham University, UK.

Massey, M. K., Kotsialos, A., Qaiser, F., Zeze, D. A., Pearson, C., Volpati, D., Bowen, L., and Petty, M. C. (2015b). Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites. *Journal of Applied Physics*, 117(13):134903.

Miller, J. F. and Downing, K. (2002). Evolution in materio: Looking Beyond the Silicon Box. In *2002 NASA/DOD Conference on Evolvable Hardware*, pages 167–176. IEEE Computer Society Press.

Miller, J. F., Harding, S., and Tufte, G. (2014). Evolution-in-materio: evolving computation in materials. *Evolutionary Intelligence*, 7(1):49–67.

Mohid, M., Miller, J. F., Harding, S. L., Tufte, G., Lykkebo, O. R., Massey, M. K., and Petty, M. C. (2014). Evolution-in-materio: Solving function optimization problems using materials. In *2014 14th UK Workshop on Computational Intelligence (UKCI)*, pages 1–8. IEEE.

Nichele, S., Giskeødegard, A., and Tufte, G. (2016). Evolutionary Growth of Genome Representations on Artificial Cellular Organisms with Indirect Encodings. *Artificial Life*, 22(1):76–111.

Nichele, S. and Tufte, G. (2013). Evolution of Incremental Complex Behavior on Cellular Machines. In *ECAL 2013*, pages 63–70. MIT Press.

Omholt, S. W. (2013). From sequence to consequence and back. *Progress in Biophysics and Molecular Biology*, 111(2-3):75–82.

Pask, G. (1959). Physical analougues to the growth of a concept. In *Mechanisation of Thought Processes*, number 10 in National Physical Laboratory Symposium, pages 877–922. Her Majesty's Stationery Office, London, UK.

Sayama, H. (2015). *Introduction to the Modeling and Analysis of Complex Systems*. Open SUNY Textbooks.

Shannon, C. E. (1941). Mathematical Theory of the Differential Analyzer. *Journal of Mathematics and Physics*, 20:337–354.

Simon, H. a. (1962). The architecture of complexity. *American Philosophical Society*, 106(6):467–482.

Spicher, A., Michel, O., and Giavitto, J.-L. (2004). *Cellular Automata: 6th International Conference on Cellular Automata for Research and Industry, ACRI 2004, Amsterdam, The Netherlands, October 25-28, 2004. Proceedings*, chapter A Topological Framework for the Specification and the Simulation of Discrete Dynamical Systems, pages 238–247. Springer Berlin Heidelberg, Berlin, Heidelberg.

Thompson, A., Layzell, P., and Zebulum, R. S. (1999). Explorations in Design Space: Unconventional electronics design through artificial evolution. *IEEE Transactions on Evolutionary Computation*, 3(3):167–196.

Tufte, G. (2009). The Discrete Dynamics of Developmental Systems. In *Proc. of 2009 International Conference on Evolutionary Computation (CEC 2009)*, pages 2209–2216. IEEE.

Turing, A. M. (1937). On Computable Numbers, with an Application to the Entscheidungsproblem. In *Proceedings of the London Mathematical Society 1936-37*, volume 42 of *2*, pages 230–265. London Mathematical Society.

von Neumann, J. (1993). First draft of a report on the edvac. *IEEE Annals of the History of Computing*, 15(4):27–75.

Wolfram, S. (1984). Universality and Complexity in Cellular Automata. *Physica D*, 10(1-2):1–35.

Exploring the coevolution of predator and prey morphology and behavior

Randal S. Olson[1,3], Arend Hintze[2,3], Fred C. Dyer[2,3], Jason H. Moore[1], and Christoph Adami[2,3]

[1]University of Pennsylvania, Philadelphia, PA 19143
[2]Michigan State University, East Lansing, MI 48824
[3]BEACON Center for the Study of Evolution in Action, East Lansing, MI 48824
olsonran@upenn.edu, hintze@msu.edu, fcdyer@msu.edu, jhmoore@upenn.edu, adami@msu.edu

Abstract

A common idiom in biology education states, "Eyes in the front, the animal hunts. Eyes on the side, the animal hides." In this paper, we explore one possible explanation for why predators tend to have forward-facing, high-acuity visual systems. We do so using an agent-based computational model of evolution, where predators and prey interact and adapt their behavior and morphology to one another over successive generations of evolution. In this model, we observe a coevolutionary cycle between prey swarming behavior and the predator's visual system, where the predator and prey continually adapt their visual system and behavior, respectively, over evolutionary time in reaction to one another due to the well-known "predator confusion effect." Furthermore, we provide evidence that the predator visual system is what drives this coevolutionary cycle, and suggest that the cycle could be closed if the predator evolves a hybrid visual system capable of narrow, high-acuity vision for tracking prey as well as broad, coarse vision for prey discovery. Thus, the conflicting demands imposed on a predator's visual system by the predator confusion effect could have led to the evolution of complex eyes in many predators.

Keywords: *swarming behavior*, *predator confusion effect*, *predator-prey coevolution*, *visual acuity*

Introduction

"Eyes in the front, the animal hunts. Eyes on the side, the animal hides." So goes the common idiom in biology education when teaching students how to classify animal skulls. It is widely believed that forward-facing, high-acuity visual systems play an important role in predation, for example, in dragonflies catching flying prey (Olberg, 2012). Despite this common observation, we have little empirical evidence explaining the evolutionary history of these focused visual systems observed in so many predators. In this paper, we explore one hypothesis for why predators tend to evolve complex visual systems: the conflicting demands imposed on a predator's visual system by the well-known "predator confusion effect" could have led to the evolution of complex eyes in many predators.

In previous work, we have shown that not all prey evolve to respond to the presence of predators by hiding or fleeing (Olson et al., 2013a,b; Haley et al., 2014, 2015; Olson et al., 2015, 2016). In fact, some prey species have evolved to stay together, form swarms, and defend themselves as a group for a variety of hypothesized reasons (Krause and Ruxton, 2002). For example, swarming is hypothesized to improve group vigilance (Pulliam, 1973; Treisman, 1975; Kenward, 1978; Treherne and Foster, 1981), reduce the chance of being encountered by predators (Treisman, 1975; Inman and Krebs, 1987), dilute an individual's risk of being attacked (Hamilton, 1971; Foster and Treherne, 1981; Treherne and Foster, 1982), and reduce predator attack efficiency by confusing the predator, i.e., the predator confusion effect (Jeschke and Tollrian, 2007; Ioannou et al., 2008). As such, swarming opens the possibility for an evolutionary "arms race" between predators and their prey (Vermeij, 1987), where the predator and prey continually adapt to one another over many generations of evolution.

Here we use an agent-based computational model of evolution to study the coevolutionary dynamics between predator and prey (Olson, 2015). We implement the predator confusion effect as a simple perceptual constraint on the predator's visual system, and allow both the predator and prey behavior to coevolve over successive generations of evolution (as in Olson et al. 2013a). We further extend this work to allow the predator visual system to simultaneously evolve, which enables us to explore how the predator visual system adapts in response to the prey behavior. From these experiments, we discover a coevolutionary cycle between prey swarming behavior and the predator's visual system. From further analysis, we discover that the predator visual system is the primary driver of this cycle: When the predator evolves a focused visual system, the prey evolve to disperse; whereas when the predator evolves a broad visual system, the prey evolve to swarm. Thus, we suggest that there is a selective advantage for predators that evolve a complex visual system capable of both narrow, high-acuity vision for tracking prey as well as broad, coarse vision for prey discovery.

Methods

To study the coevolutionary cycle between predator and prey, we create an agent-based model in which predator and prey agents interact in a continuous two-dimensional virtual environment. Each agent is controlled by a *Markov Network* (MN), which is a stochastic state machine that makes control decisions based on a combination of sensory inputs (i.e., vision) and internal states (i.e., memory) (Edlund et al., 2011). We coevolve the MNs of predators and prey with a genetic algorithm, selecting for MNs that exhibit behaviors that are more effective at consuming prey and surviving, respectively. Certain properties of the sensory and motor behavior of predators and prey are implemented as constraints that model some of the differences between predators and prey observed in nature (e.g., relative movement speed, turning agility, and, for predators, maximum consumption rate). Predator confusion, described in more detail below, is implemented as a constraint on predator perception that can be varied experimentally. The source code[1] for these experiments is available online. In the remainder of this section, we summarize the evolutionary process that enables the coevolution of predator and prey behavior and visual systems, describe the sensory-motor architecture of individual agents, then present the characteristics of the environment in which predator and prey interact. A detailed description of MNs and how they are evolved can be found in Olson et al. (2016).

Coevolution of predator and prey

We coevolve the predator and prey with a *genetic algorithm* (GA), which is a computational model of evolution by natural selection (Goldberg, 1989). In a GA, pools of genomes are evolved over time by evaluating the fitness of each genome at each generation and preferentially selecting those with higher fitness to populate the next generation. The genomes here are variable-length lists of integers that are translated into MNs during fitness evaluation. Furthermore, we allow the predator visual system to evolve by attaching a single integer value to each predator genome that controls the *predator view angle*, i.e., the size of the arc that the predator's visual system covers (see Figure 1).

The coevolutionary process operates as follows. First, we create separate genome pools for the predator and prey genomes. Next, we evaluate the genomes' fitness by selecting pairs of predator and prey genomes at random without replacement, then place each pair into a simulation environment and evaluate them for 2,000 simulation time steps. Within this simulation environment, we generate 50 identical prey agents from the single prey genome and compete them with the single predator agent to obtain their respective fitness. This evaluation period is akin to the agents' lifespan, hence each agent has a potential lifespan of 2,000 time steps. The fitness values, calculated using the fitness function described below, are used to determine the next generation of

---

[1] Model code: https://github.com/adamilab/eos
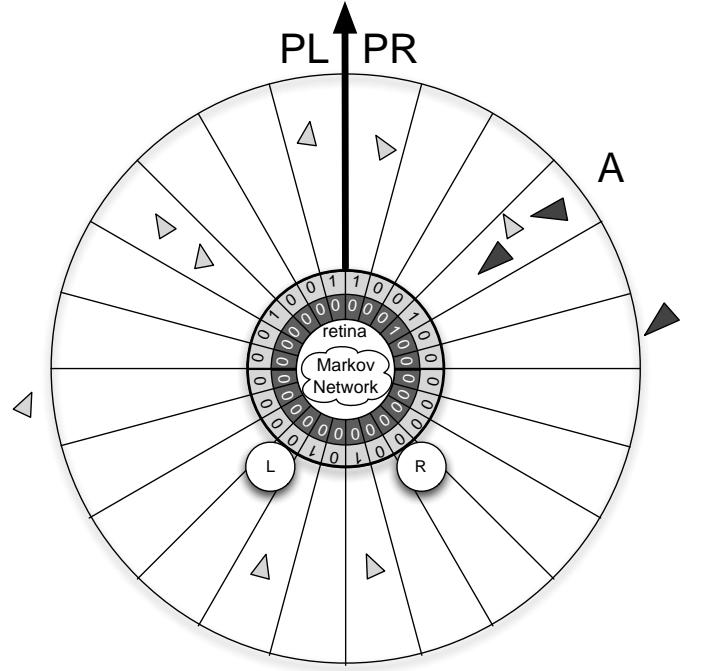
Figure 1: An illustration of the predator and prey agents in the model. Light grey triangles are prey agents and the dark grey triangle is a predator agent. The prey agents have a 180° limited-distance visual system (100 virtual meters) to observe their surroundings and detect the presence of the predator and prey agents, whereas the predator agents have a variable-sized visual system that can see for 200 virtual meters. "PL" and "PR" correspond to the sensors just to the left and right of the agent's heading, respectively. Each agent has its own Markov Network, which decides where to move next based off of a combination of sensory inputs and memory. The left and right actuators (labeled "L" and "R") enable the agents to move forward, left, and right in discrete steps.

the respective genome pools. At the end of the lifetime simulation, we assign the predator and prey genomes separate fitness values according to the fitness functions:

$$W_{\text{predator}} = \sum_{t=1}^{2,000} S - A_t \qquad (1)$$

$$W_{\text{prey}} = \sum_{t=1}^{2,000} A_t \qquad (2)$$

where t is the current simulation time step, S is the starting swarm size (here, $S = 50$), and A_t is the number of prey agents alive at simulation time step t. It can be shown that the predator fitness (Eq. 1) is proportional to the mean kill

251

rate k (mean number of prey consumed per time step), while the prey fitness (Eq. 2) is proportional to $(1 - k)$. Thus, predators are awarded higher fitness for capturing more prey faster, and prey in turn are rewarded for surviving longer. We only simulate a portion of the prey's lifespan where they are under predation because we are investigating swarming as a response to predation, rather than a feeding or mating behavior.

In this case, we use a GA with a population size of 100 (100 prey, 100 predators), per-gene mutation rate of 1%, gene duplication rate of 5%, gene deletion rate of 2%, and mutation rate of 5% for the predator visual system that adds a number in the range [-50°, 50°] to the arc size while keeping it constrained between [1°, 360°].

Once we evaluate all of the predator-prey genome pairs in a generation, we perform fitness-proportionate selection on the populations via a Moran process (Moran, 1962), allow the selected genomes to asexually reproduce into the next generation's populations, apply random mutations to the newborn offspring, increment the generation counter, and repeat the evaluation process on the new populations until the final generation (25,000) is reached.

We perform 30 replicates of each experiment, where for each replicate we seed the prey population with a set of randomly-generated MNs and the predator population with a pre-evolved predator MN that exhibits rudimentary prey-tracking behavior with a 180° visual system. Seeding the predator population in this manner only serves to speed up the coevolutionary process, and has negligible effects on the outcome of the experiment (Figure S1 from Olson et al. 2013a).

Predator and prey agents

Figure 1 depicts the sensory-motor architecture of predator and prey agents in this system. The retina sensors of both predator and prey agents are logically organized into "layers," where a layer includes 12 sensors, with each sensor having a field of view of 15° and a range of 100 virtual meters. Moreover, each layer is attuned to sensing a specific type of agent. Specifically, the predator agents have a single-layer retina that is only capable of sensing prey. In contrast, the prey agents have a dual-layer retina, where one layer is able to sense conspecifics, and the other senses the predator. (We note that there is only a single predator active during each simulation, hence the lack of a predator-sensing retinal layer for the predator agent.)

Regardless of the number of agents present in a single retina slice, the agents only know the agent type(s) that reside within that slice, but not how many, representing the wide, relatively coarse-grain visual systems typical in swarming birds such as Starlings (Martin, 1986). For example in Figure 1, the fourth retina slice to the right (labeled "A") has one prey (light grey triangle) and two predators (dark grey triangles) in it, so both the predator and prey sen-

sors activate and inform the MN that one or more predators *and* one or more prey are currently in that slice. Furthermore, since the prey near the seventh retina slice from the left is just outside the range of the retina slice, the prey sensor for that slice does not activate. We note that although the agent's sensors do not report the number of agents present in a single retina slice, this constraint does not preclude the agent's MN from evolving and making use of a counting mechanism which reports the number of agents present in a set of retina slices. Once provided with its sensory information, the prey agent chooses one of four discrete actions: (1) stay still; (2) move forward 1 unit; (3) turn left 8° while moving forward 1 unit; or (4) turn right 8° while moving forward 1 unit.

Likewise, the predator agent detects nearby prey agents using a limited-distance (200 virtual meters), segmented retina covering an evolvable angle in front of the predator that functions just like the prey agent's retina. Similar to the prey agents, predator agents make decisions about where to move next, but the predator agents move 3x faster than the prey agents and turn correspondingly slower (6° per simulation time step) due to their higher speed.

Simulation environment

We use a simulation environment to evaluate the relative performance of the predator and prey agents. At the beginning of every simulation, we place a single predator agent and 50 prey agents at random locations inside a closed 512×512 unit two-dimensional simulation environment. Each of the 50 prey agents are controlled by clonal MNs of the particular prey MN being evaluated. We evaluate the swarm with clonal MNs to eliminate any possible effects of selection at the individual level, e.g., the "selfish herd" effect (Wood and Ackland, 2007; Olson et al., 2013b).

During each simulation time step, we provide all agents their sensory input, update their MN, then allow the MN to make a decision about where to move next. When the predator agent moves within 5 virtual meters of a prey agent it can see (i.e., the prey agent is anywhere within the predator's visual field), it automatically makes an attack attempt on that prey agent. If the attack attempt is successful, the target prey agent is removed from the simulation and marked as consumed. Predator agents are limited to one attack attempt every 10 simulation time steps, which is called the *handling time*. The handling time represents the time it takes to consume and digest a prey after successful prey capture, or the time it takes to refocus on another prey in the case of an unsuccessful attack attempt. Shorter handling times have negligible effects on the outcome of the experiment, except for when there is no handling time at all (Figure S2 from Olson et al. (2013a)).

To investigate predator confusion as an indirect selection pressure driving the coevolution of swarming, we implement a perceptual constraint on the predator agent. When
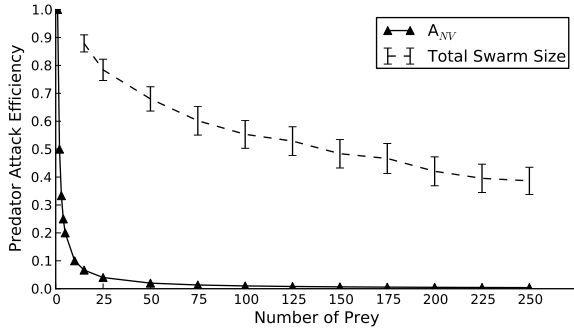
Figure 2: Relation of predator attack efficiency (# successful attacks / total # attacks) to number of prey. The solid line with triangles indicates predator attack efficiency as a function of the number of prey within the visual field of the predator (A_{NV}). Similarly, the dashed line with error bars shows the actual predator attack efficiency given the predator attacks a group of swarming prey of a given size, using the A_{NV} curve to determine the per-attack predator attack success rate. Error bars indicate two standard errors over 100 replicate experiments.

the predator confusion mechanism is active, the predator agent's chance of successfully capturing its target prey agent (P_{capture}) is diminished when any prey agents near the target prey agent are visible anywhere in the predator's visual field. This perceptual constraint is similar to previous models of predator confusion based on observations from natural predator-prey systems (M. Jeschke and Tollrian, 2005; Jeschke and Tollrian, 2007; Ioannou et al., 2008), where the predator's *attack efficiency* (# successful attacks / total # attacks) is reduced when attacking swarms of higher density.

P_{capture} is determined by the equation $P_{\mathrm{capture}} = \frac{1}{A_{\mathrm{NV}}}$, where A_{NV} is the number of prey agents that are visible to the predator, i.e., anywhere in the predator agent's visual field, *and* within 30 virtual meters of the target prey. By only counting prey near the target prey, this mechanism localizes the predator confusion effect to the predator's retina, and enables us to experimentally control the strength of the predator confusion effect.

Although our predator confusion model is based on the predator's retina, it is functionally equivalent to previous models that are based on the total swarm size (Figure 2, dashed line), see, e.g., (M. Jeschke and Tollrian, 2005; Tosh et al., 2006; Jeschke and Tollrian, 2007; Ioannou et al., 2008). As shown in Figure 2 (solid line with triangles), the predator has a 50% chance of capturing a prey with one visible prey near the target prey ($A_{\mathrm{NV}} = 2$), a 33% chance of capturing a prey with two visible prey near the target prey ($A_{\mathrm{NV}} = 3$), etc. As a consequence, prey are in principle able to exploit the combined effects of predator confusion and handling time by swarming.

Results

To evaluate the evolved prey behavior quantitatively, we obtain the line of descent (LOD) for every replicate experiment by tracing the ancestors of the most-fit prey MNs in the final population until we reach the randomly-generated ancestral MN with which the starting population was seeded (see Lenski et al. 2003 for an introduction to the concept of a LOD in the context of computational evolution). For each ancestor in the LOD, we characterized the behavior with a common behavior measurement called *swarm density* (Huepe and Aldana, 2008). We measured the swarm density as the mean number of prey within 30 virtual meters of each other over a lifespan of 2,000 simulation time steps, which provides an indication of how closely the prey are staying near each other on average. Similarly, we evaluate the predator's view angle by tracing the LOD of the most-fit predator and observing the view angle of each ancestor.

In Olson et al. 2013a, we showed that when the predator's visual system only covered the frontal 60° or less, swarming to confuse the predator was no longer a viable adaptation (as indicated by a mean swarm density of 0.68 ± 0.02 at generation 1,200). In this case, the predator had such a narrow view angle that few swarming prey were visible during an attack, which minimizes the confusion effect and correspondingly increases the predator's capture rate (Figure S8 from Olson et al. 2013a). When the predator's visual system was incrementally modified to cover the frontal 120° and beyond, swarming again became an effective adaptation against the predator due to the confusion effect (indicated by a mean swarm density of 6.13 ± 0.76 at generation 1,200). This suggests that the predator confusion mechanism may not only provide a selective pressure for the prey to swarm, but it could also provide a selective pressure for the predator to narrow its view angle to become less easily confused.

When we allow the predator view angle to coevolve along with the predator and prey behavior, we observe that the predator populations do indeed evolve focused visual systems in response to prey swarming behavior (Figure 3), as indicated by the predator view angle evolving to $< 100°$ once the prey begin to swarm. Interestingly, the predator and prey populations appear to repeatedly cycle between different states of view angles and behaviors, respectively, such that there is a significant negative correlation between the predator view angle and swarm density across all 30 coevolution experiments (Figure 4). This finding is surprising because the predator population should be able to effectively "defeat" the swarming prey population by shrinking their visual system to the point that the prey will no longer evolve to swarm. Why then would the predator population evolve to widen their visual system once the prey evolve to disperse, and allow the prey population to again evolve swarming behavior to reduce the predators' attack efficiency?

Shown in Figure 5, when predators with fixed view angles are competed against dispersive prey, the predators with

253
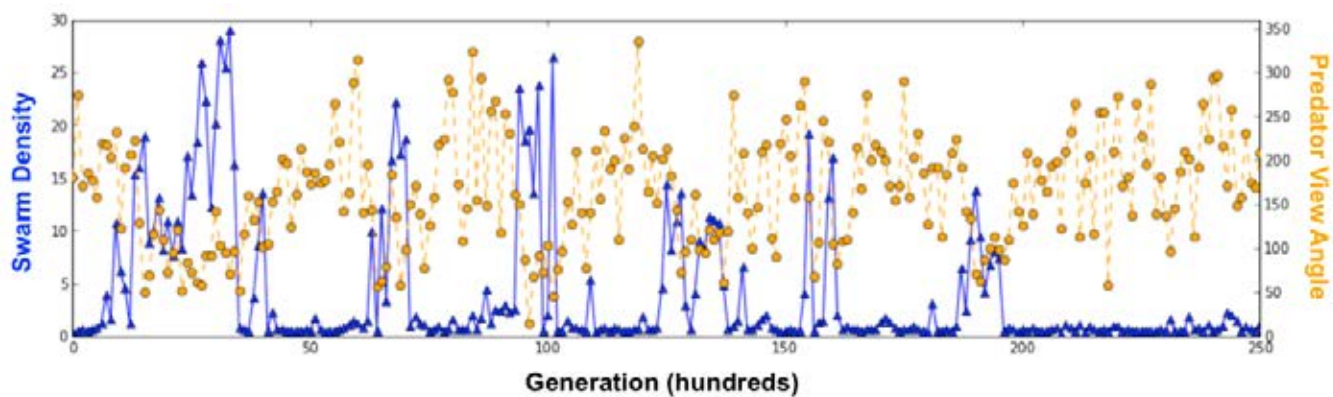
Figure 3: Swarm density and predator view angle from the LOD of a single coevolution experiment. The predator and prey populations repeatedly cycle between different states of view angles and behaviors.
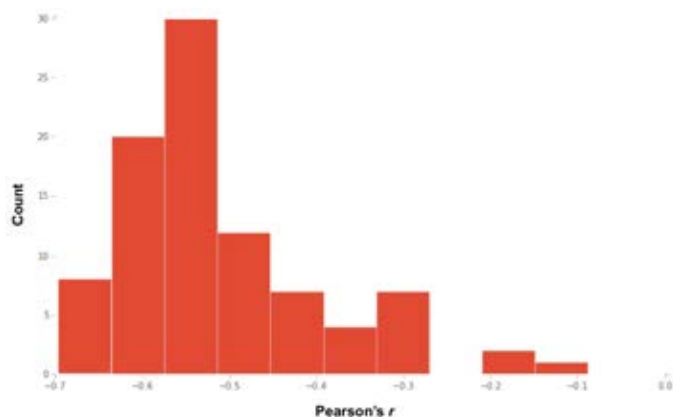
Figure 4: Pearson's *r* between swarm density and predator view angle from the LODs of 30 coevolution experiments. All coevolution experiments have a negative correlation between swarm density and predator view angle, indicating that when swarm density goes up, predator view angle goes down, and vice versa. $P <= 0.001$ for all correlations.
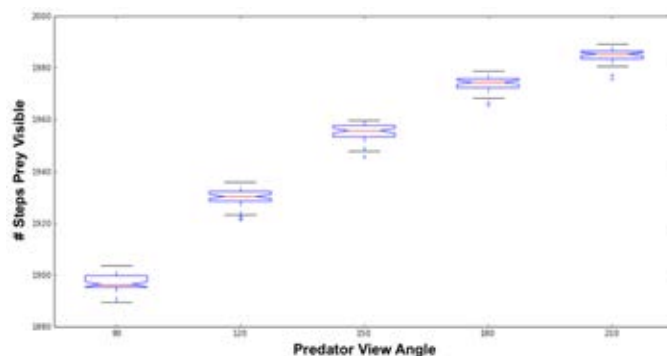
Figure 5: Number of simulation time steps that prey are present anywhere in an evolved predator's visual system depending on the predator's view angle. Each box plot represents 30 replicates, and the notches represent the 95% confidence interval around the median. Here, the predator is competed against dispersive prey. Predators with higher view angles are more likely to have prey anywhere in their visual system at a given time. $P <= 0.001$ between all view angles with a Kruskal-Wallis multiple comparison.
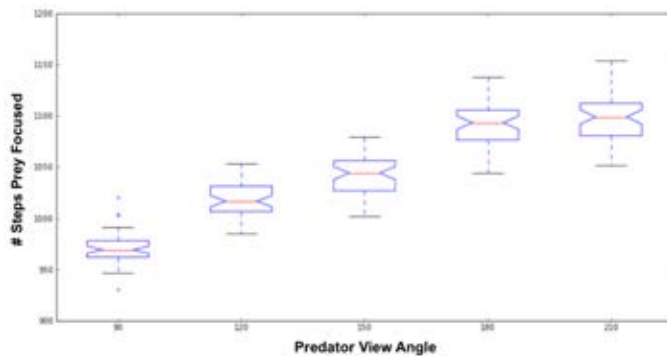
Figure 6: Number of simulation time steps that prey are visible in a portion of an evolved predator's visual system that it pays attention to, depending on the predator's view angle. Each box plot represents 30 replicates, and the notches represent the 95% confidence interval around the median. Here, the predator is competed against dispersive prey. Predators with higher view angles are more likely to spot prey at a given time, which increases their foraging efficiency. $P <= 0.001$ between all view angles except 180 vs. 210 with a Kruskal-Wallis multiple comparison.
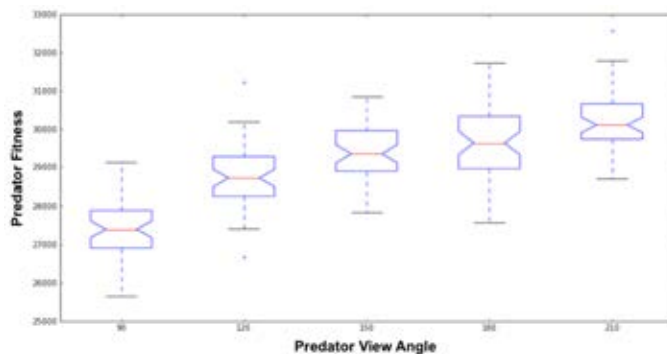
Figure 7: Fitness of an evolved predator when competed against dispersive prey, depending on the predator's view angle. Each box plot represents 30 replicates, and the notches represent the 95% confidence interval around the median. Predators with higher view angles forage for prey more efficiently, thus capturing more prey in their lifetime and improving their fitness. $P <= 0.001$ between all view angles except 150 vs. 180 and 180 vs. 210, Kruskal-Wallis multiple comparison.
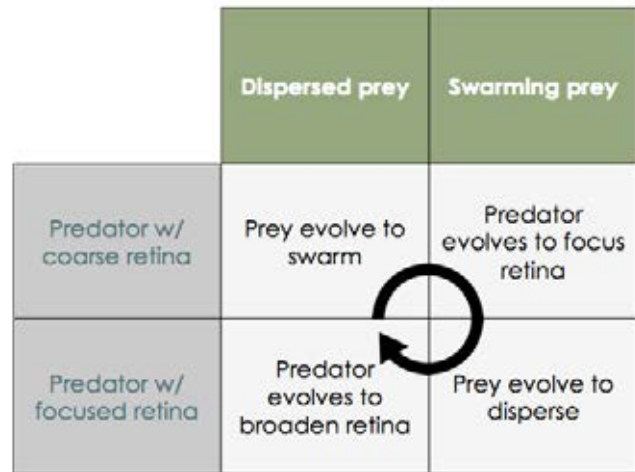
Figure 8: Diagram depicting the observed coevolutionary cycle between the predator and prey in the presence of the predator confusion effect.

broader visual systems are more likely to find a prey anywhere in their visual system at any time. Further, Figure 6 demonstrates that predators with broader visual systems are also more likely to find dispersive prey in a portion of their visual system that they pay attention to, which means they spend less time searching for prey. Thus, the increased foraging efficiency that broader visual systems offer predators against dispersive prey results in higher predator fitness (Figure 7), which explains why predators evolve higher view angles in the presence of dispersive prey.

Discussion

As demonstrated in Figure 3, selection favors predators with a more focused visual system once swarming has evolved in prey. However, once the predators evolve a focused visual system, the prey evolve dispersive behavior in response and a coevolutionary cycle commences between the predator visual system and prey behavior. Some time ago, researchers commonly assumed that the evolution of social behavior is a one-way street, and that once social integration has arisen in a population it must be so advantageous (compared to the cost of living in close proximity to conspecifics) that it would not be lost (Wcislo and Danforth, 1997). Our findings demonstrate that at least one form of social behavior—the tendency to form cohesive swarms—can readily be gained and lost, and that the gain and loss is governed by a coevolutionary cycle that could occur between natural predators and prey due to the predator confusion effect, as depicted in Figure 8.

Furthermore, the findings in this paper highlight a trade-off that natural predators likely experience when hunting for prey: Broader, less-focused visual systems are more useful

for initially spotting prey, but focused visual systems are better adapted for tracking an individual prey down and avoiding the effects of predator confusion when hunting prey in groups. Thus, these conflicting demands imposed on the predator's visual system by the predator confusion effect could select for the evolution of complex eyes that satisfy both needs. Indeed, many animals—both vertebrate and invertebrates—do have such complexity in the arrangement of their retinae, including the presence of a fovea in vertebrates (Moore et al., 2012) or "acute zones" in invertebrates (Land, 1997). Our system could not have evolved such complexity because the retinal slices could not vary independently.

In future work, we plan to implement a more advanced predator visual system that will allow the number of retina slices to vary, and allow each individual slice to vary in size. Through such a visual system, evolution will be capable of adjusting each retina slice as needed and allow us to explore under what conditions complex eyes will evolve. Another interesting approach would be to allow the prey visual system to coevolve as well in order to explore how evolution shapes prey visual systems in response to predation.

Conclusions

In this paper, we implemented a computational model of evolution that allowed us to explore the coevolution of predator and prey morphology and behavior. In particular, we explored the coevolution of the predator's visual system and prey behavior and discovered that a repeated coevolutionary cycle occurs when we introduce the predator confusion effect. Furthermore, we provided evidence that the predator visual system is what drives this coevolutionary cycle, and suggested that the cycle could be closed if the predator evolves a hybrid visual system capable of narrow, high-acuity vision for tracking prey as well as broad, coarse vision for prey discovery. Thus, the conflicting demands imposed on a predator's visual system by the predator confusion effect could have led to the evolution of complex eyes in many predators.

Acknowledgements

References

Edlund, J., Chaumont, N., Hintze, A., Koch, C., Tononi, G., and Adami, C. (2011). Integrated information increases with fitness in the evolution of animats. *PLoS Comput. Biol.*, 7:e1002236.

Foster, W. and Treherne, J. (1981). Evidence for the dilution effect in the selfish herd from fish predation on a marine insect. *Nature*, 293:466–467.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Boston, MA, 1st edition.

Haley, P. B., Olson, R. S., Dyer, F. C., and Adami, C. (2014). Exploring conditions that select for the evolution of cooperative group foraging. In *Proceedings of the International Conference on the Simulation and Synthesis of Living Systems (ALIFE 2014)*, pages 310–311, Cambridge, MA, USA. MIT Press.

Haley, P. B., Olson, R. S., Dyer, F. C., and Adami, C. (2015). Evolving an optimal group size in groups of prey under predation. In *Proceedings of the Thirteenth European Conference on the Synthesis and Simulation of Living Systems (ECAL 2015)*, page 620. MIT Press.

Hamilton, W. (1971). Geometry for the selfish herd. *J. Theor. Bio.*, 31:295–311.

Huepe, C. and Aldana, M. (2008). New tools for characterizing swarming systems: A comparison of minimal models. *Physica A*, 387:2809–2822.

Inman, A. and Krebs, J. (1987). Predation and group living. *Trends Ecol. Evol.*, 2:31–32.

Ioannou, C., Tosh, C., Neville, L., and Krause, J. (2008). The confusion effect–from neural networks to reduced predation risk. *Behav. Ecol.*, 19:126–130.

Jeschke, J. and Tollrian, R. (2007). Prey swarming: Which predators become confused and why? *Anim. Behav.*, 74:387–393.

Kenward, R. (1978). Hawks and doves: Factors affecting success and selection in Goshawk attacks on Woodpigeons. *J. Anim. Ecol.*, 47:449–460.

Krause, J. and Ruxton, G. (2002). *Living in groups*. Oxford University Press, USA.

Land, M. F. (1997). Visual acuity in insects. *Annual Review of Entomology*, 42(1):147–177.

Lenski, R., Ofria, C., Pennock, R., and Adami, C. (2003). The evolutionary origin of complex features. *Nature*, 423:139–44.

M. Jeschke, J. and Tollrian, R. (2005). Effects of predator confusion on functional responses. *Oikos*, 111:547–555.

Martin, G. (1986). The eye of a passeriform bird, the European starling (Sturnus vulgaris): Eye movement amplitude, visual fields and schematic optics. *J. Comp. Physiol. A*, 159:545–557.

Moore, B. A., Kamilar, J. M., Collin, S. P., Bininda-Emonds, O. R., Dominy, N. J., Hall, M. I., Heesy, C. P., Johnsen, S., Lisney, T. J., Loew, E. R., et al. (2012). A novel method for comparative analysis of retinal specialization traits from topographic maps. *Journal of Vision*, 12(12):13–13.

Moran, P. (1962). *The statistical processes of evolutionary theory*. Clarendon Press.

Olberg, R. M. (2012). Visual control of prey-capture flight in drag-onflies. *Current Opinion in Neurobiology*, 22(2):267–271. Neuroethology.

Olson, R., Hintze, A., Dyer, F., Knoester, D., and Adami, C. (2013a). Predator confusion is sufficient to evolve swarming behaviour. *J. Roy. Soc. Interface*, 10:20130305.

Olson, R. S. (2015). *Elucidating the evolutionary origins of collective animal behavior*. PhD thesis, Michigan State University.

Olson, R. S., Haley, P. B., Dyer, F. C., and Adami, C. (2015). Exploring the evolution of a trade-off between vigilance and foraging in group-living organisms. *Royal Society Open Science*, 2(9).

Olson, R. S., Knoester, D. B., and Adami, C. (2013b). Critical interplay between density-dependent predation and evolution of the selfish herd. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2013)*, pages 247–254.

Olson, R. S., Knoester, D. B., and Adami, C. (2016). Evolution of swarming behavior is shaped by how predators attack. *Artif. Life*, 22:to appear. http://arxiv.org/abs/1310.6012.

Pulliam, H. R. (1973). On the advantages of flocking. *Journal of Theoretical Biology*, 38:419–422.

Tosh, C. R., Jackson, A. L., and Ruxton, G. D. (2006). The confusion effect in predatory neural networks. *Am. Nat.*, 167:E52–E65.

Treherne, J. and Foster, W. (1981). Group transmission of predator avoidance behaviour in a marine insect: The Trafalgar effect. *Anim. Behav.*, 29:911–917.

Treherne, J. and Foster, W. (1982). Group size and anti-predator strategies in a marine insect. *Anim. Behav.*, 30:536–542.

Treisman, M. (1975). Predation and the evolution of gregariousness. I. Models for concealment and evasion. *Anim. Behav.*, 23:779–800.

Vermeij, G. (1987). *Evolution and escalation: An ecological history of life*. Princeton University Press.

Wcislo, W. T. and Danforth, B. N. (1997). Secondarily solitary: the evolutionary loss of social behavior. *Trends Ecol. Evol.*, 12:468–474.

Wood, A. and Ackland, G. (2007). Evolving the selfish herd: Emergence of distinct aggregating strategies in an individual-based model. *P. Roy. Soc. B*, 274:1637–42.

Evolvability

The Evolution of Evolvability in Evolutionary Robotics

David Shorten[1] and Geoff Nitschke[1]

[1]Department of Computer Science, University of Cape Town
dshorten@cs.uct.ac.za, gnitschke@cs.uct.ac.za

Abstract

Previous research has demonstrated that computational models of *Gene Regulatory Networks* (GRNs) can adapt so as to increase their *evolvability*, where evolvability is defined as a population's *responsiveness* to environmental change. In such previous work, phenotypes have been represented as bit strings formed by concatenating the activations of the GRN after simulation. This research is an extension where previous results supporting the evolvability of GRNs are replicated, however, the phenotype space is enriched with time and space dynamics with an *evolutionary robotics* task environment. It was found that a GRN encoding used in the evolution of a way-point navigation behavior in a fluctuating environment results in (robot controller) populations becoming significantly more responsive (evolvable) over time. This is as compared to a direct encoding of controllers which was unable to improve it's evolvability in the same task environment.

Introduction

An open question in artificial and natural life is whether digital and natural organisms undergoing an evolutionary process are able to become more *responsive* to changes in their environment, that is to become more *evolvable* (Wagner and Altenberg, 1996a). A prevailing hypothesis is that if the environment sufficiently varies over time, then organisms evolve the ability to be able to evolve suitable adaptations to such environmental changes faster (Wagner and Altenberg, 1996a; Draghi and Wagner, 2008). Crombach and Hogeweg (2008) as well as Draghi and Wagner (2008) have demonstrated that computational models of *Gene Regulatory Networks* (GRNs) exhibit such *evolvability*. This study's main goal is to replicate the results of this previous research (Crombach and Hogeweg, 2008; Draghi and Wagner, 2008), but in the context of *evolutionary robotics* (Nolfi and Floreano, 2000) experiments that test robot controller (behavior) evolution in environments where the goal tasks vary over time.

The representation problem in *Evolutionary Computation* (EC) (Eiben and Smith, 2003) addresses the issue of how to represent and adapt (mutate and recombine) genotypes such that a broad range of complex solutions are represented by relatively simple genotype encodings (Wagner and Altenberg, 1996b). Representation choice and associated operators has a significant impact on the evolution of viable solutions and representations which facilitate evolution have been termed *evolvable* (Wagner and Altenberg, 1996b; Rothlauf, 2006a; Simões et al., 2014). Similarly, in nature, genetic information defining the form and function of an organism is stored within its genotype, however the developmental process which translates this information into phenotypes is not well understood (Pigliucci, 2010). It has become clear that the mapping between genotype and phenotype is neither one-to-one nor linear (Gjuvsland et al., 2013). In many organisms, including the case of *Ribonucleic acid* (RNA) folding (Draper, 1992), it has been found that many genotypes can code for a single phenotype and that genetic change resulting from mutation is not proportional to phenotypic change (Pigliucci, 2010; Parter et al., 2008).

In EC, this is known as a *developmental* or *generative* (indirect encoding) genotype representation (Stanley and Miikkulainen, 2003), where effects of mutations are not only determined by representation and associated mutation operators, but also by the population's position in genotype space. This is distinct from a one-to-one mapping (direct encoding) where, for a given phenotype and its associated genotype, mutational effects are determined by the representation, mutation operators and fitness function. Thus the population's location (genotype values) in the genotype space can be viewed as an integral component of representation (Rothlauf, 2006b).

An open question in biology is whether developmental representations have occurred by chance, or if such representations have also been subject to evolution (Parter et al., 2008). A current hypothesis is that an organism's genotype representation is itself evolvable due to the evolution of *evolvability* (Wagner and Altenberg, 1996b), (Pigliucci, 2008). However, this is complicated by multiple definitions of *evolvability*[1] in both evolutionary biology(Pigliucci,

---

[1]For a review of evolvability in biology, the reader is referred to Pigliucci (Pigliucci, 2008).

2008), (Pigliucci, 2010; Parter et al., 2008) and EC (Tarapore and Mouret, 2014), for example, evolvability can refer to either populations or individuals (Wilder and Stanley, 2015). Similarly, within EC, numerous definitions and associated metrics have been proposed. For example, those that focus exclusively on solution fitness (Grefenstette, 1999) or variability of offspring (Lehman and Stanley, 2013). Tarapore and Mouret (Tarapore and Mouret, 2014) developed a metric which incorporated both the fitness and diversity of offspring.

The principle aim of this work is to extend the demonstration of evolvability in GRN's by Crombach and Hogeweg (2008) as well as Draghi and Wagner (2008) to an evolutionary robotics domain. In these previous studies a population's evolvability was defined as its *responsiveness*, that is, the population's ability to rapidly adapt to changes in the fitness landscape. This work maintains consistency with this definition. Hence, we define evolvability to be tantamount to a population's *adaptability* (Kirschner and Gerhart, 1998). This implies that we do not predefine sets of features that will likely propagate beneficial phenotypes (behaviors) in the evolutionary process. Rather, in line with biological literature (Pigliucci, 2008; Flatt, 2005), evolvability is an organism's (genotype's) capability to adapt and survive in its environment.

We investigate evolvability in the context of an evolutionary robotics task, where robot controller (behavior) evolution is tested using both indirect (GRN) and direct genotype encodings. The hypothesis is that an indirect encoding facilitates the improvement of responsiveness over the course of evolution in a robotics task domain with changing task environments, whereas indirect encoding does not. Here we use *responsiveness* and *evolvability* interchangeably to mean the speed with which a population adapts given task environment changes. The robotics task was way-point navigation, where responsiveness was tested via having the environment fluctuate with its own task variants. In order to facilitate this, two different way-point layouts were used.

Results indicated that evolution using the indirect (GRN) encoding facilitated the evolution of controllers that were significantly more responsive (adapted) to task environment fluctuations over evolutionary time. Comparatively, evolution using the direct (bit-string) encoding applied to this task indicated that evolved behaviors were unresponsive and unable to appropriately adapt to task environment variations over time.

Methods

Simulation and Task Environment

The evolutionary robotics simulation used a bounded two-dimensional continuous environment[2], where the environment was imposed with a 400 x 200 grid for ease of specifying task environment and simulation parameters (table 1).

[2]An extension of the *RoboRobo* simulator (Bredeche et al., 2013) was used for all experiments.

The task tested was *way-point navigation*. During controller evolution, the task variants were switched in order to simulate a fluctuating environment (Evolutionary Algorithm Section). Two task variants were specified, each requiring the robot to pass by (within a given distance, table 1) of a pre-specified number of way-points. The task required the robot to pass the way-points in a specific order and within its lifetime (a given number of simulation time steps, table 1). The number of way-points the robot passed by during its lifetime equalled its fitness.

Figure 1 shows the layout of the way-points for each of the two task variants, where the way-points were specially positioned to encourage the emergence of a wall-following behavior.

Robot Controller

The robot controller was a fully connected feed-forward *Artificial Neural Network* (ANN) with twelve connection weights. That is, three hidden layer neurons (Sigmoidal units), connected to two sensory input and two motor output neurons. The two sensory inputs were distance sensors, each placed $\pi/3$ radians on either side of the direction in which the robot was facing. These sensors operated similar to Infrared proximity sensors. ray casting in the sensor's field of view. If this line intersected a wall in the sensor's range, then the sensor's reading was d/r, where d was the distance to the wall and r was the sensor's range. If there was no wall in sensor range, then the sensor reading was 1.0. The controller's motor outputs determined the robot's speed and heading, where outputs were normalized in the range $[0.0, 1.0]$ and corresponded to minimum and maximum speed and heading values (table 1).

Gene Regulatory Network

The *Gene Regulatory Network* (GRN) model for robot controller encoding is based on that used in previous related work (Crombach and Hogeweg, 2008; Draghi and Wagner, 2008). Nodes in the GRN are genes, and connections between the nodes are either *excitatory* or *inhibitory*. All nodes are updated synchronously via equation 1.

$$s_i(t+1) = \begin{cases} 1: & \sum_j w_{ij} s_j(t) > \theta_i \\ 0: & \sum_j w_{ij} s_j(t) \le \theta_i \end{cases} \quad (1)$$

Where, $s_i(t)$ is the activation of the ith node at simulation iteration t, w_{ij} is the connection weight of the directed edge from the ith to the jth node, and θ_i is the threshold of the ith node. If no such connection exists then $w_{ij} = 0$. Table 2 presents the GRN parameters. In order to facilitate the conversion of activations into bit-strings all nodes were given a unique value in the range $[0, l]$, where l is one less than the number of nodes (Gene Regulatory Network Encoding section).
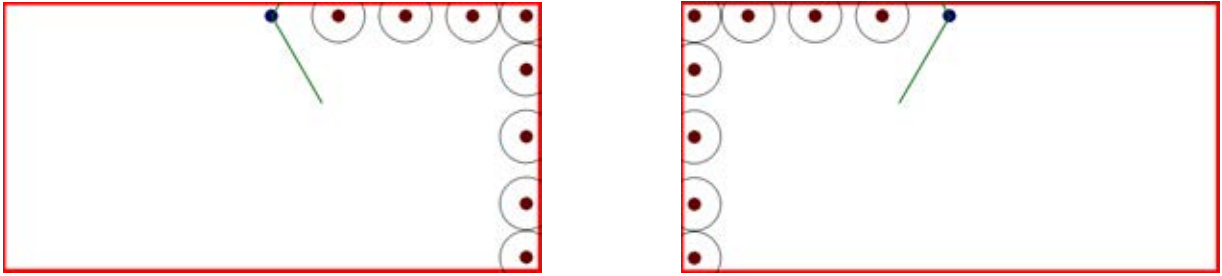
Figure 1: Visualization of the *way-point navigation task* for each of the two task variants (left and right). Way-points (brown dots) line the top-right and top-left corners of each task variant. The circles surrounding each way-point represent the distance within which the robot must pass the way-point. The robot is presented as the blue dot on either side (far left and far right) with two lines extending (representing sensor fields of view).

Mutation Operators. Table 3 specifies the mutation operators used in the *Evolutionary Algorithm* (EA) applied to evolve the GRN. The `mutate_weight`, `add_edge`, and `delete_edge` operators were applied to the GRN node connections, where for every connection, the operator was applied with a given probability (table 4). All other mutation operators acted on nodes activation and threshold functions with a given probability (table 4).

Binary (Direct) Encoding. A direct mapping function was used to map the binary genotype encoding to an ANN controller. To convert the sixty element binary string genotype to the twelve connection weight values which specify an ANN controller, the genotype string was split into twelve smaller strings of five elements each. These twelve strings were then converted into real numbers in the range $[0.0, 1.0]$ using equation 2.

$$2\left(\frac{\sum_{i\in\{0...4\}} a_i 2^i}{2^5 - 1} - 0.5\right) \tag{2}$$

Where, a_i is the ith element of substring a.

Gene Regulatory Network (Indirect) Encoding. Using methods from previous work (Crombach and Hogeweg, 2008; Draghi and Wagner, 2008), the GRN was simulated for a given number of iterations (table 2). During this simulation time, convergence to a point attractor was tested for by determining whether node activations in the last and penultimate iterations were identical. If the GRN did not settle on a point attractor then it was marked for removal and no further evaluation took place. Preliminary testing indicated that the removal of these GRNs had a negligible effect on the evolutionary dynamics in this study's experiments. If the GRN settled on a point attractor then the node activations were converted into a bit string, where the ordering of the activations was determined by unique node identifiers (Gene Regulatory Network section). This GRN convergence test was done to maintain consistency with previous work (Crombach and Hogeweg, 2008), (Draghi and Wagner, 2008). Bit-

strings were then converted into an ANN controller using the genotype to ANN direct encoding mapping method (Binary Encoding section, equation 2).

Evolutionary Algorithm (EA)

The binary and GRN encoded genotypes were evolved with an EA using deterministic tournament selection (Eiben and Smith, 2003), applied 200 times per generation. Also, the EA used mutation only (there was no recombination operator). Table 4 presents the EA parameters. The following subsections detail the EA setup for controller evolution using the binary and GRN encodings, respectively.

Direct Encoding When the EA was started, a population of bit-string genotypes were randomly generated. Each generation, each genotype was systematically selected, decoded into an ANN controller (Binary Encoding section) and tested in the way-point navigation task (Simulation and Task Environment Section) for one *robot lifetime* (table 1), after which fitness was assigned to the tested controller (genotype). One generation was when all genotypes had been tested and evaluated. Selection and mutation operators were then applied (table 4).

In preliminary mutation operator testing it was found that using a constant mutation rate for each bit (gene) in each genotype resulted in a significantly lower task performance compared to controller evolution with GRN encoding. That is, at a high mutation rate, genotypes with high fitness were quickly found, however, convergence was sub-optimal. For relatively low mutation rates, the population converged to a set of fit genotypes, however genotypes with optimal fitness were not found. To address this, we executed 100 evolutionary runs of controller evolution with GRN encoding and recorded the Hamming distance between parent and child genotypes (Gene Regulatory Network Encoding section). The probability of each Hamming distance (number of bit-flips) occurring was then calculated and these probabilities were used to determine the number of bit-flips in a mutation of the binary direct encoding. That is, if on average, the as-

Parameter	Value
Robot speed	5 units per iteration
Robot maximum angular velocity	0.5 radians per iteration
Robot heading	$[0, 2\pi)$
Sensor range	75 units
Collision radius	20 units
Environment size	400 x 200 units
Way-point radius (navigation task)	20 units
Number of way-points	$[0, 59]$
Simulation iterations (robot lifetime)	250

Table 1: Simulation and task parameter values.

Parameter	Value Range
GRN Weights (w_{ij})	$[-2.0, 2.0]$
ANN Weights	$[-1.0, 1.0]$
ANN sensory and outputs	$[-1.0, 1.0]$
Thresholds (θ_{ij})	$[-3, 3]$
Number of nodes	60
Incoming / outgoing connections per node	$[0, 59]$
Simulation iterations (maximum t)	20

Table 2: Parameters for the *Binary* and *Gene Regulatory Network* controller encoding.

Operator	Description
`mutate_weight`	A new value for the weight of an edge is chosen from the allowed values.
`mutate_activation`	The value of the initial activation of a node is flipped.
`mutate_threshold`	A new value for the threshold of a node is chosen from the allowed values.
`add_edge`	A new incoming edge is added to the node. It connects to a random node and has a random weight.
`delete_edge`	One of the node's edges is chosen at random and removed.

Table 3: Mutation operators for the Gene Regulatory Network.

sociated bit strings of GRNs had a Hamming distance of h from their parent genotypes (with probability p), then when a binary encoded genotype was mutated, h bits would be flipped with probability p. This probability distribution was then assigned as the mutation rate for the direct encoding approach. Table 5 presents the probability of a given number of bit-flips occurring whenever the binary encoding mutation operator was applied. It was found, when these probabilities were used (table 5), that the task performance of controller evolution with direct binary encoding was comparable to the task performance to early generations of controller evolution using the GRN encoding (Results section).

Gene Regulatory Networking Encoding When the EA was started, GRNs were randomly initialized with given parameter constraints, and each GRN simulated for a given number of iterations (table 2). If the GRN settled on a point attractor, then the GRNs activations were mapped to a bit-string, which was then decoded into an ANN controller (Gene Regulatory Network Encoding section). Each decoded ANN controller was then simulated in the way-point navigation task (Simulation and Task Environment Section) for one *robot lifetime* (table 1), and fitness assigned to the tested genotype. One generation was when all genotypes

had been tested and evaluated. Selection and mutation operators were then applied. The mutation operators specified in table 3 were applied to every node of the child GRNs with the probability specified in table 4. However, the `mutate_weight` operator was applied to every connection of the GRNs with a lower probability (table 4).

Experiments, Results, Discussion

Controller evolution was run for 10000 generations in the *way-point navigation* task, where robot controllers were encoded using direct binary or indirect GRN encoding. In order to investigate the conditions facilitating evolvability in this evolutionary robotics case study, task variants were switched every 200 generations. Hence, given these two controller encodings, two different evolutionary setups were run, where each setup was run 100 times to ensure viability of statistical tests on results data.

Figure 2 presents task performance (average and best fitness) results for the way-point navigation task. Table 6 presents average fitness results and table 7 presents statistical test results for within and between comparisons of direct and GRN encoded populations. Table 7 presents statistical test results from pair-wise comparisons on average maximum and average fitness results in table 6. The Mann-

Parameter	Value
Population size	1000
Genotypes replaced per generation	100
Tournament size	4
Recombination	None
Generations per task variant switch	200
Number of generations	10000
Binary encoding mutation	Bit-flip
Binary encoding mutation rate	See table 5
GRN mutation	See table 3
GRN weight mutation rate	0.002
GRN node mutation rate	0.02
Genotype bit-string length	60

Bit flips	Probability
0 / 13	0.5088 / 0.0023
1 / 14	0.2466 / 0.0017
2 / 15	0.1068 / 0.0012
3 / 16	0.0471 / 0.0009
4 / 17	0.0231 / 0.0006
5 / 18	0.0139 / 0.0004
6 / 19	0.0103 / 0.0003
7 / 20	0.0087 / 0.0002
8 / 21	0.0076 / 0.0001
9 / 22	0.0065 / 0.0001
10 / 23	0.0053 / 0.0000
11 / 24	0.0042 / 0.0000
12 / 25	0.0032 / 0.0000

Table 4: Evolutionary Algorithm Parameters

Table 5: Mutation rates for direct binary encoding. Probabilities match mutation probabilities of the binary GRN encoding (for given number of bit flips). Genotypes were sixty gene bit-strings, however, mutations of more than twenty-three bit flips never occurred

Whitney U test ($p < 0.01$) with Bonferroni correction (Flannery et al., 1986) for multiple comparisons was applied to gauge statistical significance.

Table 6 presents average and maximum fitness results. These fitness results are presented for *early* and *late* stages of the evolutionary process. Early stages were at generation 25 and 425 and late stages were at generations 9225 and 9625. These generation intervals were chosen as they were an eighth into the allotted generations for task variant one, and were deemed a good measure of the population's early response to the environmental change early and late in the artificial evolution process.

One may note that we measure average and maximum fitness, rather than the rate of fitness change (relative fitness) as an indicator of a population's responsiveness. We elected to use absolute fitness, since measuring relative fitness would unfairly benefit genotypes whose fitness dropped the most after a change in the task environment. That is, given two populations, the one with the highest fitness a given interval after a task change is concluded to be the most responsive. Also, this interpretation holds in the case that the other population suffers a greater fitness decrease after the task change which might subsequently lead it to having a faster fitness increase.

In figure 2 one may observe that populations using the GRN encoding were able to significantly improve (with statistical significance, table 7) their responsiveness between early and late stages of the evolutionary process. Here we use *responsiveness* and *evolvability* interchangeably to mean the speed with which a population adapts given task environment changes (task variants). The responsiveness of populations using the direct encoding was statistically comparable

during both the early and late stages of the evolutionary process (table 7).

Hence, statistical tests run between average and average maximum fitness values of populations using the GRN encoding in the early and late stages of evolution (table 7) confirm that in the late stages of evolution, populations respond more quickly to environmental change (task variants). Results indicate that populations using the GRN encoding were significantly more responsive in the late generations compared to populations using the direct encoding (table 7). Also, there was not a significant difference in the responsiveness of populations using direct encoding between the early and late stages of evolution. This is supported by previous work (Crombach and Hogeweg, 2008; Draghi and Wagner, 2008) and supports this study's hypothesis that direct genotype encoding does not facilitate responsiveness in changing environments (Introduction section). Although, in the early stages of evolution, the average fitness of populations using the direct encoding was more responsive than populations using the GRN encoding (figure 2).

In terms of the responsiveness of populations using the direct encoding, there was no significant difference between the average and average maximum fitness values of these populations between early and late stages of evolution (table 7). This indicates that direct binary encoded populations did not evolve a responsiveness to the changing task environment of this way-point navigation task, as was observed in the case of GRN encoded populations (table 7). However, in terms of the average fitness, the direct encoding was significantly more responsive in the early stages of evolution.

Thus, results demonstrate that a GRN encoding of populations of robot controllers (behaviors) evolve to become more
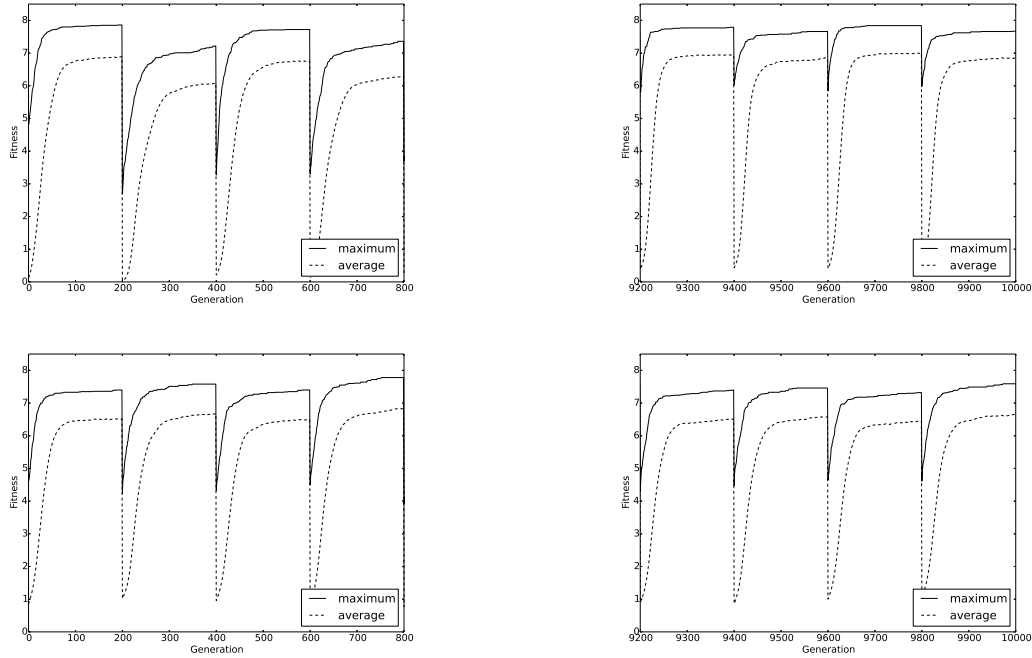
Figure 2: Average and maximum fitness for evolved GRN (top left, right) and binary encoded (bottom left, right) controllers in the *way-point navigation* task.

responsive to changes in their task environment. That is, a statistically significant difference (for *average* and *maximum* fitness values) was observed in the improvement of responsiveness of GRN versus directly binary encoded populations in the *way-point navigation* task. The contribution is that results support previous work on the efficacy of GRN encodings for conferring evolvability in changing task environments (Crombach and Hogeweg, 2008), as well as extending previous work (Crombach and Hogeweg, 2008; Draghi and Wagner, 2008) into an evolutionary robotics task environment with both time and space dynamics.

A goal of this study was to investigate the environmental and evolutionary conditions that facilitate the evolution of evolvability. Previous researchers have demonstrated that many-to-one genotype-to-phenotype mapping (redundant mappings) result in evolvability in EAs (Shipman, 1999; Ebner et al., 2001a) as well as increased EA task performance. That is, a highly redundant mapping enables some mutations to have negligible impact on the fittest phenotypes, meaning the EA is better able explore the search space via neutral networks (Ebner et al., 2001b).

Redundancy, and the closely related notion of robustness (Wagner, 2005), is theorized to have played a key role in the increased responsiveness of evolved GRN encoded behaviors, given that the GRN encodings are more redundant than the direct encodings. During their evaluation, GRN's are decoded into bit-strings before these bit-strings are de-

coded into ANN's, which are then evaluated by the EA. Given that the decoding from GRN's into bit-strings is a many-to-one mapping, the redundancy in the GRN encoded search space is considerably higher than that in the directly encoded search space.

That is, the bit-strings are sixty characters long, which implies that there are $2^{60} \approx 10^{18}$ genotypes in this space. In the GRNs, each node has fifty-nine possible connections and each connection can either connect to one of the sixty nodes or not connect. This implies that each node has 59^{61} possible configurations. Note that although many of these configurations are equivalent, where the only difference is the ordering of the connections, each one forms a distinct encoding and thus represents part of a distinct genotype. There are sixty nodes, so the number of genotypes in this space is $\left(59^{61}\right)^{60} \approx 10^{6000}$.

Also, consider that in this study's way-point navigation task, there were only nine fitness values, where fitness was equated with how many of eight way-points a robot passed in its lifetime. The ninth fitness value was to account for the robot not passing any way-points (Simulation and Task Environment Section). Thus, given the GRN encoding, nine possible phenotypes (way-point navigating behaviors) were represented by a high dimension and highly redundant genotype space. That is, in this task, there were many possible genotype to phenotype (controller) mappings, where controller behavior was equated with one of nine fitness values.

Measure	Fitness
Early maximum GRN encoding	0.88 (0.18)
Late maximum GRN encoding	0.96 (0.11)
Early maximum direct encoding	0.86 (0.17)
Late maximum direct encoding	0.85 (0.17)

Measure	Fitness
Early average GRN encoding	0.33 (0.11)
Late average GRN encoding	0.45 (0.16)
Early average direct encoding	0.39 (0.15)
Late average direct encoding	0.39 (0.16)

Table 6: Maximum (**left**) and average (**right**) fitness and standard deviations (in parentheses) for *way-point navigation* for *early* and *late* stages of evolution. Results have been normalized, where given values are portions of the minimum and maximum possible task performance: 0 and 1.0, respectively. Early stages were at generation 25 and 425 and late stages were at generations 9225 and 9625.

	EM GRN	LM GRN	EM DE	LM DE	EA GRN	LA GRN	EA DE	LA DE
EM GRN	•	✓	✗	•	•	•	•	•
LM GRN	✓	•	•	✓	•	•	•	•
EM DE	✗	•	•	✗	•	•	•	•
LM DE	•	✓	✗	•	•	•	•	•
EA GRN	•	•	•	•	•	✓	✓	•
LA GRN	•	•	•	•	✓	•	•	✓
EA DE	•	•	•	•	✓	•	•	✗
LA DE	•	•	•	•	•	✓	✗	•

Table 7: Statistical test results from pair-wise comparisons on average fitness results in table 6. ✓ signifies a statistically significant difference between two data-sets ($p < 0.01$) using the Mann-Whitney U test and Bonferroni correction. ✗ signifies that the difference between two data-sets is not significant and • signifies that a test was not done. EM is an abbreviation for *Early Maximum* (average maximum fitness in early evolution), LM is *Late Maximum*, EA is *Early Average* (average fitness in early evolution), LA is *Late Average*, and DE is *Direct Encoding*.

It is theorized that the larger size of the space of the GRN encoding is the cause of it exhibiting lower evolvability in the early stages of evolution. That is, given that phenotypes may not be uniformly distributed over the space (Pigliucci, 2010; Parter et al., 2008), finding target phenotypes after initialization may be more challenging. During the course of evolution, however, the population can move to areas of the space biased towards the targets. Moreover, other work has shown that direct encodings do exhibit a baseline of evolvability that is comparable to certain generative encodings (Tarapore and Mouret, 2015).

This study's results are also supported by related work (Ciliberti et al., 2007), that similarly modeled GRNs, where GRN instances were individual genotypes decoded into expression patterns (phenotypes). Ciliberti et al. (2007) discovered that such a GRN encoding was robust (and redundant) as a large number of genotypic changes had no phenotypic impact.

To demonstrate this for our experimental results, current research is investigating the relationship between robustness, redundancy and evolvability for the GRN versus directly encoded search spaces. This is being done for way-point navigation and more complex evolutionary robotics tasks.

Conclusion

This research presented an evolutionary robotics study that replicated and extended previous work testing the *evolvability* of populations of *Gene Regulatory Networks* (GRNs). Evolvability was defined as a population's speed of adaptation to changing task environments. Direct binary encodings of robot controllers were compared to indirect GRN encodings in controller evolution to accomplish a way-point navigation task. Task variants were alternated during controller evolution to confirm previous results that GRNs facilitate the emergence of evolvability in environments with alternating tasks. Results indicated that, for the GRN encoding approach, populations became significantly more adapted to task variation over time, and thus evolvable. This was compared to a direct encoding of controllers which was unable to achieve a high level of evolvability in the same task environment. This work thus demonstrates that the previous results are valid in a substantially more complicated domain and suggests approaches for aiding robots in dealing with dynamic environments. The findings were theorized to be a result of increased redundancy and robustness of the indirect GRN encoding of the search space. However, definitively demonstrating increased robustness and redundancy resulting in increased evolvability for the GRN encoding of this and other more complex evolutionary robotics tasks remains the subject of ongoing research.

References

Bredeche, N., Montanier, J.-M., Weel, B., and Haasdijk, E. (2013). Roborobo: A fast robot simulator for swarm and collective robotics. *arXiv preprint*.

Ciliberti, S., Martin, O., and Wagner, A. (2007). Innovation and robustness in complex regulatory gene networks. *Proceedings of the National Academy of Sciences*, 104(34):13591–13596.

Crombach, A. and Hogeweg, P. (2008). Evolution of evolvability in gene regulatory networks. *PLoS Comput Biol*, 4(7):e1000112.

Draghi, J. and Wagner, G. P. (2008). Evolution of evolvability in a developmental model. *Evolution*, 62(2):301–315.

Draper, D. (1992). The rna-folding problem. *Acc. Chem. Res.*, 25(4):201–207.

Ebner, M., Langguth, P., Albert, J., Shackleton, M., and Shipman, R. (2001a). On neutral networks and evolvability. In *Proceedings of the 2001 Congress on Evolutionary Computation*, pages 1–8. IEEE.

Ebner, M., Shackleton, M., and Shipman, R. (2001b). How neutral networks influence evolvability. *Complexity*, 7(2):19–33.

Eiben, A. and Smith, J. (2003). *Introduction to Evolutionary Computing*. Springer, Berlin, Germany.

Flannery, B., Teukolsky, S., and Vetterling, W. (1986). *Numerical Recipes*. Cambridge University Press, Cambridge, UK.

Flatt, T. (2005). The evolutionary genetics of canalization. *The quarterly review of biology*, 80(3):287–316.

Gjuvsland, A., Vik, J., Beard, D., Hunter, P., and Omholt, S. (2013). Bridging the genotype-phenotype gap: what does it take? *J Physiol.*, 591(8)::2055–2066.

Grefenstette, J. (1999). Evolvability in dynamic fitness landscapes: A genetic algorithm approach. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3. IEEE.

Kirschner, M. and Gerhart, J. (1998). Evolvability. *Proceedings of the National Academy of Sciences*, 95(15):8420–8427.

Lehman, J. and Stanley, K. (2013). Evolvability is inevitable: Increasing evolvability without the pressure to adapt. *PloS one*, 8(4):e62186.

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press, Cambridge, USA.

Parter, M., Kashtan, N., and Alon, U. (2008). Facilitated variation: how evolution learns from past environments to generalize to new environments. *PLoS Comput Biol*, 4(11):e1000206.

Pigliucci, M. (2008). Is evolvability evolvable? *Nature Reviews Genetics*, 9(1):75–82.

Pigliucci, M. (2010). Genotype phenotype mapping and the end of the genes as blueprint metaphor. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 365(1540):557–566.

Rothlauf, F. (2006a). Introduction. In *Representations for Genetic and Evolutionary Algorithms*, pages 1–7. Springer Berlin Heidelberg.

Rothlauf, F. (2006b). *Representations for Genetic and Evolutionary Algorithms*. Springer-Verlag, Berlin, Germany.

Shipman, R. (1999). Genetic redundancy: Desirable or problematic for evolutionary adaptation. In *Proceedings of the 4th International Conference on Artificial Neural Networks and Genetic Algorithms*, pages 337–344. Springer.

Simões, L. F., Izzo, D., Haasdijk, E., and Eiben, A. E. (2014). Self-adaptive genotype-phenotype maps: neural networks as a meta-representation. In *Parallel Problem Solving from Nature–PPSN XIII*, pages 110–119. Springer.

Stanley, K. and Miikkulainen, R. (2003). A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130.

Tarapore, D. and Mouret, J.-B. (2014). Comparing the evolvability of generative encoding schemes. In *Proceedings of ALife*, pages 1–8. MIT Press.

Tarapore, D. and Mouret, J.-B. (2015). Evolvability signatures of generative encodings: beyond standard performance benchmarks. *Information Sciences*, 313:43–61.

Wagner, A. (2005). Distributed robustness versus redundancy as causes of mutational robustness. *Bioessays*, 27(2):176–188.

Wagner, G. and Altenberg, L. (1996a). Complex adaptations and the evolution of evolvability. *Evolution*, 50(1):967–976.

Wagner, G. and Altenberg, L. (1996b). Perspective: Complex adaptations and the evolution of evolvability. *Evolution*, pages 967–976.

Wilder, B. and Stanley, K. (2015). Reconciling explanations for the evolution of evolvability. *Adaptive Behavior*.

The Evolution of Evolvability: Changing Environments Promote Rapid Adaptation in Digital Organisms

Rosangela Canino-Koning[1,2], Michael J. Wiser[2,3], and Charles Ofria[1,2,3]

[1]Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA
[2]BEACON Center for the Study of Evolution in Action, Michigan State University, East Lansing, MI, USA
[3]Ecology, Evolutionary Biology, and Behavior, Michigan State University, East Lansing, MI, USA
caninoko@msu.edu

Abstract

Genetic spaces are often described in terms of fitness landscapes or genotype-to-phenotype maps, where each potential genetic sequence is associated with a set of properties and connected to other genotypes that are a single mutation away. The positions close to a genotype make up its "mutational landscape" and, in aggregate, determine the short-term evolutionary potential of a population. Populations with wider ranges of phenotypes in their mutational neighborhood tend to be more evolvable. Likewise, those with fewer phenotypic changes available in their local neighborhoods are more mutationally robust. As such, forces that alter the distribution of phenotypes available by mutation can have a profound effect on subsequent evolutionary dynamics.

We demonstrate that cyclically-changing environments can push populations toward more evolvable mutational landscapes where a wide range of alternate phenotypes are available, though purely deleterious mutations remain suppressed. We further show that populations in environments with drastic changes shift phenotypes more readily than those in environments with more benign changes. We trace this effect to repeated population bottlenecks in the harsh environments, which result in shorter coalescence times and keep populations in regions of the mutational landscape where the phenotypic shifts in question are more likely to occur.

Introduction

Fitness landscapes are a mathematical tool to map genetic sequences to expected evolutionary fitness. Many studies have examined the important role that different types of fitness landscapes play on evolutionary dynamics and outcomes, both in biological populations (Khan et al., 2011; Szendro et al., 2013; Weinreich et al., 2006; Nahum et al., 2015) and in evolutionary computation settings (Merz and Freisleben, 2000; Humeau et al., 2013; Kallel et al., 2013). However, real-world fitness landscapes are far more complex and varied than the idealized models that are used in most of these studies. Neighboring regions of real landscapes can have starkly different properties from each other based on the effects of and interactions among mutations (i.e., the mutational landscape). Examples of the type of properties that we are interested in include robustness, epistasis, and modularity, all of which are measurements of how information is organized inside of a genome and commonly categorized as components of an organism's "genetic architecture". Isolated pockets in a landscape can often be characteristically different from the landscape as a whole due to the amount and organization of genetic information. In fact, in most natural fitness landscapes, the vast majority of neighborhoods consist entirely of non-replicating genomes with zero fitness (and thus no genetic information), making life itself appear to be a rare exception (Gavrilets, 2004).

Evolution on these convoluted landscapes is clearly limited to those regions that have non-zero fitness, with a selective pressure for fitness to increase. Beyond that, however, populations can evolve toward neighborhoods with specific local properties based on the evolutionary forces acting upon the populations. For example, high mutation rates drive populations toward neighborhoods with a higher fraction of neutral mutations in the effect dubbed survival of the flattest (Wilke et al., 2001). Similarly, sexual populations tend toward regions of the fitness landscape with more modularity (Misevic et al., 2006) and more negative epistasis (Misevic et al., 2010) than otherwise equivalent asexual populations.

Understanding these dynamics is of broad interest. It is important to evolutionary computation given the strong influence of local landscape properties on the quality of the final solutions that an evolving population is able to obtain. Its relevance to evolutionary biology is equally obvious – the local landscape that a population occupies will influence the selective forces at play in the population, creating a feedback cycle between these two important evolutionary factors (Zaman et al., 2014; Meyer et al., 2012). Disentangling such interactions is likely to provide further insights into fundamental evolutionary dynamics. Computational artificial life systems have the advantage of being able to bridge these two realms: they have unconstrained evolutionary dynamics similar to natural systems, while maintaining the ability to rapidly perform experiments and collect any data we need about populations or their local landscapes.

Evolvability and Genetic Architecture

Evolvability refers to a series of distinct but overlapping concepts that are generally concerned with adaptation, variation, and/or novelty generation (Pigliucci, 2008). For the purposes this paper, we will focus on evolvability as the capability of genomes to generate adaptive variation in response to mutation. This kind of evolvability depends primarily on the organization and interrelation of information in the genome; that is, the genetic architecture, and the resulting genotype-to-phenotype map (Gunter P. Wagner and Altenberg, 1996). An example of evolvable architecture can be found in some bacterial genomes that contain highly mutable genome regions, called contingency loci. Small sets of insertions or deletions to these regions create transcription frameshifts that alter the expression of nearby coding regions, thus allowing populations to easily switch phenotypes via minor mutations. Contingency loci are most often seen in the genomes of pathogens, which are subject to frequent environmental shifts caused by the host immune system (Bayliss et al., 2001). Thus, these populations are able to produce large amounts of heritable variation despite the reduction in population diversity resulting from population bottlenecks.

Mutational Landscapes Properties of genetic architectures such as evolvability and robustness are determined by the shape of the resulting mutational landscape (Andreas Wagner, 2008). Robust genetic architectures that can tolerate more mutations without altering their phenotype reside in mutational landscapes that connect to more neutral mutants. Similarly, architectures that more easily switch phenotypes in response to mutation without substantial reduction in fitness, reside in more evolvable regions of genotype-space.

It is worth noting that not all regions of the mutational landscape are equally accessible. Some genome regions may be more resistant to mutation than others (Lee et al., 2012), thereby altering the probabilities of mutations occurring that lead into certain regions of the mutational landscape. This kind of differential probability may therefore moderate a population's diffusion through the mutational landscape. Further, in regions of the landscape where there are fewer available mutations that provide potentially adaptive traits, response to selection is likely to be weaker than in regions where there are many adaptive variants available within a few mutational steps (Alberch, 1991; Carter et al., 2005).

Changing environments create more paths to different kinds of phenotypes

Directional selection acts to change the composition of phenotypes and genotypes in a population (Wright, 1931). This change moves the population across the mutational landscape to local regions of higher fitness. As populations ar-

rive at a fitness peak, they tend to cluster there, and the accumulation of new phenotype-altering mutations decreases (Wright, 1964; Kauffman and Levin, 1987). In changing environments, however, the direction of selection is not fixed. Instead, as the environment changes, populations are driven to explore new regions of the mutational landscape (Kashtan et al., 2007; Connelly et al., 2015). As they proceed, populations accumulate and carry with them the history of prior explorations and adaptations, and use them as raw material for new adaptation (McClintock, 1993). Indeed, earlier work has shown that changing environments promote evolvability in many contexts, without compromising robustness(Crombach and Hogeweg, 2008; Wilke et al., 2001). Strength of selection is also an important component of this exploration, since the harshness of the environment drives the speed with which organisms adapt to new conditions (Goddard et al., 2005).

In this paper, we show how changing environments not only drive exploration of the mutational landscape, but also create populations whose genetic architectures are qualitatively different than those from populations evolved in static environmental conditions. In particular, we show that populations evolved under harsh cyclically-changing environments have many more changes along their phylogenetic histories than those evolved in static or benign changing environments. They also individually contain large reservoirs of pseudogene-like vestigial loci that were acquired and deactivated through repeated adaptation and fixation cycles. As a result, populations evolved in these harsh cyclically-changing environments are low in standing neutral diversity at the population level, but they still connect with many more phenotypically-interesting regions of the mutational landscape than more diverse populations evolved in static or benign environments.

Digital Evolution

Digital Evolution is a sub-field of Artificial Life that focuses on studying evolutionary dynamics using self-replicating computer programs as model organisms (McKinley et al., 2008). Unlike theoretical simulations, digital organisms self-replicate, mutate, and compete with their peers for resources and space in which to reproduce. Because populations of digital organisms have a source of variation, inheritance of genetic material across generations, and are subject to selective pressures, they undergo evolution by natural selection.

Digital organisms do not suffer from many of the drawbacks of experimentation on natural organisms. Three of the advantages of digital organisms are particularly relevant for our study. First, the rates of reproduction in digital systems are much faster than in even the most rapidly-reproducing physical organisms; we can get generations in seconds, rather than the hours for the fastest biological organisms (Ryan, 1953; Lenski et al., 1991), or the days

or even weeks needed for complex multicellular organisms (Anderson et al., 2010; Stearns et al., 2000).Second, using digital organisms allows us to tightly control and verify experimental conditions. For example, in physical organisms, factors such as mutation rate can generally only be measured after the fact, or coarsely altered through mutagens. In digital organisms, however, we can not only control mutation rates with fine-grained precision, but also types and probabilities of different types mutations (e.g., point mutations, insertions, deletions). Further, we are also able to track and replay the evolutionary history of every organism at any point in time to verify that unusual or unexpected results do not represent measurement error. This ability to exactly replicate evolutionary results at an individual organism level is firmly out of reach for experiments with physical organisms. Finally, we can precisely and exhaustively map the mutational landscape of a digital organism, and identify the role of every site in its genome(Ofria and Adami, 2002); this is not feasible in even the simplest physical organisms. All of these factors make digital organisms ideal for studying the effects of changing environments on the mutational landscape.

Methods

Avida Digital Evolution Platform

We used Avida (Lenski et al., 2003) to examine the effects of cyclic changing environments on the genomes of evolved digital organisms. Avida is a software platform for performing evolution experiments with digital organisms in a virtual world.
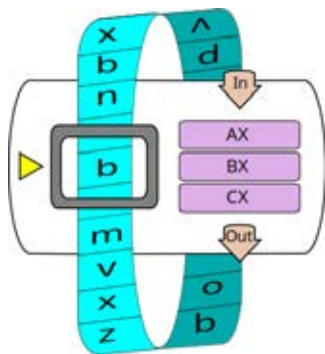
Figure 1: **An example virtual CPU from Avida**, with a circular genome (blue), three registers (purple), input and output handlers (tan), and an instruction pointer (yellow) indicating the next instruction to be executed.

An Avida organism is composed of a circular genome of assembly-like computer instructions that are executed in a virtual CPU (Figure 1). Populations of these organisms are placed in a toroidal world in individual cells where they are allowed to execute, reproduce, compete for space, mutate, and evolve.

Organisms in Avida are self-replicating, and experience mutation. The genome in the initial default organism contains all of the instructions necessary for reproduction. However, the instructions are not copied into an offspring with perfect fidelity. By default, the reproductive copy instruction is faulty, meaning that it will probabilistically introduce errors (mutations) into the offspring genomes. These offspring organisms carry and execute the mutations to their genomes, and in turn pass them on, along with new mutations, to their own offspring (i.e., variation in the systems is heritable).

Avida worlds can be space- or resource-constrained. Avida allows the experimenter to configure many aspects of the environment, thus subjecting the organisms to various kinds of selective pressures. In many cases, these environments will include resources that can be metabolized by performing specific functions or activities, resulting in a boost to execution speed that gives the organisms a competitive advantage. However, even without explicit external pressures, organisms still experience an implicit pressure to execute more quickly and efficiently. The organisms that run fastest are typically able to also reproduce fastest, and thus outcompete their peers for space.

Thus, because populations have a source of variation, inheritance, and experience selection, evolution by natural selection is an inevitable consequence. Further, because the Avida genome instruction set is Turing-complete[1], populations may evolve potentially infinite complexity of behavior(Ofria et al., 2002).

Avida is available for download without cost from `http://avida.devosoft.org/`, and specific versions along with data-files to reproduce the experiments described in this paper may be found at `https://github.com/voidptr/avida` and `https://github.com/voidptr/alife2016`.

Experimental Design

We subjected a total of 150 replicate populations of digital organisms to two different treatments of two-phase cyclical changing environments, plus a static control. The environment cycles between equal-length periods of reward and punishment. Each cycle extends for 1000 updates, or roughly 30 generations. In the static control, there is no cycle. Rather, the rewards remain constant. The complete experiment extends for 200 cycles, or 200,000 updates, approximately 6,000 generations.

We structured the environment to provide large rewards to organisms for performing two challenging bit-wise logical tasks: XOR and EQU. XOR is rewarded with a CPU speed (and thus fitness) multiple of 8, while EQU is rewarded with a CPU speed multiple of 32. In the harsh treatment, as the cycle progresses, the XOR reward remains constant, while

---

[1]The Avida instruction set is a super-set of the Tierra instruction set, which has been shown to be Turing-complete(Maley, 1994).

the EQU reward cycles between a 32-fold bonus and a correspondingly harsh 32-fold penalty (i.e., CPU speed is divided by 32 when EQU is performed in the off cycle). The benign treatment is identical to the harsh treatment, except that the reward merely goes away in the off-cycle as opposed to incurring a severe penalty.

We identify EQU as the *Fluctuating Task*. XOR, because it is rewarded continuously, is the *Backbone Task*, and is used as a background for comparing the separation or intertwining of functional genetic components in the evolution of EQU. Further, the 4-fold difference in reward level between XOR and EQU encourages the evolution and maintenance of EQU when possible.

For all of the experiments described in this section, we held the individual genomes at a fixed length of 121^2 instructions instructions, but mutated the new genome after each successful replication event at a substitution probability of 0.00075 per site. We configured the Avida world to have local interactions on a toroidal grid that is 60 cells by 60 cells (3600 cells in total), and we seeded the initial populations with an ancestor that was previously evolved to perform XOR and EQU under a static reward. The genetic architecture for performing XOR and EQU is tightly intertwined in this ancestral organism, as it was evolved with no selective pressure for modularity.

Results and Discussion

Our experiments demonstrate that digital organisms that were evolved in cyclic changing environments differ substantially from those evolved in static environments in a number of ways. These differences include the number of mutations that fix in the lineage from the ancestor (the "phylogenetic depth"), key metrics of their genetic architecture, and the presence of reservoirs of pseudogenes that change the nearby mutational landscape.

Evolutionary History and Population Structure   Evolution in the harsh changing environment resulted in populations with substantially higher phylogenetic depth as compared to those evolved in static or benign environments. At each environmental shift, adaptive mutations rapidly swept and fixed in the populations. (Figure 2)

The populations evolved in the control and benign environments displayed much more genetic diversity compared to those evolved in the harsh cyclic environment, which underwent what was effectively a bottleneck at each cycle shift. Because a selective sweep reduces current diversity within a population, the smaller number of sweeps in the benign and control treatments led populations in them to

[2]As part of the initial exploratory protocol, we hand-wrote an organism with separated sections that performed XOR and EQU. In order to compare the hand-written organism with a sample of evolved organisms, we matched their genome lengths, which were 121 instructions.
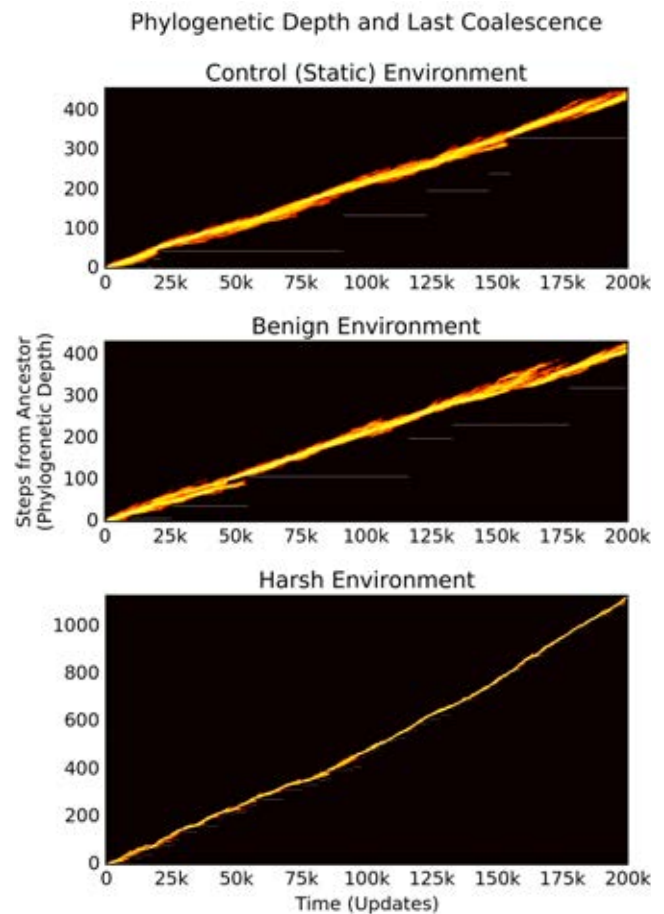
Figure 2: **Phylogenetic depth of representative populations** evolved in each of the three treatments. White horizontal lines mark the depth of the most recent common ancestor, and discontinuities in this line indicate that the most recent common ancestor has changed, and thus that a sweep occurred. The control treatments had a mean of 18 sweeps (STD=9.05), the benign treatments had a mean of 21 (STD=19.05), and the harsh treatments had a mean of 88 sweeps (STD=23.37). Note the difference in scales between y-axes: the control-evolved population has a maximum depth of 400 mutational steps from ancestor, while the harsh-evolved has upward of 1100.

have higher standing diversity for most of their evolutionary history than those populations from the harsh changing environment. Despite this higher standing diversity in the benign and control treatments, regions of low diversity are still evident in the genomes of these populations, implying purifying selection on the traits encoded at these sites (see Figure 3).

Genetic Architecture   The selective shifts in both benign and harsh changing environments result in qualitatively dif-

Population Entropy by Site and Genotype

tuating task (EQU) were separated from the backbone task (XOR), except for a core region of overlap, which represent portions of the tasks that are shared between XOR and EQU. (Figure 4)
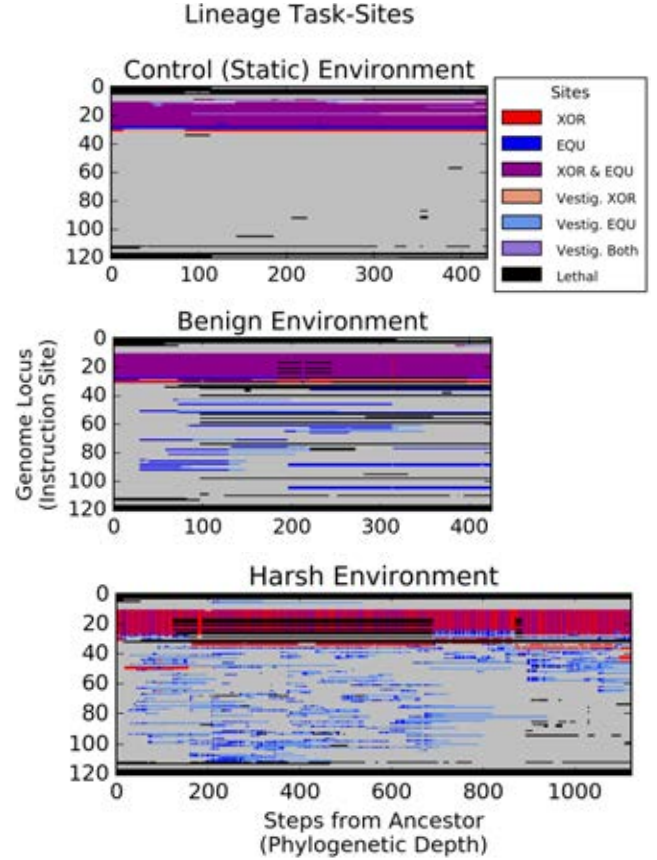
Figure 4: **Varying genetic architecture of XOR and EQU over time** for the final dominant genotype in a randomly selected replicate. Proceeding from the left of each figure, each vertical slice along the X-axis represents an ancestor of the final dominant. The Y-axis represents the tasks coded for at each genome locus. Sites in **red** are active sites that code for the XOR task only, sites in **blue** are active sites for the EQU task only, and **purple** sites code for both XOR and EQU. Knockouts to the sites in black are lethal to the organism. Sites in the lighter colors (tan, light blue, lavender) represent vestigial sites for XOR only, EQU only, or both tasks, respectively. As we proceed from left to right, we can see the evolutionary history of the final dominant genotype.

Figure 3: **Population Per-site Entropy over time.** Each vertical slice represents the per-site entropy of the population at each update, both by genetic locus (upper), and overall population mean (lower). Hotter colors (upper) indicate greater diversity at this locus. Mean population entropy indicates the relative diversity of the population at any given time, while the per-site entropy shows where in the genomes the population diversity is located.

ferent architectural styles from the static control environment. The task arrangements evolved under both experimental treatments are much more scattered than in the control. The bulk of the sites responsible for performing the fluc-

In contrast, the architecture of XOR and EQU remain tightly intertwined in the control, and site positions do not change significantly over the course of the experiment. In the benign treatment, many more regions that perform the fluctuating task (XOR) are scattered throughout the genome, but site positions remain relatively static throughout the run after an initial adaptive phase. In the harsh treatment, not

only are the active sites scattered, but the positions of active sites change and proliferate wildly.

Interestingly, populations evolved in both the benign and harsh treatments also show development of a large reservoir of formerly functional, now vestigial, sites; that is, sites that were previously active in performing a task, but were disabled by a mutation elsewhere and are now neutral. However, these vestigial pseudogene-like sites appear to be important for allowing the organisms to quickly re-adapt as the fluctuations in the environment restore the previously-rewarded functions. (Figure 5)
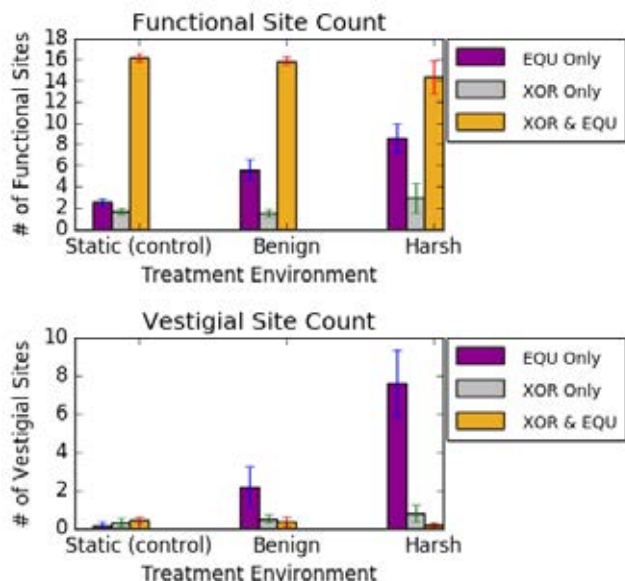
Figure 5: **Number of functional and vestigial sites by treatment**. The harsh environment has a significantly larger number of vestigial sites for the fluctuating (EQU) task compared to the benign treatment or control, while having a comparable number of functional sites (One-Way ANOVA $F(2,132) = 54.35$, $p \ll 0.0001$).

Nearby mutational landscape In order to identify the role that these pseudogene-like structures play, we performed a survey of the single-step mutational landscape surrounding the last common ancestor of each replicate population. This landscape contained approximately 3,200 distinct mutants in each of the 50 replicates per treatment, for a total of almost 500,000 mutants surveyed. We found that the availability of reservoirs of vestigial sites shifted the treatment-evolved organisms' position in genotype space such that, compared to the control-evolved organisms, a task lost due to mutation more often remains one or two mutational steps away. In this way, the treatment organisms have an advantage over organisms from the control runs in terms of the short-term evolvability of the fluctuating task. (Figures 6 and 7)
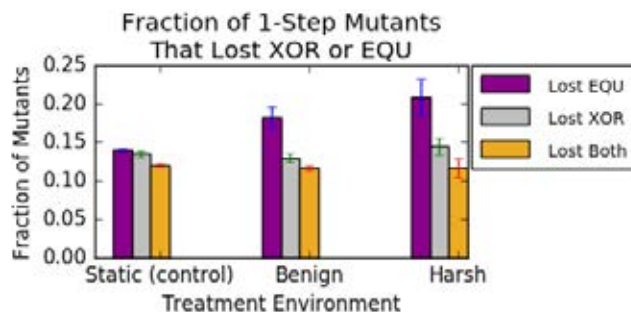
Figure 6: **A survey of the single-step mutational neighborhood** around organisms that performed the fluctuating task. Note that in both the benign and harsh treatments, there were significantly more mutants that lost the EQU task as compared to the control (Wilcoxon Rank Sum Test: $Z = -6.59$ and -6.70 respectively, $p \ll 0.0001$). This result indicates that it was easier for the organisms in both treatments to turn off the EQU task in response to one mutation.
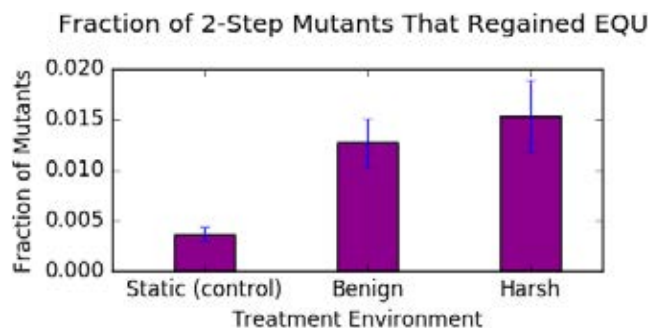
Figure 7: **A survey of the two-step mutational neighborhood** of the organisms that lost EQU function in the one-step survey. We found that in both the harsh and benign treatments, there were significantly more organisms that regained function in response to mutation than the control. (Wilcoxon Rank Sum Test: $Z = -6.11$ and -7.38 respectively, $p \ll 0.0001$). This result indicates that it was easier for the organisms in both treatment environments to regain the task in response to one additional mutation.

Conclusion

In cyclic changing environments, the direction of selection shifts frequently, and periodically drives populations to not only explore new regions of the genetic landscape, but also to carry with them the genetic heritage of previous environmental adaptations. Thus, the resulting populations are not only adapted to the local temporarily static environment, but also to the meta-environment of cyclic change. Because of their mutational history, and the paths that led them to their current region of genotype space, the genomes contain vestigial fragments of genetic material that were adapted to prior

environments. As this exploration proceeds, more mutations accumulate in the population, and each of these creates a link to a new region of the mutational landscape. As these links accumulate, they form a reservoir of mobility for the population to quickly shift to new phenotypes as dictated by shifting selective forces. In this way, the accumulation of vestigial or pseudogene-like regions acts as an adaptation to the larger pattern of changing selective forces.

By contrast, in static (non-changing) environments, the majority of neutral mutations do not connect to as many phenotypically-interesting regions of genotype-space. There are far fewer pseudogenes-like regions available that could regain functionality should conditions change. Thus, populations evolved in static environments are less evolvable in the short-term.

Limitations of Changing Environments and Future Directions

Changing environments produce unique sets of selective pressures that promote more rapid exploration of genotype space, while also building useful reservoirs of partial functionality that may be co-opted in the evolution of more complex structures. These features make changing environments useful for both their explanatory power in natural evolution, and as practical tools in the Artificial Life toolkit. Ultimately, however, cyclic changing environments only retread existing phenotypic ground, and though genotypic exploration is wider and faster than under purely directional or stabilizing selection, the space explored remains limited to the scope of the phenotypes that are being selected for.

Even so, there must certainly exist other methods of exploring genotype space that do not suffer from these limitations. For example, perhaps repeated bottlenecking of populations could promote faster traversal of the fitness landscape in quasi-random directions. Another alternative may be randomly changing environments, rather than cyclic, which might produce a broader exploration of the wider genotype space. Finally, perhaps these kinds of environments could be coupled with dynamically increasing open-ended complexity goals.

Understanding the mechanisms by which different types of environments alter fitness landscapes is vital to developing an understanding of the forces that promote evolvability and increase complexity. Cyclic changing environments provide one view into these dynamics, but we must explore further to find other mechanisms for exploring and exploiting genotype space.

Acknowledgments

References

Alberch, P. (1991). From genes to phenotype: dynamical systems and evolvability. *Genetica*, 84(1):5–11.

Anderson, J. L., Morran, L. T., and Phillips, P. C. (2010). Outcrossing and the Maintenance of Males within C. elegans Populations. *J Hered*, page esq003.

Andreas Wagner (2008). Robustness and evolvability: a paradox resolved. *Proc. R. Soc. B*, 275(1630):91–100.

Bayliss, C. D., Field, D., and Moxon, E. R. (2001). The simple sequence contingency loci of Haemophilus influenzae and Neisseria meningitidis. *J. Clin. Invest.*, 107(6):657–666.

Carter, A. J. R., Hermisson, J., and Hansen, T. F. (2005). The role of epistatic gene interactions in the response to selection and the evolution of evolvability. *Theor. Popul. Biol.*, 68(3):179–196.

Connelly, B. D., Dickinson, K. J., Hammarlund, S. P., and Kerr, B. (2015). Negative niche construction favors the evolution of cooperation. *Evol Ecol*, pages 1–17.

Crombach, A. and Hogeweg, P. (2008). Evolution of Evolvability in Gene Regulatory Networks. *PLOS Comput Biol*, 4(7):e1000112.

Gavrilets, S. (2004). *Fitness landscapes and the origin of species (MPB-41)*. Princeton University Press Princeton, NJ.

Goddard, M. R., Godfray, H. C. J., and Burt, A. (2005). Sex increases the efficacy of natural selection in experimental yeast populations. *Nature*, 434(7033):636+. 636.

Gunter P. Wagner and Altenberg, L. (1996). Perspective: complex adaptations and the evolution of evolvability. *Evolution*, pages 967–976.

Humeau, J., Liefooghe, A., Talbi, E.-G., and Verel, S. (2013). ParadisEO-MO: from fitness landscape analysis to efficient local search algorithms. *J Heuristics*, 19(6):881–915.

Kallel, L., Naudts, B., and Rogers, A. (2013). *Theoretical aspects of evolutionary computing*. Springer Science & Business Media.

Kashtan, N., Noor, E., and Alon, U. (2007). Varying environments can speed up evolution. *PNAS*, 104(34):13711–13716.

Kauffman, S. and Levin, S. (1987). Towards a general theory of adaptive walks on rugged landscapes. *Theor. Popul. Biol.*, 128(1):11–45.

Khan, A. I., Dinh, D. M., Schneider, D., Lenski, R. E., and Cooper, T. F. (2011). Negative Epistasis Between Beneficial Mutations in an Evolving Bacterial Population. *Science*, 332(6034):1193–1196.

Lee, H., Popodi, E., Tang, H., and Foster, P. L. (2012). Rate and molecular spectrum of spontaneous mutations in the bacterium Escherichia coli as determined by whole-genome sequencing. *PNAS*, 109(41):E2774–E2783.

Lenski, R. E., Ofria, C., Pennock, R. T., and Adami, C. (2003). The evolutionary origin of complex features. *Nature*, 423(6936):139–44.

Lenski, R. E., Rose, M. R., Simpson, S. C., and Tadler, S. C. (1991). Long-term experimental evolution in Escherichia coli. I. Adaptation and divergence during 2,000 generations. *Am. Nat.*, pages 1315–1341.

Maley, C. C. (1994). The computational completeness of Ray's Tierran assembly language. In Langton, C. G., editor, *Artificial Life III*, volume 17, pages 503–514. Addison-Wesley, Redwood City, CA.

McClintock, B. (1993). *The significance of responses of the genome to challenge*. Singapore: World Scientific Pub. Co.

McKinley, P., Cheng, B., Ofria, C., Knoester, D., Beckmann, B., and Goldsby, H. (2008). Harnessing Digital Evolution. *Computer*, 41(1):54–63.

Merz, P. and Freisleben, B. (2000). Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *IEEE Transactions on Evolutionary Computation*, 4(4):337–352.

Meyer, J. R., Dobias, D. T., Weitz, J. S., Barrick, J. E., Quick, R. T., and Lenski, R. E. (2012). Repeatability and Contingency in the Evolution of a Key Innovation in Phage Lambda. *Science*, 335(6067):428–432.

Misevic, D., Ofria, C., and Lenski, R. E. (2006). Sexual Reproduction Reshapes the Genetic Architecture of Digital Organisms. *Proc. R. Soc. B*, 273(1585):457–464.

Misevic, D., Ofria, C., and Lenski, R. E. (2010). Experiments with Digital Organisms on the Origin and Maintenance of Sex in Changing Environments. *J. Hered.*, 101(suppl 1):S46–S54.

Nahum, J. R., Godfrey-Smith, P., Harding, B. N., Marcus, J. H., Carlson-Stevermer, J., and Kerr, B. (2015). A tortoisehare pattern seen in adapting structured and unstructured populations suggests a rugged fitness landscape in bacteria. *PNAS*, 112(24):7530–7535.

Ofria, C. and Adami, C. (2002). Evolution of genetic organization in digital organisms. In *Evolution as Computation*, pages 296–313. Springer.

Ofria, C., Adami, C., and Collier, T. C. (2002). Design of evolvable computer languages. *Evolutionary Computation, IEEE Transactions on*, 6(4):420–424.

Pigliucci, M. (2008). Is evolvability evolvable? *Nat Rev Genet*, 9(1):75–82.

Ryan, F. J. (1953). Evolution observed. *Sci. Am.*, 189:78–82.

Stearns, S. C., Ackermann, M., Doebeli, M., and Kaiser, M. (2000). Experimental evolution of aging, growth, and reproduction in fruitflies. *PNAS*, 97(7):3309–3313.

Szendro, I. G., Schenk, M. F., Franke, J., Krug, J., and Visser, J. A. G. M. d. (2013). Quantitative analyses of empirical fitness landscapes. *J. Stat. Mech.*, 2013(01):P01005.

Weinreich, D. M., Delaney, N. F., DePristo, M. A., and Hartl, D. L. (2006). Darwinian evolution can follow only very few mutational paths to fitter proteins. *Science*, 312(5770):111–114.

Wilke, C. O., Wang, J. L., Ofria, C., Lenski, R. E., and Adami, C. (2001). Evolution of digital organisms at high mutation rates leads to survival of the flattest. *Nature*, 412(6844):331–3.

Wright, S. (1931). Evolution in Mendelian Populations. *Genetics*, 16(2):97–159.

Wright, S. (1964). Stochastic processes in evolution. *Stochastic models in medicine and biology*, 25:199–241.

Zaman, L., Meyer, J. R., Devangam, S., Bryson, D. M., Lenski, R. E., and Ofria, C. (2014). Coevolution drives the emergence of complex traits and promotes evolvability. *PLoS Biol*, 12(12):e1002023.

The Relationship Between Evolvability and Robustness in the Evolution of Boolean Networks

David Shorten[1] and Geoff Nitschke[1]

[1]Department of Computer Science, University of Cape Town
dshorten@cs.uct.ac.za, gnitschke@cs.uct.ac.za

Abstract

Robustness and evolvability have traditionally been seen as conflicting properties of evolutionary systems, due to the fact that selection requires heritable variation on which to operate. Various recent studies have demonstrated that organisms evolving in environments fluctuating non-randomly become better at adapting to these fluctuations, that is, increase their *evolvability*. It has been suggested that this is due to the emergence of biases in the mutational neighborhoods of genotypes. This paper examines a potential consequence of these observations, that a large bias in certain areas of genotype space will lead to increased robustness in corresponding phenotypes. The evolution of boolean networks, which bear similarity to models of gene regulatory networks, is simulated in environments which fluctuate between task targets. It was found that an increase in evolvability is concomitant with the emergence of highly robust genotypes, where evolvability was defined as the population's *adaptability*. Analysis of the genotype space elucidated that evolution finds regions containing robust genotypes coding for one of the target phenotypes, where these regions overlap or are situated in close proximity. Results indicate that genotype space topology impacts the relationship between robustness and evolvability, where the separation of robust regions coding for the various targets was detrimental to evolvability.

Introduction

An open question in artificial and natural life is whether digital and natural organisms undergoing an evolutionary process are able to become better at adapting to the selective pressures presented to them, that is to become more *evolvable* (Wagner and Altenberg, 1996a). However, this question is complicated by multiple definitions of *evolvability*. In both evolutionary biology (Pigliucci, 2008, 2010; Wagner, 2008; Parter et al., 2008) and *Evolutionary Computation* (EC) (Tarapore and Mouret, 2014), for example, evolvability refers to either populations or individuals (Wilder and Stanley, 2015). Similarly, within EC (Eiben and Smith, 2003) numerous definitions and associated metrics have been proposed. For example, those that focus exclusively on solution fitness (Grefenstette, 1999; Reisinger and Miikku-

lainen, 2007) or variability of offspring (Lehman and Stanley, 2013). Tarapore and Mouret (2014) developed a metric which incorporated both the fitness and diversity of offspring. Reisinger et al. (2005) measured evolvability as the ability of genotypes with various representations to detect invariant patterns as commonalities in a changing fitness function.

Two significant definitions from the biological literature[1] which are pertinent to this work are those of *phenotypic accessibility* and *adaptability*. The former refers to the proportion of phenotypes that can be accessed through evolution (Wagner, 2008; Ciliberti et al., 2007a,b). This proportion is determined by the topology of the genotype space (that is, which other genotypes a given genotype can directly mutate to) and the $G \rightarrow P$ mapping (which phenotype each genotype codes for). Adaptability refers to the ability of organism to adapt to changes in the environment. It has the advantage of not imposing an experimenter-biased measure on the system, but merely asks whether evolution *delivers the goods* (Budd, 2006; Pigliucci, 2008). A prevailing hypothesis is that if the environment sufficiently varies over time, then organisms evolve the ability to be able to evolve suitable adaptations to such environmental changes faster (Wagner and Altenberg, 1996a; Wagner, 2008; Draghi and Wagner, 2008). Crombach and Hogeweg (2008) as well as Draghi and Wagner (2008) have demonstrated that computational models of *Gene Regulatory Networks* (GRNs) exhibit such *evolvability*.

The representation problem in EC addresses the issue of how to represent and adapt (mutate and recombine) genotypes in order that a broad range of complex solutions can be represented by relatively simple genotype encodings (Wagner and Altenberg, 1996b). The choice of representation and associated operators has a significant impact on the evolution of viable solutions and representations which facilitate evolution have been termed *evolvable* (Wagner and Altenberg, 1996b; Rothlauf, 2006; Simões et al., 2014). Similarly, in nature, genetic information defining the form and

---

[1]For a review of evolvability in biology, the reader is referred to Pigliucci (2008).

function of an organism is stored within its genotype, however, the developmental process which translates this information into phenotypes (the G→P map) is not well understood (Pigliucci, 2010). Yet, it has become clear that the G→P map is neither one-to-one nor linear (Gjuvsland et al., 2013). In many organisms and *Ribonucleic acid* (RNA) folding (Draper, 1992), it has been found that many genotypes can code for a single phenotype and that genetic change resulting from mutation is not proportional to phenotypic change (Pigliucci, 2010; Wagner, 2008; Parter et al., 2008).

These features of the G→P map have two important consequences on evolutionary dynamics. The first is that they allow for the emergence of robust genotypes. As many genotypes can code for one phenotype, it is possible that some number of a genotype's mutational neighbors code for the same phenotype as it does, thus affording the genotype a degree of mutational robustness (Wagner, 2008). The second consequence is that it becomes plausible that certain phenotypes are found in greater abundance in certain regions of the genotype space. This in turn results in genotypic mutations having non-random effects on phenotypes, opening up the possibility that the distribution of these effects is in some way advantageous (Hogeweg, 2012; Watson et al., 2014; Parter et al., 2008; Meyers et al., 2005; Gerhart and Kirschner, 2007; Pavlicev et al., 2010).

We hypothesize that both robustness and mutational bias facilitate evolvability, however, the complex relationship between robustness and evolvability makes this non-trivial to elucidate. Recent work has indicated that a certain degree of robustness has a large benefit on evolvability interpreted either as phenotypic accessibility (Wagner, 2008; Ciliberti et al., 2007a) or adaptation to a goal (Draghi et al., 2010). These studies are predicated on the assumption that a certain proportion of genotypes are non-viable and will never produce offspring. This constrains the portion of the genotype space that is accessible from a given genotype. That is, once robustness rises above a certain threshold, genotypes of a given phenotype get connected in large neutral networks which can be traversed in order to access a large variety of phenotypes and this access to variation allows for faster adaptation towards a stationary task target.

Mutational biases, however, can increase the likelihood of a given phenotype occurring in the neighborhood of a given genotype, thus facilitating adaptation towards it. Various recent studies have demonstrated that organisms evolving in environments fluctuating non-randomly are able to become better at adapting to these fluctuations, thus increasing their *evolvability* (Crombach and Hogeweg, 2008; Draghi and Wagner, 2008). It has been suggested that this is due to the emergence of biases in the mutational neighborhoods of genotypes (Hogeweg, 2012; Watson et al., 2014).

Hence, this study aims to examine a potential consequence of these biases, which is that increasing the bias

Parameter	Value Range
Weights (w_{ij})	$[-2, 2]$
Thresholds (θ_{ij})	$[-3, 3]$
Number of nodes	20
Incoming connections per node	$[1, 10]$
Simulation iterations (maximum t)	6

Table 1: Parameters for the networks composed of nodes with threshold functions

Parameter	Value Range
Number of nodes	20
Incoming connections per node	2
Simulation iterations (maximum t)	6

Table 2: Parameters for the networks composed of nodes with threshold functions

towards a small number of phenotypes within a region of the genotype space will increase the robustness of the genotypes within that region. Experiments were conducted using the simulated evolution of boolean networks in a fluctuating environment. These boolean networks were comprised of nodes containing either NAND logic gates or threshold functions. The networks composed of threshold functions closely resemble models of gene regulatory networks used to demonstrate the emergence of evolvability in varying environments (Crombach and Hogeweg, 2008; Draghi and Wagner, 2008). The networks composed of NAND logic gates were similar to those used to demonstrate the emergence of modularity (Kashtan and Alon, 2005) as well as task performance speedup (Kashtan et al., 2007) in fluctuating environments.

Results indicate that an increase in evolvability is concomitant with the emergence of highly robust genotypes. Analysis of the genotype space elucidated that evolution finds regions containing robust genotypes coding for one of the target phenotypes and that these regions are either overlapping or situated in close proximity. It was further found that the greater the separation between robust regions, the greater the drop in fitness during target changes. This implies that less separated robust regions allows for greater evolvability to emerge.

Methods

Network Models

Networks were formed of nodes which could hold an activation value of either zero or one. Each node had an activation function which was either a threshold or a NAND function. The activations of other connecting nodes were used as the inputs to these functions, the specification of which nodes'

Operator	Description
mutate_weight	A new value for the weight of an edge is chosen from the allowed values.
mutate_connection	An incoming connection to a node is given a different source.
mutate_threshold	A new value for the threshold of a node is chosen from the allowed values.
add_edge	A new incoming edge is added to the node. It connects to a random node and has a random weight.
delete_edge	One of the node's edges is chosen at random and removed.

Table 3: Mutation operators for the networks.

Parameter	Value
Population size	5000
Births per generation	5000
Tournament size	10
Recombination	None
Generations per task variant switch	10
Number of generations	500
Mutation	See table 3
NAND connection mutation rate	0.05
Threshold connection mutation rate	0.005
Node mutation rate	0.02

Table 4: Evolutionary Algorithm Parameters

	Target One	Target Two
Pair One	0000011001100000	0110111111110110
Pair Two	0000011001100000	0000111001100100

Table 5: Task target pairs used. The bit strings represent the desired outputs for each input permutation, where the permutations are ordered by their integer interpretation. Thus, for instance, the first bit of the string represents the desired output for the permutation 0000, the last bit for permutation 1111 and the fourth bit for permutation 0100.

activations to use thus implied the connectivity of the network. Updates were done synchronously. The threshold function used is specified in equation 1.

$$s_i(t+1) = \begin{cases} 1: & \sum_j w_{ij}s_j(t) > \theta_i \\ 0: & \sum_j w_{ij}s_j(t) \leq \theta_i \end{cases} \quad (1)$$

Where, $s_i(t)$ is the activation of the ith node at simulation iteration t, w_{ij} is the connection weight of the directed edge from the ith to the jth node, and θ_i is the threshold of the ith node. If no such connection exists then $w_{ij} = 0$. Table 2 presents the parameters of the NAND networks and table 1

Setup	Silhouette score
NAND, pair one	0.27 (0.08)
NAND, pair two	0.08 (0.05)
Threshold, pair one	0.06 (0.03)
Threshold, pair two	0.06 (0.05)

Table 6: Silhouette scores for generated genotypes in the neighborhood of the best genotype at then end of multiple runs for each setup. Standard deviations are in parentheses.

presents the parameters of the threshold networks.

Mutation Operators

Table 3 specifies the mutation operators used in the evolutionary algorithm applied to evolve the networks. The mutate_weight operators were applied to the GRN node connections, where for every connection, the operator was applied with a given probability (table 4). All other mutation operators acted on nodes' activation and threshold functions with a given probability (table 4). In the evolution of networks with NAND functions only the mutate_connection operator was used, whereas all the operators were used in the evolution of networks with threshold functions.

Evaluation

Networks were evolved to perform specific boolean functions with four inputs and one output. Each network was therefore run sixteen times, once for each of the possible input permutations. For each run, nodes, designated as the *input nodes*, had their activations set to the values of the given input permutation and their activations were clamped to these values for the duration of the run. The network was then simulated for the number of time steps specified in tables 1 and 2.

At the end of the simulation, the value of a designated *output node* was read and if it matched the output value of

the target function, for the given input, the fitness of the network was incremented by one. Networks could therefore have fitness values in the range $[0, 16]$.

Each evolutionary run had a pair of target functions. The target against which networks were evaluated alternated between the members of this pair. The target pairs are specified in table 5. Target one of pair one is the function (a XOR b) AND (c XOR d) and target two of pair one is (a XOR b) OR (c XOR d). This was done to maintain consistency with previous work on evolving boolean networks in fluctuating environments (Kashtan and Alon, 2005; Kashtan et al., 2007).

Target one of pair two is the same as in pair one, however the second target was chosen to differ randomly in two outputs, as can be seen in table 5. This choice was made so as to ascertain the effect of alternating between targets that are more similar. Moreover, these various experiment parameters were chosen because preliminary testing showed that they were conducive to the emergence of evolvability.

Evolutionary Algorithm

The networks were evolved with an *Evolutionary Algorithm* (EA) using tournament selection and elitist survivor selection (Eiben and Smith, 2003). Also, the EA used mutation only (there was no recombination operator). At each generation, 5000 tournaments of ten genotypes were created. The winner of each tournament went on to produce a single child. The fittest 5000 of the 10000 genotypes composed of children and current generation's population went on to form the subsequent generation's population. Table 4 presents the EA parameters. The choices of selection operators as well as algorithm parameters were made as preliminary experiments showed that they were conducive to the emergence of evolvability.

Experiments

For each of the two network types, evolution was run twenty times for 750 generations for each of two different task target pairs. Thus, four different evolutionary setups were each run twenty times. During evolution the goal was switched between the two targets of the pair every ten generations.

During each evolutionary run, at the end of each generation, if there was at least one genotype in the population with maximum possible fitness, then a test on the average robustness of such genotypes was run. That is, if at least one genotype had reached the current target, as specified in table 5, then such a test was run. This test consisted of randomly drawing 200 such genotypes from the population, allowing for a given genotype to be drawn multiple times, and then creating 15 mutated copies of this genotype, using the same EA mutation operator and parameters as specified in the Evolutionary Algorithm section. The average robustness of the maximum fitness genotypes was then computed

as the proportion of these mutated copies which were also of maximum fitness.

Results and Discussion

Figure 1 presents plots of the maximum fitness, average fitness and robustness of maximum fitness genotypes averaged over the 20 runs for each of the four setups. In early generations $[0, 150]$ the average and maximum fitness respond more slowly to changes in the task target as compared to later generations $[710, 750]$. A further observation is that the average robustness of the maximum fitness genotypes that are present in the population increases over evolutionary time.

A useful statistic in this context is the correlation between the average robustness achieved after evolving towards one goal for the allotted ten generations and how quickly the population is able to adapt back to that goal after it has spent the next ten generations evolving towards the other goal. That is, we need to test the hypothesis that robustness and evolvability are correlated. In order to measure rate of adaptation, the average fitness two generations after the change back to the original goal was recorded.

We measure average absolute fitness, rather than the rate of fitness change (relative fitness) as an indicator of a population's evolvability. We elected to use absolute fitness, since measuring relative fitness would unfairly benefit genotypes whose fitness dropped the most after a change in the task environment. That is, given two populations, the one with the highest fitness a given interval after a task change is concluded to be the most evolvable. Also, this interpretation holds in the case that the other population suffers a greater fitness decrease after the task change, which might subsequently lead it to having a faster fitness increase.

Thus, for each of the four setups, the Pearson correlation (Freedman et al., 2007) was applied between the average robustness of the maximum fitness genotypes and the average fitness early into the evolution back to the goal. Specifically, the correlation measure was applied between robustness at the end of each period (where genotypes evolve towards the first target of the pair), and the average fitness two generations into the subsequent period evolving back towards this target. Results deriving from periods where no genotypes in the population were of maximum fitness (at the end of a period of evolution towards the first target) were excluded from the correlation computation. That is, robustness could not be calculated in these instances. It was found that in all four setups, there was a positive correlation between the robustness achieved and evolvability ($p < 0.01$). This supports our hypothesis that robustness and evolvability, defined as adaptability, are concomitant phenomena.

A further pertinent question was the structure of this robustness across genotype space. It is plausible that evolution could have found an area of robust genotypes for both targets interspersed.

279
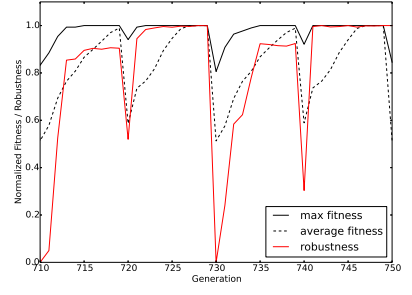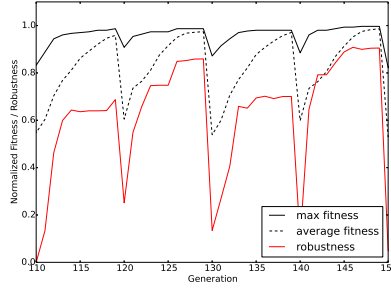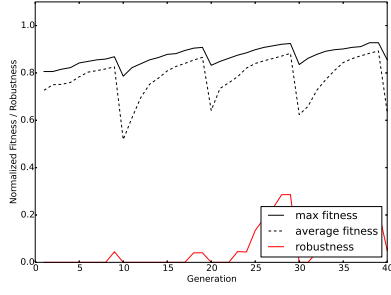
NAND Nodes, Target Pair One
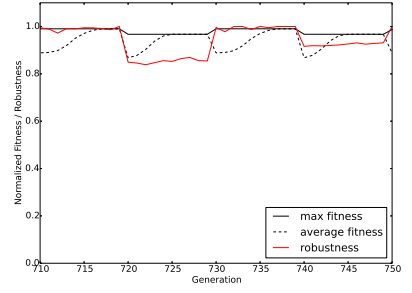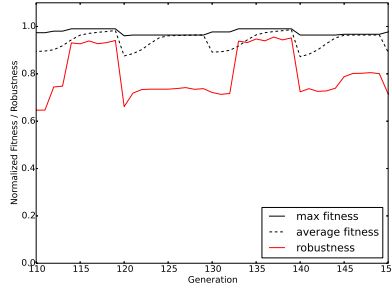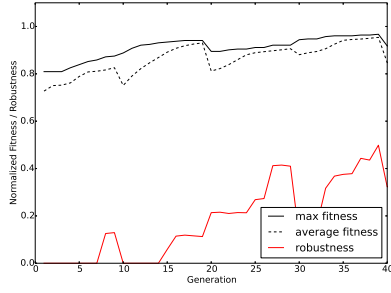
NAND Nodes, Target Pair Two
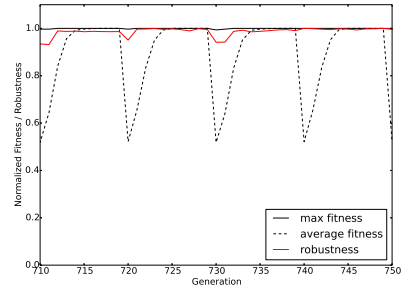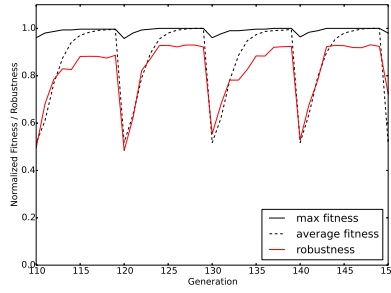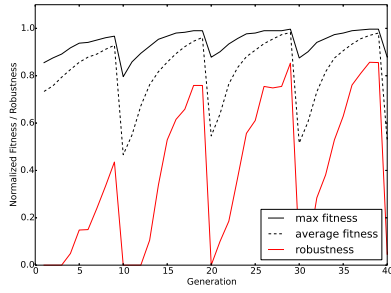
Threshold Nodes, Target Pair One
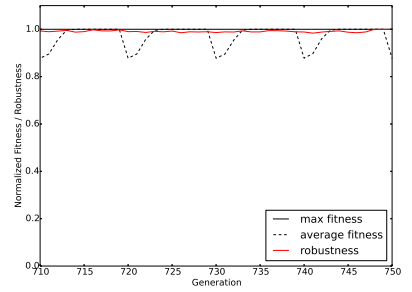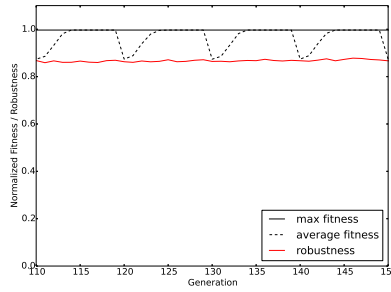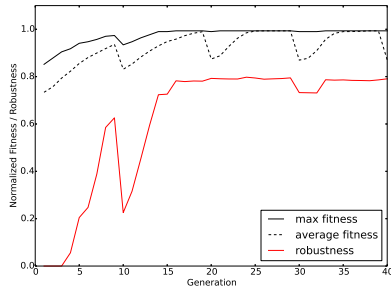
Threshold Nodes, Target Pair Two

Figure 1: Plots of the population average and maximum fitness as well as the average robustness of maximum fitness genotypes averaged over the 20 runs for each of the four setups. The left column contains plots from early generations [0, 40], the middle from slightly later generations [110, 150] and the right from generations near the end of the run [710, 750].

Alternatively it could be alternating between separate areas of robustness where these areas are adjacent or separated by a greater distance and connected by a strong fitness gradient. In order to ascertain this structure, visualizations of the genotype space were constructed using the t-SNE algorithm (Van der Maaten and Hinton, 2008).

For each of the four setups, evolution was run for 210 generations, where this number was chosen so that the population would be between robust regions (assuming their existence). At the end of these runs, mutated copies of the fittest genotype in the population were created using the same EA mutation operator and parameters as specified in the Evolutionary Algorithm section , although all mutation probability parameters were increased by a factor of two. This was because preliminary testing found smaller mutation probabilities caused the generation of sufficient target phenotypes to be too computationally expensive.

These mutated copies were created until there were 300 distinct genotypes which coded for each of the two target phenotypes. These genotypes were subsequently fed into the t-SNE algorithm which arranged them in a two-dimensional space to preserve distance in the higher dimensional (visualized) space. That is, genotypes which were closer together in the genotype space, formed clusters in the visualization.

The metric used in the t-SNE algorithm was the Hamming distance between genotypes. This is because two networks which differ only in the wiring of one connection should always be considered to be the same distance apart, regardless of the numerical identification values assigned to the nodes. A similar argument can be made for weights and thresholds.

The resultant visualizations are displayed in figure 2. When NAND networks were being used on target pair one there was a visible separation between the two robust regions. However, there was less separation in the other runs.

In order to gain a more quantitative understanding of this separation, the *silhouette score* (Rousseeuw, 1987) for each data set was computed. The silhouette score is a measure of the efficacy of a clustering algorithm. Scores are in the range $[-1, 1]$ and a positive score indicates distinct clusters, a negative score indicates data points being placed in the wrong cluster and a score of zero indicates overlapping clusters. Thus, in this instance, the silhouette score measured the efficacy of the clustering of the genotype space, where corresponding phenotypes were the labels used in the silhouette score calculation.

In order to facilitate statistical comparisons, this process was run 20 times. The results of these runs are displayed in table 6. The score for the NAND networks on target pair one is larger than for the other setups. Furthermore, this difference is statistically significant ($p < 0.01$, Mann-Whitney U test with Bonferroni correction (Flannery et al., 1986)).

The results indicate that an increase in evolvability, is concomitant with an increase in the robustness of genotypes coding for the target phenotypes. This can be observed in the plots of fitness and robustness displayed in figure 1, where rapid adaptation to new targets coincided with increased robustness. The link between evolvability and robustness is further supported by a statistically significant correlation between evolvability and robustness.

These results also support the hypothesis that increased evolvability is driven by biases in regions of genotype space (Hogeweg, 2012; Watson et al., 2014) and the inference that a strong bias towards certain genotypes will increase the robustness of these genotypes. Furthermore, finding a genotype within a given region will be aided by an abundance of this genotype. Thus, as it can be argued that evolvability implies robustness and robustness implies evolvability, the position of this paper is that the two phenomena are linked and occur in tandem.

Visualizations of the genotype space, as shown in figure 2, demonstrated that the nature of these biases can be nontrivial. That is, evolution does not necessarily settle on a region which is uniformly biased towards both of the target phenotypes. Although this case was observed, the corresponding case of adjacent regions, each biased to one of the phenotypes was also seen. The implications of these visualizations were supported by the significant difference in the silhouette scores between these cases.

Furthermore, the difference between these cases had a strong influence on the evolutionary dynamics, with a greater non-uniformity of the bias corresponding with larger drops in population fitness during task target changes.

It is theorized that the separation of the robust regions in the evolution of NAND networks on target pair one is due to the fact that, in that setup, phenotypes of either target are less likely to occur in close proximity. Nodes in the NAND networks had fewer possible connections than those in threshold networks (tables 1 and 2). Moreover, their thresholds and connection weights were not subject to mutation. This meant that the genotype space was of a much lower dimension, reducing the chance of two given phenotypes being nearby. Furthermore, the targets in pair one were more dissimilar. Should this hypothesis be correct, the implication is that the evolution of robust boolean networks is facilitated by a high dimension genotype space.

Moreover, these observations have led to further research questions that are the subject of current work. For example, confirmation of the above hypothesis, how large the distance between separated regions can be and whether analogous structures emerge when evolution fluctuates between larger numbers of task targets. We are also investigating the relationship between these results and improvements in adaptation, facilitated by robustness, increasing the accessibility of the genotype space (Draghi et al., 2010).

Conclusion

This study demonstrated that, in the simulation of evolving boolean networks in a fluctuating task environment, an
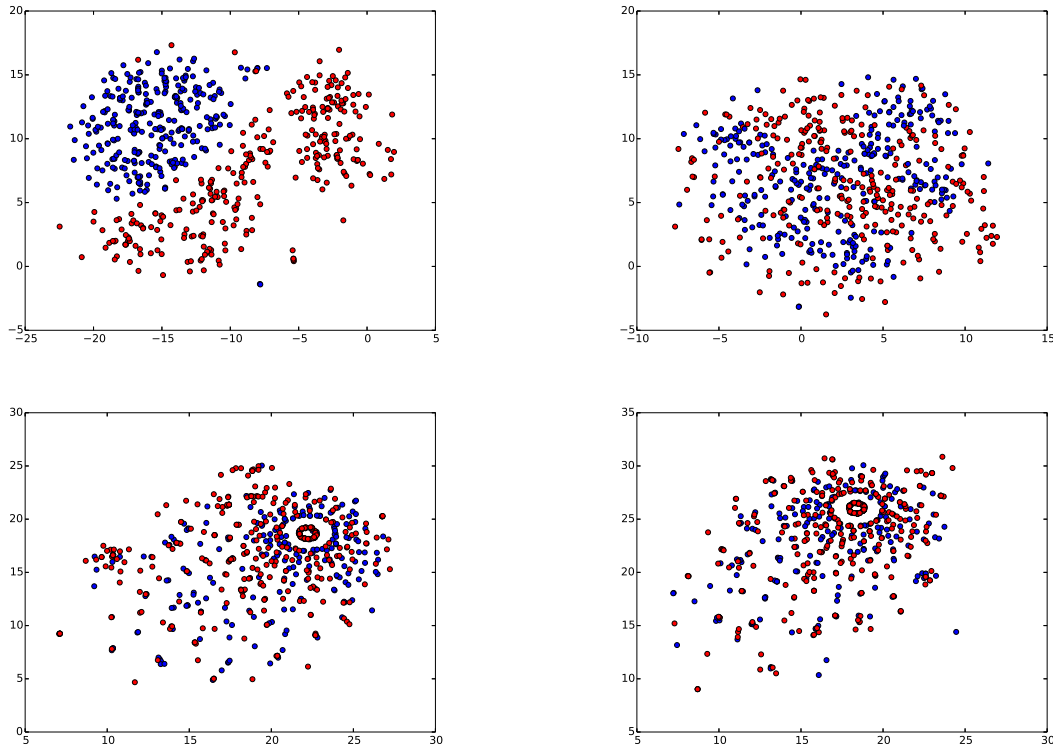
Figure 2: Visualizations of genotype space in the region of the best genotype at the end of an evolutionary run. Red dots are for genotypes that code for task target one and blue dots are for genotypes that code for target two. The positioning of the dots was determined using the t-SNE algorithm (Van der Maaten and Hinton, 2008) which aims to preserve the distance between the dots in the higher dimensional space. The left column contains visualizations based on target pair one and the right for target pair two. The top row contains visualizations for NAND networks and the bottom for threshold networks.

increase in *evolvability* was concomitant with an increase in the *robustness* of genotypes coding for the target phenotypes. These results support the hypothesis that evolvability, defined as adaptability, is driven by biases in the genotype space, however, visualizations of the genotype space showed unexpected structure in the nature of these biases. It was found that instead of biases towards different task targets occurring in the same region, in some instances they were separated into adjacent regions. These results contribute to a growing body of work (Wagner, 2008; Ciliberti et al., 2007a; Draghi et al., 2010) demonstrating the process by which robustness facilitates evolvability. However, elucidating the exact nature of the robustness, evolvability relationship remains the subject of ongoing research. Moreover, the impact of experiment parameters, notably the frequency of environment change, are yet to be determined.

References

Budd, G. E. (2006). On the origin and evolution of major morphological characters. *Biological Reviews*, 81(04):609–628.

Ciliberti, S., Martin, O., and Wagner, A. (2007a). Innovation and robustness in complex regulatory gene networks. *Proceedings of the National Academy of Sciences*, 104(34):13591–13596.

Ciliberti, S., Martin, O., and Wagner, A. (2007b). Robustness can evolve gradually in complex regulatory gene networks with varying topology. *PLoS Computational Biology*, 3:e15.

Crombach, A. and Hogeweg, P. (2008). Evolution of evolvability in gene regulatory networks. *PLoS Comput Biol*, 4(7):e1000112.

Draghi, J. and Wagner, G. P. (2008). Evolution of evolvability in a developmental model. *Evolution*, 62(2):301–315.

Draghi, J. A., Parsons, T. L., Wagner, G. P., and Plotkin, J. B. (2010). Mutational robustness can facilitate adaptation. *Nature*, 463(7279):353–355.

Draper, D. (1992). The rna-folding problem. *Acc. Chem. Res.*, 25(4):201–207.

Eiben, A. and Smith, J. (2003). *Introduction to Evolutionary Computing*. Springer, Berlin, Germany.

Flannery, B., Teukolsky, S., and Vetterling, W. (1986). *Numerical Recipes*. Cambridge University Press, Cambridge, UK.

Freedman, D., Pisani, R., and Purves, R. (2007). *Statistics*. International student edition. W.W. Norton & Company.

Gerhart, J. and Kirschner, M. (2007). The theory of facilitated variation. *Proceedings of the National Academy of Sciences*, 104(suppl 1):8582–8589.

Gjuvsland, A., Vik, J., Beard, D., Hunter, P., and Omholt, S. (2013). Bridging the genotype-phenotype gap: what does it take? *J Physiol.*, 591(8)::2055–2066.

Grefenstette, J. (1999). Evolvability in dynamic fitness landscapes: A genetic algorithm approach. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3. IEEE.

Hogeweg, P. (2012). Toward a theory of multilevel evolution: long-term information integration shapes the mutational landscape and enhances evolvability. In *Evolutionary Systems Biology*, pages 195–224. Springer.

Kashtan, N. and Alon, U. (2005). Spontaneous evolution of modularity and network motifs. *Proceedings of The National Academy of Sciences*, 102(39):13773–13778.

Kashtan, N., Noor, E., and Alon, U. (2007). Varying environments can speed up evolution. *Proceedings of the National Academy of Sciences*, 104(34):13711–13716.

Lehman, J. and Stanley, K. (2013). Evolvability is inevitable: Increasing evolvability without the pressure to adapt. *PloS one*, 8(4):e62186.

Meyers, L. A., Ancel, F. D., and Lachmann, M. (2005). Evolution of genetic potential. *PLoS Comput Biol*, 1(3):e32.

Parter, M., Kashtan, N., and Alon, U. (2008). Facilitated variation: how evolution learns from past environments to generalize to new environments. *PLoS Comput Biol*, 4(11):e1000206.

Pavlicev, M., Cheverud, J. M., and Wagner, G. P. (2010). Evolution of adaptive phenotypic variation patterns by direct selection for evolvability. *Proceedings of the Royal Society of London B: Biological Sciences*, page rspb20102113.

Pigliucci, M. (2008). Is evolvability evolvable? *Nature Reviews Genetics*, 9(1):75–82.

Pigliucci, M. (2010). Genotype phenotype mapping and the end of the genes as blueprint metaphor. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 365(1540):557–566.

Reisinger, J. and Miikkulainen, R. (2007). Acquiring evolvability through adaptive representations. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1045–1052. ACM.

Reisinger, J., Stanley, K., and Miikkulainen, R. (2005). Towards an empirical measure of evolvability. In *Proceedings of the 2005 workshops on Genetic and evolutionary computation*, pages 257–264. ACM.

Rothlauf, F. (2006). Introduction. In *Representations for Genetic and Evolutionary Algorithms*, pages 1–7. Springer Berlin Heidelberg.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

Simões, L. F., Izzo, D., Haasdijk, E., and Eiben, A. E. (2014). Self-adaptive genotype-phenotype maps: neural networks as a meta-representation. In *Parallel Problem Solving from Nature–PPSN XIII*, pages 110–119. Springer.

Tarapore, D. and Mouret, J.-B. (2014). Comparing the evolvability of generative encoding schemes. In *Proceedings of ALife*, pages 1–8. MIT Press.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85.

Wagner, A. (2008). Robustness and evolvability: a paradox resolved. *Proceedings of the Royal Society of London B: Biological Sciences*, 275(1630):91–100.

Wagner, G. and Altenberg, L. (1996a). Complex adaptations and the evolution of evolvability. *Evolution*, 50(1):967–976.

Wagner, G. and Altenberg, L. (1996b). Perspective: Complex adaptations and the evolution of evolvability. *Evolution*, pages 967–976.

Watson, R. A., Wagner, G. P., Pavlicev, M., Weinreich, D. M., and Mills, R. (2014). The evolution of phenotypic correlations and developmental memory. *Evolution*, 68(4):1124–1138.

Wilder, B. and Stanley, K. (2015). Reconciling explanations for the evolution of evolvability. *Adaptive Behavior*.

Evolvability of Minimally Cognitive Agents

Matthew Setzler[1,3], Eduardo Izquierdo[2,3]

Center for Research on Concepts and Cognition[1]
School of Informatics and Computing[2]
Cognitive Science Program, Indiana University, Bloomington[3]
mattsetz@gmail.com

Extended Abstract

This work investigates evolvability of continuous-time recurrent neural networks to support the behavior of model-agents subject to fitness criteria that changes over the evolutionary timescale. A population of agents is alternatingly evolved to perform two tasks with inverted fitness awards. Evidence of evolvability is reported; it is shown that the population locates a region of "meta-fitness" in the landscape in which sub-regions of optimality for each task are easily accessible from one another.

This work investigates evolvability of continuous-time recurrent neural networks (CTRNN) to support the behavior of model-agents subject to fitness criteria that changes over the evolutionary timescale. In this way, two broad thrusts of interest in recent computational approaches to artificial life and cognitive science are synthesized. On the one hand, evolvability is a hot topic of interest in and of itself – what kinds of developmental and phenotypic characteristics enable organisms to successfully adapt to changing environments? This question was raised by Richard Dawkins (1989), and has since received attention in paradigms ranging from binary circuits, discrete-time feed forward neural networks and toy developmental scenarios (Kasthan and Alon, 2005; Kovitz, 2015).

On the other hand there is work motivated by the joint perspectives of situated and embodied cognition, in which CTRNNs are embedded in agents with basic sensory-motor capacities. These agents are themselves embedded into dynamical environments, and genetic algorithms are used to evolve their nervous systems for *minimally cognitive behavior*, a term borrowed from Randall Beer to indicate "the simplest behavior that raises issues of genuine cognitive interest" (Beer, 1996). This methodology facilitates an investigation of cognition as it manifests in adaptive behavior occurring in dynamically coupled brain-body-environment systems. Furthermore, using stochastic search methods to configure nervous systems supporting cognitively interesting behaviors minimizes *a priori* assumptions about the kinds of cognitive and representational capacities an agent needs to support said behaviors.

More specifically, this work is inspired by two projects on either side of the aforementioned motivational fault line. Kashtan and Alon (2005) demonstrated that evolvability (via modularity) can be achieved in a connectionist network by alternatingly evolving it to perform two separate but related tasks over many epochs. Their work was conducted with feed forward binary networks evolved to perform discrete logical operations and association tasks. Motivated by the joint perspectives of embodied and situated cognition, we apply a similar methodology to a model-agent controlled by a CTRNN. Here the different tasks correspond to cognitively interesting behaviors carried out in a dynamical environment.

This evolvability study extends an object discrimination task first developed by Beer (1996), in which an agent with an array of distal sensors is required to distinguish circles from lines – moving towards the former and away from the latter in a simulated 2D environment. As in previous work (Beer, 1996), fitness is assigned on a scale of 0-100%, where 50% corresponds to a random solution and 100% is assigned to an optimal solution. Here, a population is evolved to perform this task (Task A) until the best individual reaches a fitness threshold of 80%, at which point the fitness criteria is reversed so that agents have to move towards lines and avoid circles (Task B); these fitness reversals will henceforth be called *swaps*. Note that Task A and Task B are mutually exclusive - an agent with high fitness for one of them has proportionately low fitness for the other. The population is evolved in this alternating fashion, which we refer to as *evolutionary swapping*, for 2500 generations.

To benchmark success of evolutionary swapping, both tasks were first evolved for in isolation. For each task, 20 evolutionary runs were performed over 500 generations with a population of 240. Across all 20 runs, the average value for the fitness of the best individual was 79.1% in Task A and 70% in Task B, suggesting that Task B is more difficult than Task A. More importantly, this indicates that the 80% threshold used in our swapping paradigm is non-trivial - in 500 generations only 8 of 20 runs in Task A produced agents exceeding this threshold, while no agents reached 80% fitness on Task B. In line with this, out of 20 evolutionary swapping runs, only 8 achieved one or more swaps. This is not so surprising, as the first swap corresponds to achieving 80% on Task A given a random starting population, which we know is not guaranteed to occur.

While there is not enough positive data to say that in general the swapping methodology makes it easier to reach the fitness threshold, particular swapping runs do show clear evidence of evolvability. Figure 1 shows the fitness trajectory for one such run. This trajectory is best characterized as periods of ascent up a fitness gradient up to a peak, at which point fitness suddenly

drops off. Peaks correspond to the fitness threshold (indicated by the dotted black line) being reached; the subsequent drop-offs are a result of task swapping - recall that fitness measures of each task are inversions of one another, so we should expect a population with high fitness with respect to one task to have a sudden drop in fitness when tasks are swapped. 12 swaps were achieved (marked by dotted red lines), the first 4 are labeled for the sake of discussion.

The horizontal distance between successive swaps corresponds to the number of generations taken to reach the fitness threshold for one of the tasks starting with the population of the previous swap (i.e. of the opposite task); this will henceforth be called the *swapping interval*. To the extent that evolvability is achieved in this paradigm, the swapping interval should be less than the number of generations it takes to reach 80% when either task is evolved for in isolation. We would also expect swapping intervals to generally decrease over the course of a run. Both of these properties are demonstrated in Figure 1. Starting from a random population, 80% fitness with respect to Task A is reached after 245 generations (*A1*). It then takes 1408 generations to achieve 80% fitness on Task B (*B1*), followed by a 198-generation swapping interval for the second evolution on Task A (*A2*), and then a 45-generation swapping interval for the second evolution on Task B (*B2*). The average swapping interval for the 8 remaining swaps is about 73 generations. That these short swapping intervals occur successively marks a huge improvement over evolving for either task in isolation, where for the most part the 80% threshold is not achievable in 500 generations.

How did such a dramatic increase in evolutionary efficiency occur? One possibility is that a highly diffuse population, which simultaneously contains high fitness individuals for both tasks is generated. This would result in small swapping intervals during which the population could remain relatively static. Alternatively, it could be that the population locates an area of "meta-fitness" in the landscape, in which sub-regions of optimality for each task are easily accessible from one another. This would allow for the population to cluster around a region of optimality for a given task and then quickly transition to a region of optimality for the other task in the following swapping interval. These two explanations are not mutually exclusive, but it is useful to differentiate between them, as one could be more at play than the other.

To help answer this question, principal component analysis (PCA) was performed on the 47-dimensional set of all genotypes in populations at *A1*, *B1*, *A2* and *B2*. Figure 2A shows these populations in a reduced 2D space. The first thing to notice is that populations at each swap are clustered around roughly distinct regions of parameter space (*A2* and *B2* appear to be overlaid but on closer examination it becomes evident that they are clustered in distinct regions, see Figure 2B). This provides support for the second explanation, that the population as a whole is moving through parameter space during swapping intervals, as opposed to a static and highly diffuse population. Furthermore, the proximity of the population clusters to one another sheds light on the swapping intervals displayed in Figure 1. The long duration of *A1-B1* corresponds to a large distance in parameter space separating *A1* and *B1*, while the relatively close proximity of *B1* to *A2* is consistent with the short evolutionary duration of *B1-A2* necessary to make such a transition in parameter space. Following in this manner, *A2* and

B2, which have the shortest reported swapping interval, also appear to be closest in parameter space.

Thus it appears that the search found a region of meta-fitness in the landscape, in which sub-regions of optimality for either task exist closely to one another. Presumably the close proximity between these sub-regions translates to important structural relations between agents with high fitness on either task. Identifying what these relations are, and investigating how they underpin the dynamical behaviors conducive to success in each task constitutes promising work for the future.
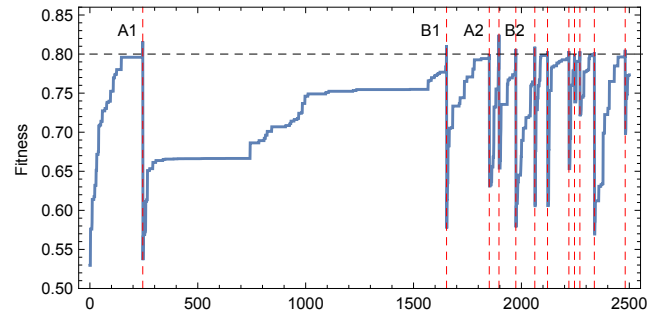
Figure 1: Evolutionary swapping run. Red-dotted vertical lines mark task swaps, the black-dotted horizontal line represents the 80% fitness threshold. The first 4 task swaps are successively labeled A1, B1, A2 and B2.
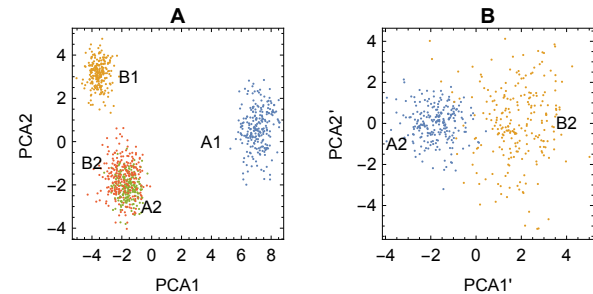
Figure 2: Populations at task swaps projected onto reduced 2D spaces. Figure 2A depicts populations of A1 (blue), B1 (yellow), A2 (green) and B2 (red) in a 2D space obtained by from a PCA on their union. Figure 2B depicts A2 (blue) and B2 (yellow) in a separate 2D space, obtained from a PCA on their union.

References

Beer, Randy D. (1996) "Toward the Evolution of Dynamical Neural Networks for Minimally Cognitive Behavior." From Animals to Animals 4: Proc. 4th Int. Conf. on Simulation of Adaptive Behavior: 421-29.

Dawkins, R. (1989). "The evolution of evolvability." Artificial Life, C. Langton (ed) Addison Wesley.

Kashtan, N., and U. Alon. (2005) "Spontaneous Evolution of Modularity and Network Motifs." Proceedings of the National Academy of Sciences 102.39: 13773-3778.

Kovitz, Ben. (2015) "Experiments with cascading design." EvoEvo workshop, 13th European Conference on ArtificialLife (ECAL).

Cooperation and Collective Behavior

Small-world property promotes the evolution of distributive altruism

Fuki Ueno[1,2] and Takaya Arita[1]

[1]Nagoya University, [2]Chukyo University
fukitosa@nagoya-u.jp

Introduction The question of how altruism evolves despite selfish behaviors benefitting agents has for many years been the central theme in the fields such as biology, ecology, sociology, economy and psychology. Several mechanisms promoting the evolution of cooperation have been proposed and can be broadly classified into two categories: direct fitness benefits or indirect fitness benefits (West, 2011). The former covers mutually beneficial cooperation, profiting both the actor and the recipient of the behavior (hence not truly altruistic). Altruistic cooperation is explained by the latter: by helping a close relative reproduce, an individual is still passing copies of its genes on to the next generation, albeit indirectly. Two mechanisms allow related individuals to interact: kin discrimination and limited dispersal.

The prisoner's dilemma game has been commonly used in computational studies on the emergence of altruism. Over the past two decades, many researchers investigated its behavior on networks. The two main properties of the network structure were found to favor survival of cooperators. The first one is clustering coefficient. The success of cooperative behavior is maintained by local interactions within a spatial structure, because cooperators can survive and grow only if they form clusters (Nowak and May, 1992). However, an inverse relationship between the formation of clusters and the success of cooperation has also been reported (Hauert and Szabo, 2005). The second network property is degree heterogeneity. Cooperation has been shown to emerge around the largest hub (Pacheco and Santos, 2005).

While it seems that the riddle of cooperation has been largely elucidated as above, recent empirical research brings a more refined view of cooperation as not just a single, homogeneous trait but several different traits with different costs, benefits and contexts. Warneken and Tomasello (2009) investigated three types of altruism by comparing behaviors of children and chimpanzees. The three types were: helping (when agents help others achieve their goals), sharing (when agents share valuable goods such as food with others) and informing (when agents share with others things the others need or want to know). They found that although chimpanzees help others instrumentally, they are less likely to share resources at their own expense. Also, they do not share information helpful to others. However, both infants and young children were observed to be helpful, generous, and informative. Thus, the authors suggest that sharing and informing are types of altruism specific to humans.

Based on the above, we have proposed the **distribution dilemma game** (DDG) (Ueno and Arita, 2015) that aims to model the altruism in the distribution of resources (material goods or information), but can also capture the emergent properties of resources. Specifically, DDG can describe both how the total value of a resource is changed when it is distributed among agents and how the value is changed synergistically when an agent owns different kinds of resources. Our preliminary study investigated the behavior of an evolutionary model with DDG on a one-dimensional torus and observed the emergence of altruism in certain scenarios. In this paper, we extend the study by investigating the effects of more realistic interaction networks on the emergence of altruism. Specifically, we focus on the behavior of DDG on small-world network topology.

The model Agents are on a network, which is generated using the Watts-Strogatz model. DGG is composed of a repetition of a resource distribution step and a strategy imitation step. In the distribution step, each agent distributes a unit of its unique kind of resource equally among its neighbors and itself. Each agent has its strategy represented by an integer value S: the number of nearest neighbors to whom it distributes resource (0 means the agent is selfish). The initial values are randomly set 0 or 1. If S is larger than the number of direct neighbors, recipients are selected from the neighbors of neighbors (and further if necessary).

In the imitation step, it will take over the strategy (S) of the neighbor who obtained the highest gain in the last distribution step. Mutation changes each S by 1 or -1 with a probability of 0.01 during imitation.

The gain of an agent (G_i), i.e. the resultant value of resource each agent owns at the end of a round is calculated using the following equations, with D and K being model parameters. D and K express the extent to which the overall value of one's resources is affected by the act of resource distribution.

$$G_i = \left(\sum_j F_{ij} \right) * (1 + K * V_i) \ , \tag{1}$$

$$F_{ij} = \left(\frac{1}{1+S} \right)^D \ , \tag{2}$$

$$V_i = -\sum_{j=1}^n P(q_{ij}) \log_2 P(q_{ij}) \ , \tag{3}$$

$$q_{ij} = \frac{F_{ij}}{\sum F_{ij}} \ . \tag{4}$$

Distributing property D (see Eq. 2) determines the type of resources being shared. $D = 0$ expresses that the resource is purely informational (each receiver obtains the entire copy),

Figure 2: (a) Small-world-ness coefficient as a function of rewiring probability p in the Watts-Strogatz model. Also shown: clustering coefficient and average path-length normalized by their values for regular lattice ($p = 0$). (b) A correlation of S and small-world-ness of $K = 0$, $D = 0.2$.
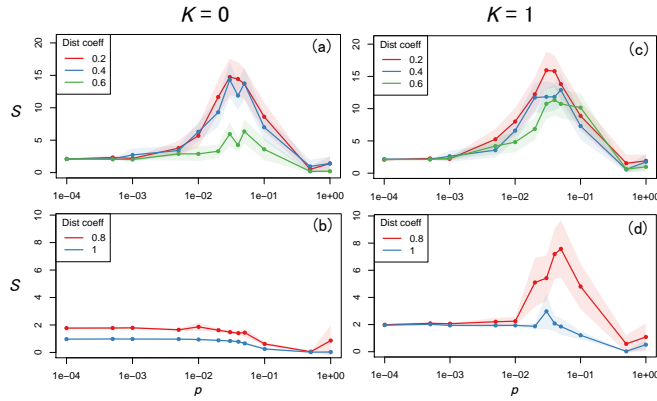
Figure 1: The average number of recipients to whom agents distribute resources (corresponding to their degree of altruism) as a function of W-S network rewiring probability p and for different values of the distributing property coefficient D. The left (a, b) and right (c, d) graphs show the cases with $K = 0$ and $K = 1$, respectively. The upper (a, c) and lower (b, d) graphs show the cases with $D = 0.2, 0.4, 0.6$, and with $D = 0.8, 1.0$, respectively.

whereas $D = 1$ expresses the resource is purely physical (each receiver obtains an equally divided). Furthermore, considering that the resource value can be changed not only by distribution cost but also depending on the number of the people who will share the resource in general, we treat various types of resources in the unified fashion by assuming D is a real number.

Gathering property K allows us to model the synergistic effect of owing different goods by using the idea of entropy. For example, a certain set of knowledge combined together may lead to the creation of a new idea. For values of K larger than 0, the gain coming from received goods is increased beyond the value of its sum.

Results   We investigated the behavior of the system by varying the distributing property of resource D between 0 (information) and 1 (material goods) and the gathering property K being either 0 or 1, where $K = 1$ corresponds to a maximum synergistic effect of gathering different resources. The network has $N = 65$ as the population size, an initial node degree 4 and a rewiring probability p ($p = 0$ is the regular network and $p = 1$ is random network). Each game lasted 100000 generations and was replayed 100 times using a different random seed.

For $K = 0$ and resources close to material goods ($D = 0.8$, 1) only a very limited propensity to share emerged only with low values of p ($p < 0.1$, Fig. 1b), in other words, with high spatial locality. For resources having a property closer to the information side ($D = 0.6, 0.4, 0.2$), the most altruistic agents emerged for intermediate values of p. Interestingly, we found that this range of p values corresponds to the networks having the highest small-world-ness coefficient (cf. Fig. 2a), measured using the method of Humphries and Gurney (2008).

In the scenario with synergistic resource effects (gathering property coefficient $K = 1$), the results for resources having a property closer to the information side were similar (Fig. 1c). The agents, however, became on average even more altruistic. The situation was very different for resource closer to material goods. The truly altruistic agents now also emerged in networks having the highest small-world-ness coefficient (Fig. 1d).
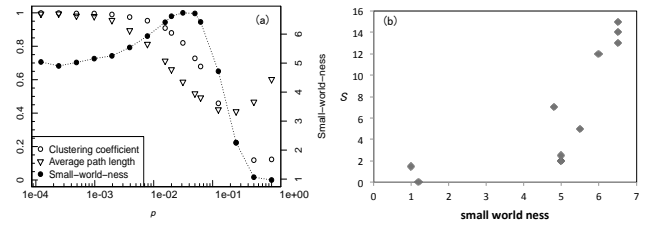
We suspect that the small levels of altruism emerging in the scenario shown in Fig. 1b can be explained by the mechanism of spatial locality: on regulator networks, close cooperators survive by benefitting each other (cooperative clustering). In the small world networks, however, we think altruism emerges owing to the degree heterogeneity (Pacheco and Santos, 2005). More precisely, the agent at the center of a hub has a high number of partners for interactions, therefore, it can get higher gain than the agent with the lower number of neighbors (assortative interactions). This causes agents around hubs to imitate altruistic strategy of the hub's central node, and potentially spreading the strategy further.

Conclusion   The correlation between small-world-ness and emergence of altruistic behavior observed in the DDG model, suggests that small-world networks may be essential for the emergence of altruism. Furthermore, we found that the gathering property of resource further strengthens the propensity of agents to behave in an altruistic way, which would otherwise not happen for resources that resemble material goods. We believe that DDG model sheds new light on the importance of social network structure for the emergence of cooperation as well as the powerful effect of synergistic resource effects.

References

Hauert, C. and Szabó, G. (2005). Game theory and physics. *American Journal of Physics*, 73: 405.

Humphries, M. D. and Gurney, K. (2008). Network 'small-world-ness': a quantitative method for determining canonical network equivalence. *PLoS ONE*, 3(4): 1–10.

Nowak, M. A. (2006). Five rules for the evolution of cooperation. *Science*, 314: 1560-1563.

Nowak, M. A. and May, R. M. (1992). Evolutionary games and spatial chaos. *Nature*, 359: 826–829.

Pacheco, J. M., Santos, F. C. (2005). Network dependence of the dilemmas of cooperation. In: Mendes, J. F. F. editor, *Science of Complex Networks: From Biology to the Internet and WWW, AIP Conf.,* 776: 90–100, AIP, Melville, NY.

Ueno, F. and Arita, T. (2015). The role of the emergent property of resource in the evolution of distributive altruism based on spatial reciprocity. *Artificial Life and Robotics*, 20(1): 78-85.

Warneken, F. and Tomasello, M. (2009). Varieties of altruism in children and chimpanzees. *Trends in Cognitive Sciences*, 13(9): 397-402

West, S. A. (2011). Sixteen common misconceptions about the evolution of cooperation in humans. *Evolution and Human Behavior*, 32: 231-262.

Social Learning, Environmental Adversity and the Evolution of Cooperation

Peter Andras[1]

[1]School of Computing and Mathematics
Keele University
Newcastle-under-Lyme, Staffordshire, ST5 5BG, UK
p.andras@keele.ac.uk

Abstract

Social learning plays a key role in the evolution of cooperation in humans and other animals. It has also been shown both theoretically and experimentally that environmental adversity is also a key determinant of the evolution of cooperation among individuals. Here we investigate the impact of social learning on the evolution of cooperation in the context of a range of levels of environmental adversity. We used an agent-based simulated world of asexual individuals that communicate and play a probabilistic version of the Prisoner's Dilemma game. We considered simulated worlds either with or without random spreading of the offspring and two variants of social learning, either copying to some extent all communication rules or copying fully some of the communication rules of the best performing neighbor individual. The results show that in the case of spreading of the offspring, social learning increases the level of cooperation and reverses the association between this and the level of environmental adversity, i.e. low adversity with social learning implies higher level of cooperation. Copying fully some communication rules also increases the steady-state level of communication complexity in the simulated agent communities. The results suggest that the level of cooperation in communities of individuals may get boosted alternately by highly adverse environments and by layers of social learning in low adversity environments.

Introduction

The emergence and evolution of cooperation among individual humans and animals is a fundamental question of social evolution (Axelrod, 1997). In general it is assumed that cooperation emerges either because of kin selection, or direct or indirect reciprocation, or because of some form of social clustering or due to the group level selection of groups with more cooperating individuals (Rand and Nowak, 2013). It has been also shown that social factors, such as enforcement of rule following, also contribute significantly to the evolution of cooperation (Sigmund et al, 2010).

An external factor that has critical influence on the level of cooperation is environmental adversity, which includes both the harshness of the environment (i.e. scarcity of resources) and the variability of the environment (i.e. the variability and extent of the lack of predictability of the level of available resources) (Andras et al, 2007; Andras et al. 2003). Theoretical, simulation and real world experimental results confirm that in general higher level of environmental adversity implies higher level of cooperation in communities

of individuals existing in the presence of such environmental constraints (Krams et al, 2010; Spinks et al, 2000; Rand et al, 2014; Potts and Faith, 2015).

Social learning in general means that individuals within a community adapt their behavior such that they follow behavioral patterns of other individuals (Flinn, 1997). The other individual may be chosen on various grounds, for example it can be the most successful individual in some pre-defined sense or it can be the oldest neighboring individual. It has been suggested that social learning supports cooperation in communities of individuals, especially in the context of humans (Boyd and Richerson, 2009), however, there are also claims of the opposite effect in the relevant literature (Heyes, 2013).

The level of cooperation can be measured directly in experiments (both in the case of agent-based simulations and in the real world). Having additional measures of correlates of cooperation is useful to understand better the context of the measured level of cooperation. One such measure is the communication complexity of the interactions between the individuals of the community (Andras, 2008). Communication complexity decreases with increased environmental adversity and this contributes to the increase in the level of cooperation (Andras, 2008).

Here we present the use of an agent-based simulation which implements communicating agents that play prisoner's dilemma games (Axelrod, 1997) to investigate the role of social learning in the context of evolution of cooperation in communities of selfish individuals. We considered two variants of the simulated world, one where the offspring of the asexual individuals do not get dispersed widely from the location of their parent, and another where the offspring is dispersed away from the location of the parent – in general the simulation without dispersal of the offspring have higher levels of cooperation at the steady-state level than the simulation with offspring dispersal (Andras et al, 2003; Andras et al, 2006). We also considered two variants of social learning, in one case the learner copies to some extent the communication rules of the most successful neighbor, in the other case the learner copies fully some of the communication rules of the most successful neighbor. The results show that copying fully some communication rules increases the level of cooperation more than the considered alternative social learning method. Both social learning methods have much more effect if the offspring are widely dispersed. Social

learning with full copying of some communication rules also reverses the association between environmental adversity and the level of cooperation, making the level of cooperation increase with the decrease of the environmental adversity. In addition, social learning with full copying of some communication rules leads to higher communication complexity in the simulated agent communities than the use of the other social learning method or simulations without any social learning. The results suggest that the level of cooperation in communities of individuals may get boosted alternately by highly adverse environments and by layers of social learning in low adversity environments.

The rest of the paper is structured as follows. First we review briefly the relevant results from the literature. Next we provide a description of the agent-based simulation that we used. This is followed by the presentation of the detailed results. Finally the paper is closed by the discussion and conclusions section.

Background

There are several theories about the mechanisms behind the emergence and evolution of cooperation in communities of selfish individuals. Kin selection assumes that related individuals recognize each other on the basis of their similarity and the likelihood of their cooperation with their kin is high in order to support the success and spreading of the kin (Rand and Nowak, 2013). Direct reciprocity assumes that individuals are likely to reciprocate the cooperative help received from others and expect further reciprocation of cooperation by others who benefit from this (Rand and Nowak, 2013). Indirect reciprocity relies on the assumption that individuals observe the behavior of other individuals and they are more likely to cooperate with those who are seen to cooperate with others (Rand and Nowak, 2013). Group selection based mechanisms assume that individuals belonging to groups characterized by higher level of cooperation are more likely to survive and have offspring because their group has a better chance of survival as a group due to the benefits from the high level of cooperation within the group (Rand and Nowak, 2013). Other models rely on emergent population structure (e.g. spatial constraints) that drive cooperators together and exclude non-cooperators, giving in such way an advantage to the emergent communities of cooperators over other emergent communities not dominated by cooperators (Rand and Nowak, 2013).

Environmental adversity is an important determinant of the emergence and evolution of cooperation (Andras et al, 2007; Andras et al, 2003). Theoretical analysis shows that higher environmental adversity (harsher environment or more variable environment) implies higher level of cooperation among individuals in communities that survive in high adversity environment (Andras et al, 2007; Andras et al, 2003). Experimental results about a range of animals, humans and agent-based simulation results confirm this theoretical result, showing that indeed, exposure to higher predation risk or higher variability of environmental resources or risks lead to higher frequency of cooperative behavior between individuals (Krams et al, 2010; Spinks et al, 2000; Rand et al, 2014).

Theoretical and agent-based simulation analysis of the environmental risk conceptualized as the variance of the available resources shows that the experienced subjective risk is always bigger than the objective risk and the difference is bigger in harsher environments (Andras et al, 2007). It has been also shown that the effective risk after taking into account the effect of cooperation is always smaller than the subjective risk and that the effective risk is practically stable across a range of subjective risk levels and this stable effective risk is slightly increasing with the harshness of the environment (Andras et al, 2006). This implies that higher subjective risk perceived by individuals triggers more cooperation in order to bring down the level of the effective risk to the stable level of this.

In the context of communicating individuals who negotiate before making the decision about cooperation the complexity of the communication language that they use contributes to the overall environmental risk. It has been shown through agent-based simulation studies that indeed communication complexity is lower in the case of higher external environmental risk (Andras, 2008). Note that the language complexity is measured in terms of the variability of the communication rules and not as the length of communication sequences preceding the decision on cooperation. This result implies that the communication complexity measure is a useful correlate of the extent of reduction of the effective risk through reduction of the unreliability of communications between individuals.

Social learning plays a key role in organizing the social role of individuals in the context of their social environment provided by their community (Flinn, 1997). The essence of social learning is the copying or imitating the behavior of one individual by another individual. There are several mechanisms of social learning, some being context-dependent others being content-dependent, some are oriented towards specific individuals (e.g. richest, most successful, oldest, most similar) others are driven by frequency of behaviors (e.g. most frequent is copied) or by the state of individuals (e.g. experiencing high dissatisfaction) (Rendell et al, 2010). Social learning may work by copying a fully or partially a whole sequence of consecutive behaviors or by aiming to emulate the outcome of a sequence of behaviors, or by some intermediate variant of behavioral copying (Rendell et al, 2010). Social learning may also be supported by enforcement of rules in various forms of punishment applied to individuals who do not conform to the rules (Sigmund et al, 2010).

Agent-based simulations have been used to study various aspects of social learning (e.g. choice of social learning mechanisms) (Nakahashi et al, 2012; Seltzer and Smirnov, 2015; Molleman et al, 2013). Such simulations usually implement a small range of alternative social learning mechanisms and analyze their impact on the behavior of the simulated agent community.

The role of social learning in the context of emergence and evolution of cooperation has been considered in a number of settings. In general it is suggested that social learning is a key contributor to the evolution of cooperation among humans and possibly also among other animals (Boyd and Richerson, 2009; Rendell et al, 2010; Chudek et al, 2013) It has been shown that in simulated social networks imitation of socially distant individuals increase the level of cooperation within the

agent community (Seltzer and Smirnov, 2015). Other agent-based simulation studies show that certain forms of social learning (e.g. conformism) reduces the level of cooperation in simulated communities (Molleman et al, 2013; Burton et al, 2015). There are also more theoretical / conceptual investigations that question the level of contribution of social learning to the emergence and evolution of cooperation among humans (Heyes, 2013).

Simulated Agent Communities

The simulated world of agents is placed in a two dimensional space arranged as a torus in both dimensions and having the size of 1000 in both dimensions. The agents move randomly in this space in each turn (up to 5 units in both dimensions).

Each simulation runs for 400 time turns. In each turn each agent picks randomly another agent from its spatial neighborhood to interact with. An agent is allowed to interact with only one other agent at any time and some agent may stay without interaction partner in some of the time turns.

The agents own resources and they spend these to survive. If the resource amount of an agent goes below zero the agent dies. The agents use their current level of resources to set their level of resources in the next turn. They may also play a resource generation game with their interaction partner.

The agents interact using a communication language consisting of the symbols: '0','s','i','y','n','h' and 't'. The meaning of the communication symbols are as follows: '0' – no intention of communication, 's' – start of communication, 'i' – maintaining the communication, 'y' – indication of the willingness to engage into resource sharing, 'n' – indication of no further interest in communication, 'h' – effective sharing of the resources, 't' – not sharing the resources after an indication of willingness to engage into sharing. The last two symbols, 'h' and 't' effectively mean the resource-sharing or no-resource-sharing actions of the agents. The generation of communication symbols by agents is determined by probabilistic communication rules of the agents. These rules are expressed as follows

$$L: \qquad (1)$$
$$U_{current}, U'_{current} \rightarrow_{p1} U_{new,1}; \rightarrow_{p2} U_{new,2}; ...; \rightarrow_{pk} U_{new,k}$$

where $U_{current}$ is the current communication symbol produced by the agent, $U'_{current}$ is the current communication symbol produced by the communication partner agent, $U_{new,j}$ is the j-th possible communication symbol that may be produced by the agent following the previous production of the symbol $U_{current}$ and the production of the symbol $U'_{current}$ by the communication partner agent, and p_i is the probability of producing $U_{new,i}$ the symbol. Naturally we have that $p_1 + p_2 + ... + p_k = 1$. For example, a communication rule can be the following

$$L: \qquad (2)$$
$$i, i' \rightarrow_{0.5} i; \rightarrow_{0.2} y; \rightarrow_{0.3} n$$

which means that after producing the symbol 'i' and receiving the symbol 'i' from the communication partner, with 0.5 probability the agent will produce the symbol 'i', with 0.2 probability the symbol 'y' and with 0.3 probability the symbol 'n'.

An example of a sequence of communications between two agents is: 's_1, s_2, i_1, i_2, i_1, i_2, y_1, i_2, y_1, n_2', where the indices are the identifiers of the two agents. If the communication process between two agents carries on for too long without reaching the production of the action symbols 't' or 'h' (the length limit was set to 20 symbols), the communication terminates as it is considered too long for the time turn. The communication between two agents may also terminate if either of them starts by producing the '0' symbol, if one of them produces the 'n' symbol, or if they both produce a 't' or 'h' symbol. In the latter case the agents engage in a prisoners' dilemma game where the outcome of the game depends on the actions of the involved agents, i.e. they cooperate if both of them produce the symbol 'h' otherwise one of them or both of them tries to cheat (by producing the symbol 't').

When the agents enter the playing of the prisoners' dilemma game they jointly invest their available resources to generate new resources. The overall payoff of the game is the difference between the sum of the amounts of new resources that each agent would have without entering the game and the amount of resources that can be generated by using the combined current resources of the agents. If an agent cheats while the interaction partner is willing to cooperate the cheating agent takes the full payoff and the other agent gets no extra resources in addition to what it can generate by itself with its own available resources. If they both decide to cooperate they share the full payoff equally and this gets added to the amount of resources that they would generate individually. If both agents decide to try to cheat no extra resource is allocated to either of the agents.

The generation of effective new resources is realized in a probabilistic manner. The actual value is picked from a uniform distribution where the mean value of the distribution is given by the calculated value of new resources and the half-width (equivalent of variance) of the distribution is given by the environmental risk level (σ) that characterizes the simulated world of the agents. Low environmental risk (low variance) means than the actual value of the new resource is close to the calculated mean value of the resource value distribution, while high environmental risk (high variance) means that the actual value may differ significantly from the calculated mean value (can be also much smaller and much larger).

The agents have a memory of their most recent interactions with other agents (last ten interacting agents). The memories record the outcome of the interactions with these other agents and depending on the experience of the agent the probability of the resource sharing action of the agent is altered – it gets more likely to cooperate again with interaction partners who cooperated previously and less likely with those who cheated previously (i.e. the probabilities of the rule components $y, y' \rightarrow_p t$ and $y, y' \rightarrow_q h$ change – e.g. the latter gets bigger if the sharing gets more likely according to the past experience).

The agents engage in social learning. They select the individual with the highest amount of resources in their neighborhood as target of imitation – the neighborhood consists of the 10 closest other agents. Two kinds of social learning approaches have been implemented. In one case the agents copy to some extent the communication behavior of the imitated agent by setting their communication rule

probabilities similar to the matching probabilities of the imitated agent. This is implemented as

$$p_{revised}(U_{current}, U'_{current}, U_{new}) = (1\text{-}\eta) \cdot p_{original} \qquad (3)$$
$$(U_{current}, U'_{current}, U_{new}) + \eta \cdot p_{imitated}$$
$$(U_{current}, U'_{current}, U_{new})$$

where $p(U_{current}, U'_{current}, U_{new})$ is the probability of generating the symbol U_{new} by the agent after previously having generated the symbol $U_{current}$ and having received the symbol $U'_{current}$ from the communication partner, and η is the extent of the fidelity of the imitation. In the second social learning approach the agent copies fully some of the communication behaviors of the imitated agent. In this case η, the extent of the fidelity of the imitation, is the probability of copying for all communication rules L (i.e. includes the copying of all related probabilities).

The agents have a limited life span (60 time turns at most in the simulations that are reported here – the agent start their life at a randomly set starting age that is at most 20). When they reach the end of their life they reproduce asexually, by generating potentially mutated offspring which inherit the communication rules with possible small changes to the relevant probabilities. The number of offspring depends on the resources available to the agent (ρ) at the time of death and it is determined by the equation

$$n = [\beta \cdot ((\rho - \rho_{mean}) / \rho_{stdev}) + \gamma] \qquad (4)$$

where ρ_{mean} and ρ_{stdev} are the mean and standard deviation of the resource across the whole agent community at the time when the offspring is generated and β and γ are parameters, $[.]$ is the integer part function ($\beta = 1.5$, $\gamma = 1.5$). We also capped the number of offspring, i.e. if $n > n_{max}$ then the number of offspring is n_{max} ($n_{max} = 15$). If the above calculation gives $n < 1$ then the agent has no offspring.

The offspring of the agent may be spread closely around the location of their parent or may get widely dispersed in the full extent of the two dimensional world in which the agents exist. The first offspring location option may create clumps of cooperating agents, while the second option prevents this. We implemented both options of placing of the offspring of dying agents.

More details about the simulated agent world described above can be found in Andras et al (2003), Andras et al (2006) and Andras (2008). The code developed in Delphi for the implementation of the simulated agent worlds is available on request from the author.

Results and Analysis

We considered the following six simulation scenarios: (I) partial copying of all rules without wide dispersion of the offspring; (II) partial copying of all rules with wide dispersion of the offspring; (III) full copying of some rules without wide dispersion of the offspring; (IV) full copying of some rules with wide dispersion of the offspring; (V) no social learning and without wide dispersion of offspring; (VI) no social learning and with wide dispersion of offspring. For all scenarios with social learning we considered two variants with low and high levels of copying (η), i.e., $\eta = 0.2$ and $\eta = 0.8$. We ran 20 simulations for five levels of environmental risk ($\sigma = 0.1, 0.3, 0.5, 0.7, 0.9$) for each variant of the simulation

Figure 1. The evolution of the level of cooperation and the level of language complexity for agent communities without wide dispersion of offspring and social learning by partial copying of all language rules (scenario I): A) level of cooperation with $\eta = 0.2$; B) level of cooperation with $\eta = 0.8$; C) level of language complexity with $\eta = 0.2$; D) level of language complexity with $\eta = 0.8$; where η is the level of copying.

scenarios. All data shown in the figures are averages of 20 simulation runs, the standard deviations are small and are not shown to not clutter the figures.

In the case of scenarios without wide dispersion of offspring the starting size of the agent population is 1,800, while in the case of scenarios with wide dispersion of offspring the populations have 7,500 individuals at start. In scenarios with wide dispersal of the offspring the likelihood that an agent dies without offspring is higher than in scenarios with closely located offspring. Thus in scenarios with wide dispersal of the offspring the likelihood that a smaller agent population goes extinct is relatively high. For this reason the population size was increased in these scenarios. Simulations with larger population sizes take more time to run but do not influence the nature of the results presented here.

We measured the level of cooperation (c) by calculating the percentage of agents that engage in a cooperation interaction (i.e. both agents communicate the symbol 'h' at the end of their interaction) among all agents in the current agent population.

We also measured the complexity of the agent language. For this purpose we considered all language rules L_r, $r = 1,...,R$ (in the presented agent world simulations we had $R = 2$ language rules) and all corresponding probabilities $p_{j,r}$, $j = 1,...,k_r$ and calculated the variance of the values for each of where $K = \Sigma_{r=1,R} k_r$. This language complexity measure is inspired by the concept of Kolmogorov complexity (Li and

Figure 2. The evolution of the level of cooperation and the level of language complexity for agent communities without wide dispersion of offspring and social learning by full copying of some language rules (scenario III): A) level of cooperation with $\eta = 0.2$; B) level of cooperation with $\eta = 0.8$; C) level of language complexity with $\eta = 0.2$; D) level of language complexity with $\eta = 0.8$; where η is the level of copying.

Vitanyi, 1997) in the sense that more variable application of the language rules (higher variance of the corresponding probability values) requires a longer description of the language than the description of a language with the same number of rules but less variable application of the rules.

We expect that allowing the agents to use social learning increases the steady-state level (i.e. after many time turns, when this level gets stabilized) of cooperation in agent communities due to the copying of successful neighboring agents who are expected to be the ones that often cooperate. It is also expected that social learning will reduce the complexity of the language across the agent community, again due to the copying of language rules between agents.

First we considered the scenarios without wide dispersion of the offspring of the agents – scenarios (I), (III) and for reference also scenario (V). For both variants of social learning we analyzed the evolution of the level of cooperation and of the communication complexity for low and high levels of behavioral copying. The results are shown in Figures 1 and 2. These confirm that in both cases of social learning the steady-state level of cooperation grows with the level of environmental risk similar to previously reported results (Andras et al, 2003; Andras et al, 2007). Also, similarly to previous results (Andras, 2008) the results show that the steady-state language complexity decreases with the environmental risk.

Increased level of copying in social learning leads to smaller differences in terms of steady-state levels of

Figure 3. The evolution of the level of cooperation and the level of language complexity for agent communities without wide dispersion of offspring and social learning by partial copying of all language rules (scenarios I and V): A) level of cooperation with $\sigma = 0.3$; B) level of cooperation with $\sigma = 0.7$; C) level of language complexity with $\sigma = 0.3$; D) level of language complexity with $\sigma = 0.7$; where σ is the level of environmental risk.

cooperation for different levels of environmental risk, for example the differences for $\sigma = 0.7$ and $\sigma = 0.9$ become statistically not significantly different for both kinds of

Figure 4. The evolution of the level of cooperation and the level of language complexity for agent communities without wide dispersion of offspring and social learning by full copying of some language rules (scenarios III and V): A) level of cooperation with $\sigma = 0.3$; B) level of cooperation with $\sigma = 0.7$; C) level of language complexity with $\sigma = 0.3$; D) level of language complexity with $\sigma = 0.7$; where σ is the level of environmental risk.

Figure 5. The evolution of the level of cooperation and the level of language complexity for agent communities with wide dispersion of offspring and social learning by partial copying of all language rules (scenario II): A) level of cooperation with $\eta = 0.2$; B) level of cooperation with $\eta = 0.8$; C) level of language complexity with $\eta = 0.2$; D) level of language complexity with $\eta = 0.8$; where η is the level of copying.

social learning. On the other hand, increased level of copying in social learning leads to increased differences between the steady-state levels of language complexity associated with different levels of environmental risk.

We also note that an impact of the social learning is that at the beginning (until over 120 time turns) the ordering of the language complexity levels associated with risk levels is reversed, i.e. low risk level implies low language complexity. In the absence of social learning the steady-state ordering of risk level associated language complexity levels is already established by around 80 time turns (see Figures 3 and 4). The time point, by which the steady-state ordering of language complexity levels emerges, changes with the level of copying. Interestingly in the case of social learning with partial copying of language rules, higher extent of copying implies delaying this time point, while in the case of social learning with full copying of some rules, the increase in the extent of copying makes this time point earlier.

Next we compared the levels of cooperation and language complexity for different extents of copying in the two kinds of social learning for two fixed levels of environmental risk $\sigma = 0.3$ and $\sigma = 0.7$. The results are shown in Figures 3 and 4.

The results indicate that social learning at small extent of copying does not change the level of cooperation. However, at lager extent of copying the impact is a statistically significant (t-test, $p=0.05$) increase in the level of cooperation. In terms of language complexity both kinds of social learning has a major effect in reducing earlier and by a considerable extent the level of language complexity. Interestingly this effect is
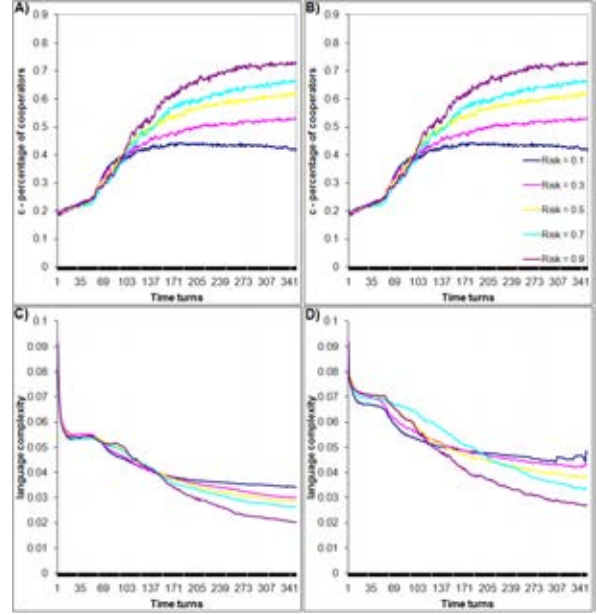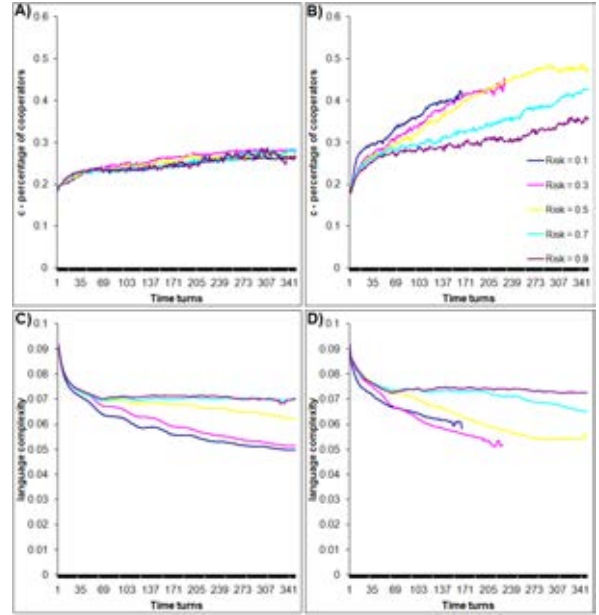
Figure 6. The evolution of the level of cooperation and the level of language complexity for agent communities with wide dispersion of offspring and social learning by full copying of some language rules (scenario IV): A) level of cooperation with $\eta = 0.2$; B) level of cooperation with $\eta = 0.8$; C) level of language complexity with $\eta = 0.2$; D) level of language complexity with $\eta = 0.8$; where η is the level of copying. The blue and red-purple lines stop early in B) and D) due to the early growth of the simulated populations beyond the population size limit.

larger at lower level of environmental risk and in the case of social learning by partial copying of all language rules the increase in the level of copying reduces the reduction effect on the language complexity.

Next we considered the simulation scenarios with wide dispersion of the offspring – scenarios (II), (IV) and (VI) for reference. The wide dispersion of the offspring reduces in general the level of cooperation in the agent communities, but the ordering of the levels of steady-state cooperation associated with levels of environmental risk remains the same as in the case without wide dispersion of the offspring in the case of agent communities without social learning.

For both kinds of social learning that we implemented we found that the steady-state level of cooperation associated with levels of environmental risk do not follow the ordering pattern found without social learning or with social learning but without wide dispersal of the offspring. In the cases of social learning with wide dispersal of the offspring lower environmental risk leads to higher level of cooperation – the difference becomes statistically significant for higher extent of copying in the social learning. In terms of language complexity again the ordering of the steady-state levels is the reverse of the ordering that we found for scenarios without wide dispersion of the offspring. Lower environmental risk implies higher language complexity in the case of agent societies with widely dispersed offspring and either form of
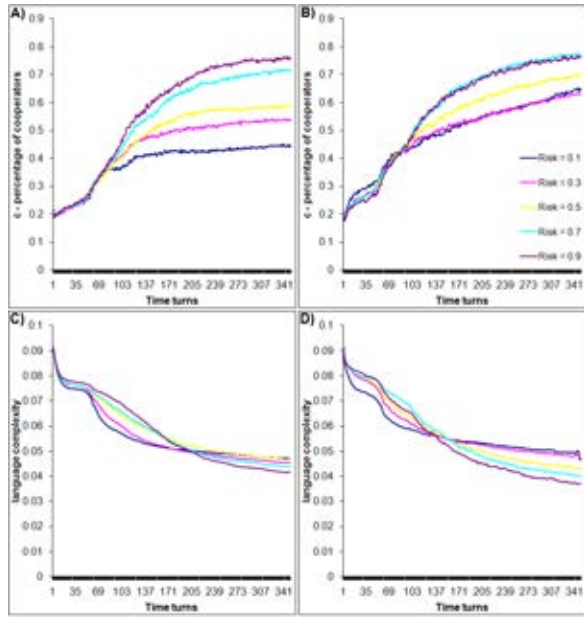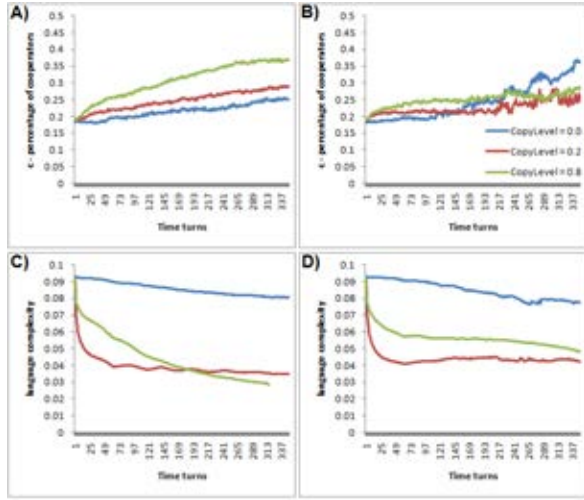
Figure 7. The evolution of the level of cooperation and the level of language complexity for agent communities with wide dispersion of offspring and social learning by partial copying of all language rules (scenarios II and VI): A) level of cooperation with $\sigma = 0.3$; B) level of cooperation with $\sigma = 0.7$; C) level of language complexity with $\sigma = 0.3$; D) level of language complexity with $\sigma = 0.7$; where σ is the level of environmental risk.
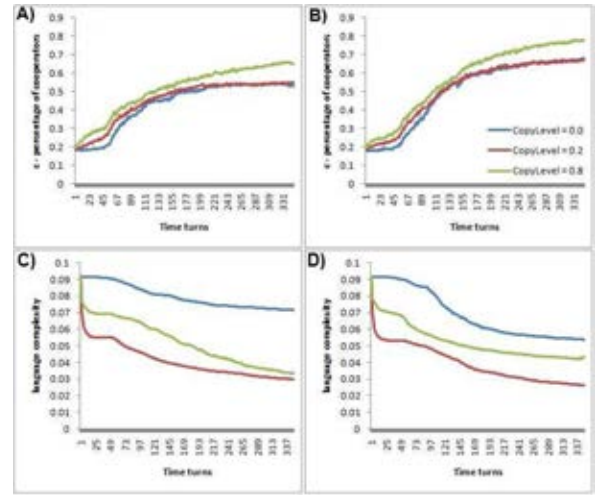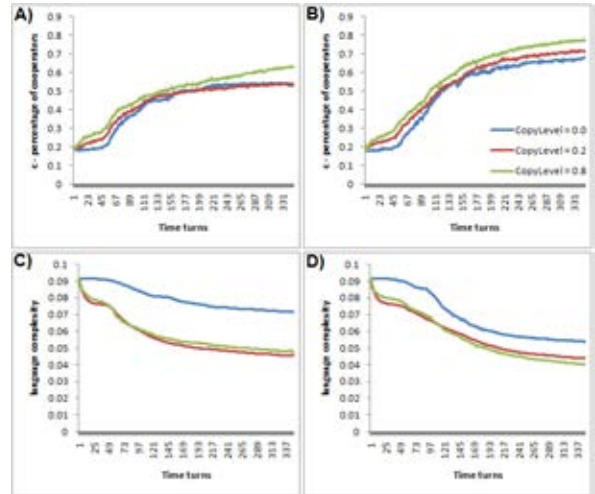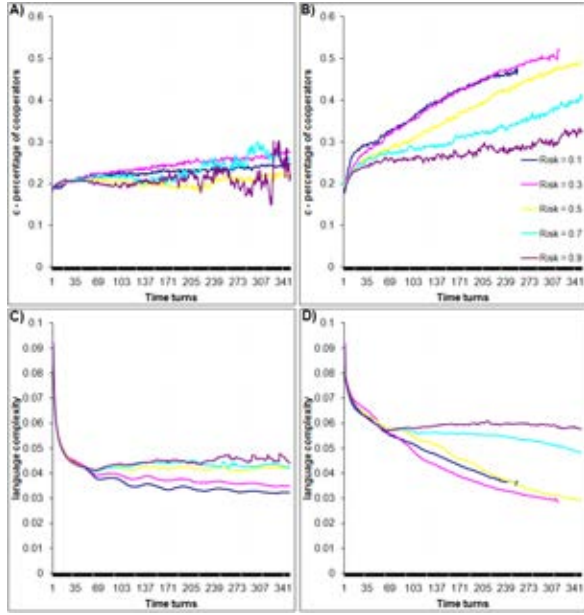
Figure 8. The evolution of the level of cooperation and the level of language complexity for agent communities with wide dispersion of offspring and social learning by full copying of some language rules (scenarios IV and VI): A) level of cooperation with $\sigma = 0.3$; B) level of cooperation with $\sigma = 0.7$; C) level of language complexity with $\sigma = 0.3$; D) level of language complexity with $\sigma = 0.7$; where σ is the level of environmental risk. The olive-green line stops early in A) and C) due to the early growth of the simulated populations beyond the population size limit.

social learning that we implemented. The results are shown in Figures 5 and 6.

The results show that higher extent of copying in social learning implies an increase in the steady-state level of cooperation for all levels of environmental risk for both kinds of social learning and this effect is stronger in the case of social learning with full copying of some language rules. Similarly, higher extent of copying in social learning increases the effect of environmental risk on the steady-state level of language complexity (i.e. the distinction between steady-state level of language complexity for high and medium level environmental risk becomes clearer). Again, the effect is more accentuated for the social learning with full copying of some language rules. We also note that the steady-state level of language complexity is lower for all levels of environmental risk in the case of social learning with partial copying of all language rules. The evolution of language complexity shows a wavy nature in all cases considered here, which is likely to be due to a generational effect (each generation of agents lasts for around 60 time units).

Further, we considered again two fixed levels of environmental risk ($\sigma = 0.3$ and $\sigma = 0.7$) and compared the corresponding levels of cooperation and language complexity for different extents of copying in the two kinds of social learning. The results are presented in Figures 7 and 8.

The results show that at lower level of environmental risk both kinds of social learning increase the level of cooperation relative to the case with no social learning. Notably even at higher levels of environmental risk, at the initial part of the evolution of the agent community the level of cooperation increases with the extent of copying in social learning. For both kinds of social learning, higher extent of copying leads to

higher level of cooperation at both environmental risk levels.

In terms of language complexity, again both kinds of social learning lead to a significant drop in comparison with the case with no social learning. This effect is much larger at the lower level of environmental risk. Higher extent of copying in social learning leads to smaller steady-state language complexity at the lower level environmental risk, at the higher level environmental risk the same effect is smaller. As we already noted, for both kinds of social learning higher level of environmental risk implies higher steady-state language complexity. The level of language complexity is lower for the social learning with partial copying of all language rules than for the social learning with full copying of some language rules for both considered values of extent of copying and for both considered levels of environmental risk.

Discussion and Conclusions

Our results show that in the simulated agent communities social learning has more effect on the level of cooperation and the level of language complexity at low level environmental risk than at high level of environmental risk. This difference is more accentuated in the case of simulations with wide dispersal of the offspring of agents.

We found that low extent social learning does not increase the level of cooperation and in the case of high environmental risk this may even reduce the level of cooperation. The extent of social learning influences the level of language complexity in all cases. More social learning leads to lower language complexity quicker in the context of low environmental risk.

A very interesting result is that in the case of simulations with wide dispersal of the offspring adding social learning to the simulations reversed the ordering of levels of cooperation and language complexity associated with levels of environmental risk, compared to the case without social learning. There is no such effect if the offspring of the agents are not dispersed widely in the space where the agents live.

The results suggest that social learning is most impactful in terms of supporting cooperation and reducing language complexity in the context of low environmental risk situations. High environmental risk situations support the emergence of relatively high level of cooperation and low level of language complexity even in the absence of social learning (Krams et al, 2010; Andras et al, 2007; Rand et al, 2014; Potts and Faith, 2015). Thus it is possible that animal or human populations develop high level of cooperation in harsh and risky environments without relying much on social learning, and these populations get to even higher level of cooperation and lower level of language complexity as they move to less harsh and less risky environments.

The results also suggest that social learning gets a much more significant role in communities where related individuals get dispersed widely in the community. In close knit communities where kin are likely to stay close to each other the simulation results suggests that the impact of social learning is mainly in terms of reducing the language complexity within the community.

The observations based on the simulation data that social learning may reduce the level of cooperation or increase the level of language complexity in high risk environments, and that in general it may have little effect on the level of cooperation at small extent of social learning, suggest that social learning has the potential to reduce cooperation in some settings (especially high environmental risk situations). This fits well with some of the experimental observations and theoretical explorations about how social learning may influence negatively the disposition towards cooperation of humans (Molleman et al, 2013; Burton et al, 2015).

In general the results presented here suggest that social learning and environmental risk may take alternating roles in driving animals and humans towards communities that rely increasingly on cooperation among individuals. High environmental risk is the first driver to higher level of cooperation in the community of individuals. Following a move to a low risk environment social learning takes over as driver toward more cooperation and lower language complexity. High level of cooperation in low risk environment combined with social learning may lead to the emergence of novel social structures that add new risks to the environment and also increase the language complexity (Boyce et al, 2012). This may lead to a new high risk environment which in turn facilitates further cooperation in the evolving community. Next, with the maturation of the previously new social structures the environmental risk may get reduced and the community may experience a new bout of increase in cooperation due to social learning. This way the evolving community may increase the level of cooperation and the extent of social institutions, in steps driven alternately by high environmental risk and social learning. The investigation of generation of novel social structures in agent-based simulations of communities will be part of future work.

References

Andras, P. (2008). Uncertainty and communication complexity in iterated cooperation games, in *Proceedings of ALife XI*, pp.9-16.

Andras, P., Lazarus, J., Roberts, G., and Lynden, S. J. (2006). Uncertainty and cooperation: Analytical results and a simulated agent society. *JASSS – Journal of Artificial Societies and Social Simulation*, 9:1/7.

Andras, P., Lazarus, J., Roberts, G. (2007). Environmental adversity and uncertainty favour cooperation. *BMC Evolutionary Biology*, 7:240.

Andras, P., Roberts, G., and Lazarus, J. (2003). Environmental risk, cooperation and communication complexity. In Alonso, E., Kudenko, D., and Kazakov, D., editors, *Adaptive Agents and Multi-Agent Systems*, pages 49-65. Springer-Verlag, Berlin.

Axelrod, R. (1997). *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*. Princeton University Press, Princeton, NJ.

Boyce, W.T., Sokolowski, M.B. and Robinson, G.E. (2012). Toward a new biology of social adversity, *PNAS*, 109:17143-17148.

Boyd, R. and Richerson, P.J. (2009). Culture and the evolution of human cooperation. *Philosophical Transactions of the Royal Society B*, 364:3281-3288.

Burton-Chellew, M.N., Nax, H.H., and West, S.A. (2015). Payoff-based learning explains the decline in ccoperation in public good games. Proceedings of the Royal Society B, 282:20142678.

Chudek, M., Zhao, W. and Henrich, J. (2013). Culture-gene coevolution, large-scale cooperation and the shaping of human social psychology. In: Sterelny, K., Joyce, R., Calcott, B. and Fraser, B (eds.) *Cooperation and Its Evolution*, MIT Press, pp.425-458.

Flinn, M.V. (1997). Culture and the evolution of social learning. *Evolution and Human Behavior*, 18:23-67.

Heyes, C. (2013). What can imitation do for cooperation? In: Sterelny, K., Joyce, R., Calcott, B. and Fraser, B (eds.) *Cooperation and Its Evolution*, MIT Press, pp.313-332.

Krams, I., Krama, T., Berzins, A. and Rantala, M.J. (2010). The risk of predation favors cooperation among breeding prey. *Communicative & Integrative Biology*, 3:243-244.

Li, M. and Vitanyi, P. M. B. (1997). *An Introduction to Kolmogorov Complexity and its Applications*. Springer, New York.

Molleman, L., Pen, I. and Weissing, F.J. (2013). Effects of conformism on the cultural evolution of coial behavior. *PLoS ONE*, 8:e68153.

Nakahashi, W., Wakano, J.Y. and Henrich, J. (2012). Adaptive social learning strategies in temporally and spatially varying environments. *Human Nature*, 23:386-418.

Potts, R. and Faith, J.T. (2015). Alternating high and low climate variability: The context of natural selection and speciation in Plio-Pleistocene hominin evolution. *Journal of Human Evolution*, 87:5-20.

Rand, D.G. and Nowak, M. (2013). Human cooperation. *Trends in Cognitive Sciences*, 17:413-425.

Rand, D.G., Peysakhovich, A., Kraft-Todd, G.T., Newman, G.E., Wurzbacher, O., Nowak, M.A. and Greene, J.D. (2013). Social heuristics shape intuitive cooperation. *Nature Communications*, 5:3677.

Rendell, L., Boyd, R., Cownden, D., Enquist, M., Eriksson, K., Feldman, M.W., Fogarty, L., Ghirlanda, S., Lillicrap, T. and Laland, K.N. (2010). Why copy others? Insights from the social learning strategies tournament. *Science*, 328:208-213.

Seltzer, N. and Smirnov, O. (2015). Degrees of separation, social learning, and the evolution of cooperation in a small-world network. Journal of Artificial Societies and Social Simulation, 18:12.

Sigmund, K., De Silva, H., Traulsen, A. and Hauert, C. (2010). Social learning promotes institutions for governing the commons. *Nature*, 466:861-863.

Spinks, A. C., Jarvis, J. U. M., and Bennett, N. C. (2000). Comparative patterns of philopatry and dispersal in two common mole-rat populations: implications for the evolution of mole-rat sociality. *Journal of Animal Ecology*, 69:224-234.

Finding a Mate With Eusocial Skills

Chris Marriott[1] and Jobran Chebib[2]

[1]University of Washington, Tacoma, WA, USA 98402
[2]University of Zürich, Zürich, Switzerland 8057
dr.chris.marriott@gmail.com

Abstract

Sexual reproductive behavior has a necessary social coordination component as willing and capable partners must both be in the right place at the right time. It has recently been demonstrated that many social organizations that support sexual reproduction can evolve in the absence of social coordination between agents (e.g. herding, assortative mating, and natal philopatry). In this paper we explore these results by including social transfer mechanisms to our agents and contrasting their reproductive behavior with a control group without social transfer mechanisms. We conclude that similar behaviors emerge in our social learning agents as those that emerged in the non-social learning agents. Social learners were more inclined towards natal philopatry. Social learners also evolved a culture of eusociality including reproductive division of labor.

Introduction

Sexual reproduction is a social behavior as two able participants must coordinate their behaviors as well as their positions in time and space. This social coordination problem is solved by sexually reproducing species in many different ways.

Some of the behaviors that enhance finding and attracting a mate include herding (Reynolds, 1987), philopatry (Clutton-Brock and Lukas, 2012), assortative mating (Jiang et al., 2013) and eusociality. These behaviors can arise through social mechanisms or non-social mechanisms (Whiten and Ham, 1992).

For instance, consider witnessing a herd of animals crossing a plane to drink water from the river. A well known explanation of the herd is that the animals in the herd follow simple social rules of cohesion, alignment and separation (Reynolds, 1987; Grünbaum and Okubo, 1994; Parrish and Edelstein-Keshet, 1999). These are social rules and social mechanisms because to follow these rules the agents must be aware of each other and make decisions based on information about others' states. However, a non-social explanation might be that all the animals were getting thirsty in the sun and independently navigated around obstacles to the river. With this second explanation, the herd is maintained by the mutual response of the individuals in the herd with no need for social awareness or exchange of information.

Prior work (from now on when we reference the prior work we mean the work in Marriott and Chebib (2015a,b)) showed that herding, philopatry, and assortative mating arose through non-social mechanisms of convergence and common descent. This work raised a few questions regarding the role that social interaction plays in many of these observed behaviors. These mating behaviors can be explained by both non-social and social mechanisms. In many cases the non-social solution is the simpler one, though it is likely that certain instances of these behaviors indeed rely on underlying social mechanisms. Further, it is not clear how these behaviors vary relative to the nature of the underlying mechanisms.

We have augmented the non-social agents from the prior work with both individual learning and social learning capabilities. Our null hypothesis is that the same breeding structures and organizations will be observed as before. However, we expect that the new adaptive mechanisms will impact these organizations quantitatively and may lead to other organizations. Our current work can be compared directly to the prior work but because of implementation differences between the models we have also conducted control tests of our own. These control tests involve the same agents but under conditions in which social learning and/or individual learning is unavailable.

Social Animals

Animals display different levels of social behavior (Michener, 1969). Social behavior often is centered on reproductive activities and caring for the young (Trivers, 1972). Some social animals can extend this social behavior to other activities like hunting, foraging, and grooming (Lovegrove and Wissel, 1988; Boesch, 1994; Creel and Creel, 1995; Nakamura, 2003). The highest categorization of social behavior in animals is *eusocial* (Crespi and Yanega, 1995). Ants, bees, termites, and some mole rats are categorized as eusocial (Wilson and Hölldobler, 2005). Humans, of course, are also very social and loosely fit under the eusocial defini-

tion (Nowak et al., 2010).

On the opposite side of the spectrum are *non-social* animals. Non-social animals engage in the minimal amount of social activity required of a sexual reproductive species, which is to mate. After mating the mother lays her eggs or gives birth to her young and leaves them to fend on their own (Starck, 1998). All parental investment in child rearing comes prior to birth.

To be classified as eusocial, animals must satisfy three conditions. Eusocial animals share responsibility for caring for their young, have reproductive division of labor, and have multi-generational communal cohabitation (and also philopatry in some definitions (Burda et al., 2000)). This means that parents live with adult children and older generations help to care for their grandchildren and others. This, in particular, allows for sharing of learned information across generations.

We can see that humans loosely fit into this definition. We certainly share responsibility for child rearing, and we certainly have multi-generational communal cohabitation. However, we do not have reproductive division of labor in the sense we normally think of. That is, we don't have a single "queen" that births us all after breeding with a few privileged, male courtiers. However, many still like to apply the label eusocial to humans while some prefer to keep humans in a category of their own. This debate is not critical to our discussion.

Model

Our simulation consists of agents in a random geometric network of resource sites. Each day the agents in a population expend energy to move from site to site, forage for resources, and engage in mating, learning and social learning. Energy in the simulation normally corresponds to the time an agent can spend doing activities during a day but excess stored energy is also used to reproduce. The resources gathered during a day determines the energy an agent has for activities in the next day. The net daily energy gain or loss determines whether an agent lives or dies (if the energy is depleted) and whether an agent is capable of reproduction (if stored energy exceeds a threshold).

Agents in our simulation implement the dual inheritance model (Marriott and Chebib, 2014, 2016b). The dual inheritance model is a model incorporating three modes of adaptation: phylogenetic, ontogenetic, and sociogenetic. That is, agents engage in genetic evolution, individual learning and social leaning. Figure 1 shows how genetic and cultural information are stored and transferred in the dual inheritance model.

Genetic and cultural information are stored in separate information stores called genomes and memomes, respectively. Genomes are inert within the lifetime of an agent meaning they remain static and are not active in behavior selection. Their only purpose is to create memomes upon
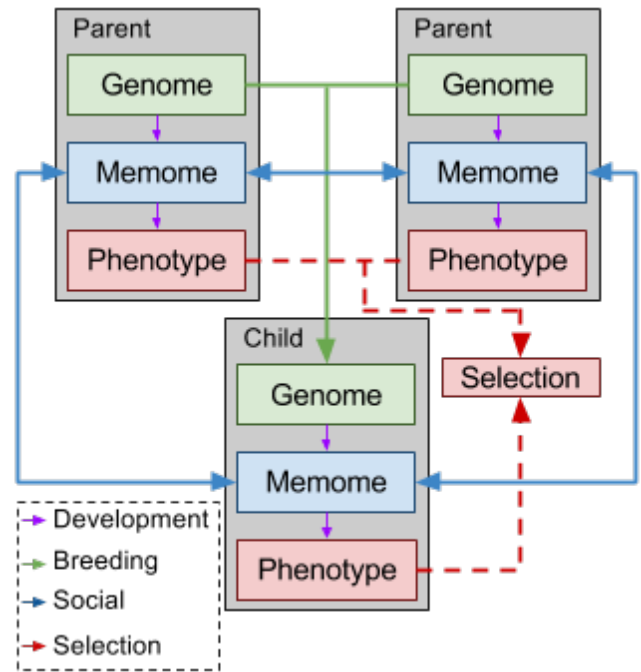
Figure 1: Dual Inheritance Model

birth and to be replicated in reproductive events. When an agent is born its memome is created by copying segments of the genome. This is a process of development that results in the agents initial cultural information.

Memomes are active in behavior selection over the lifetime of an agent and they can also be altered through individual learning and social learning events. The memome selects behaviors through interaction with the environment and these choices shape the phenotype of the agent. The phenotype is used for selection and thus an agent's reproductive success and survival is dependent upon its memome, its genome, and its environment.

The cognitive model of our agents is inspired by the pandemonium model (Jackson, 1987; Franklin, 1997). The mind of a single agent in our simulation consists of many specialized sub-agents, or daemons, that compete for control of the agent. In our model we call these sub-agents memeplexes and this internal competition is an evolutionary competition.

Genome

The possible behaviors of agents are encoded in their genomes as a path of resource sites in the network (our model extends the model from (Marriott and Chebib, 2015a,b)). We consider each site in the path a gene in the genome. A gene has three components: gathering, nongathering, and travel. At each site an agent has an opportunity to gather resources, perform non-gathering actions like breed, learn, and/or socialize, and finally must travel to the

next site.

In our current model an agent always gathers resources when it visits a site. When not depleted a site will always return a fixed number of resources to an agent gathering at that site. The energy that an agent spends on gathering is determined by an agent's strategy which is encoded in a gene corresponding to a site. The energy expended in gathering is always at least the number of resources gathered (one, two or three) and at most five.

A gene for a site will also encode how much time (i.e. energy) is spent performing breeding, learning or socializing actions. Typically the time performing these actions is considerably less than the time gathering resources. Actions at a site are performed in the following order: gathering, breeding, learning, socializing, and finally traveling to the next site.

Agents can engage in sexual reproduction when they have an energy total above a reproduction threshold. In addition, they need to find another willing and able participant at the same site. If two agents are engaged in breeding actions at the same site at the same time and they both have sufficient stored energy then they will engage in sexual reproduction. If no mate is found then agents will wait until the next opportunity to mate. If the breeding action is unsuccessful an agent must still spend the energy cost of the breeding action but not the cost of reproduction.

A genome is passed to the offspring during reproduction. In this process, the two parents' genomes are recombined into a single genome and this genome is given the opportunity to mutate. A genome remains inert during the lifetime of an agent other than during reproductive events.

At birth each agent's genome creates a memome. A memome consists of a set of memeplexes. Each memeplex in our model represents a possible set of activities for a day. Memeplexes are formed by copying segments of a genome. Starting at each gene (i.e. site) in a genome we copy gene by gene (site by site) into a memeplex (called memes in a memeplex instead of genes). This continues until the total energy of a segment approaches the maximum energy available to an agent for a single day. If copying the next gene would exceed the maximum energy the segment is complete and is stored as a memeplex. This means each memeplex represents a possible set of actions for our agent in a single day. A memeplex is stored in a memome along with other memeplexes starting with the same initial site for behavior selection (see below).

A single memeplex is formed starting at each gene in a genome. Copying occurs in the forward direction until the maximum energy is reached. If the end of a genome is reached copying continues backwards until the maximum energy is reached. Additionally, segments are copied in a backwards direction from every gene. This means every gene in a genome is responsible for two memeplexes in its memome except the endpoints that are responsible for only

a single memeplex. Notice that since every site in the environment is not necessarily represented in a genome there may be sites that do not have corresponding memeplexes.

Memome

As mentioned above memeplexes in a memome are arranged by starting site. The model bears similarities to the MAP-elites strategy of multi-objective evolutionary optimization (Mouret and Clune, 2015). For each site in the environment a memome will contain zero or more memeplexes that start at that site. Each memeplex will consist of a path of gathering sites that begin at a particular site and encode a path that takes at most the maximum energy (time) available to an agent for a single day. This means each memeplex encodes a possible course of actions for an agent for one day starting at a particular site. From the perspective of the pandemonium model, each memeplex is a daemon representing a single day's activities.

At the beginning of each day an agent must select a memeplex that will serve as its behavioral plan for that day. Only the memeplexes that begin at the current site are possible and so only these memeplexes are considered when selecting behavior. Since memeplexes in a memome are organized by starting site behavior selection begins by activating all memeplexes that start at an agent's current site. The agent selects the memeplex from this set that will maximize expected resource gain while minimizing expended energy. This memeplex is then used to determine the actions of the agent for that day. These actions interact with the environment to reward the agent with resources which serves as a selective force on the agent.

A memome is not only active in behavior selection but also adaptive over the lifetime of an agent. It serves as the information storage for both individual and social learning. This means that additional memeplexes are added to its memome as the agent interacts with its environment and other agents.

If the memeplex determining an agent's actions contains a meme with a non-zero learning component, then the agent will engage in learning during its day. During individual learning we apply an evolutionary process. We clone and mutate the memeplex selected for this day's actions and add it to the agent's memome.

Social learning is handled in the same way as breeding in our agents. If an agent wants to engage in social learning it must spend time seeking a partner for exchange. If another agent is also seeking a social learning partner at the same time and at the same site, then the two agents engage in an exchange of memeplexes. Each agent copies and exchanges the memeplex they used for that day. In transfer, memeplexs are mutated so noise is added to the system in this step. The social learning mechanism allows agents to pass their learned memeplexes on to others in the population. Agents are only allowed a single transfer in a day.

Experimental Setup

Simulations are run under two conditions: experimental and control. We ran both conditions 130 times in a variety of different environments. The experimental simulation consists of a population of agents with all of the mechanisms described above. The agents in this group are called *socializers*. There were initially two control populations, one in which agents can learn individually but not socially (where agents are called *learners*), and one in which agents can neither learn individually nor socially (and where the agents are called *breeders*). Learners did not perform significantly differently from breeders in our experiments so our analysis will focus on differences between socializers and breeders.

Each simulation run in our current experiment begins with a population of one hundred agents with randomly generated genomes. The genes generated at random in this initialization phase have a chance of having a non-zero breeding component to allow for the initial population to breed. However, all learning and social learning components of randomly generated genes are set to zero. They can only become non-zero through mutation.

Simulation runs are seeded with one hundred randomly generated individuals. Since they are randomly generated any genetic similarity occurs by chance alone. As shown in prior work it is a rare but fortunate occurrence if two randomly generated agents end up performing breeding actions at the same site at the same time (Marriott and Chebib, 2015b). Since this is the criteria for breeding in our model this means our initial population has a risk of not being viable. A run is seeded with one hundred agents so that this chance is diminished. Under these settings every seed population for a simulation run was viable.

In every run a small proportion of the initial population are fortunate enough to reproduce. Once initiated these colonies tend to quickly become viable due to forces of common descent. That is, since the offspring of these reproductive events are related to their parents there is a high chance that they will perform similar behaviors, and importantly, breed at the same time and place as their parents and others in their genetic family. This helps make the fledgling colony viable.

Each run is allowed to run for 5000 days. During the run some data was gathered continuously and other data was sampled every 50 days. For every reproductive event, relatedness of the parents is measured along both genetic and phenotypic lines. This allows us to monitor the degree of genetic and phenotypic assortative mating present in a run. Further, whenever an agent dies, the number of offspring an agent had is recorded as well as other characteristics of the agent's breeding history (like breeding sites and breeding partners).

In addition to this continuously gathered data the population is also censused every 50 rounds. During censusing, the population size is recorded as well as many statistics from
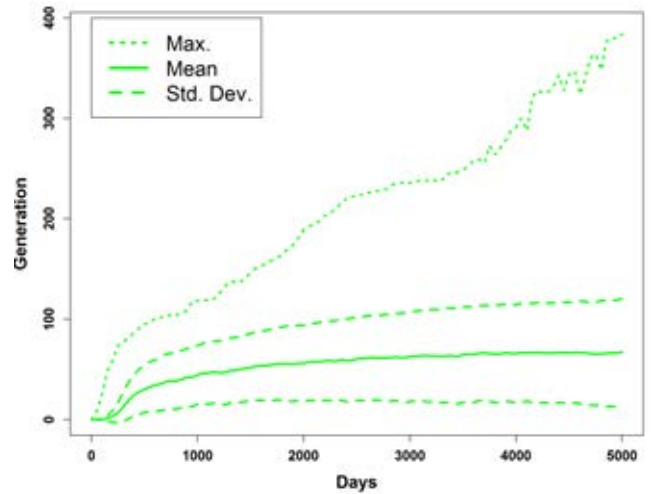
Figure 2: Average and maximum generation of active memeplexes over time in socializers with one standard deviation around the mean.

every agent alive during that day including: the agent's age, genome length, generation, memeplex generation, as well as the concentration of breeding/learning/socializing components in the agent's genome and memome.

Observations and Discussion

We indeed saw evidence of herding, assortative mating and philopatry as expected. In some cases, there are quantitative differences of note. However, the more exciting results is the emergence of eusociality in our socializers. Before discussing eusociality let's review some evidence of social learning in our agents.

Social Learning and Cultural Evolution In order to track social learning and cumulative cultural evolution we assigned each memeplex a generation. Initial memeplexes created at birth are assigned generation zero. When a memeplex is cloned the new memeplex has a generation one greater than its parent. Cloning occurs only during learning and social learning.

Breeders never clone their memeplexes and so they always act on memeplexes of generation zero. Learners can increase their memeplex generation but cannot share this with others. Socializers clone their memeplexes in individual learning and in social learning. In order to test that cumulative cultural evolution occurred in our socializers we have measured memeplex generation over time (see Fig. 2).

The maximum and average memeplex generation increases over time. The average memeplex generation grows much slower but the standard deviation also grows over time. The minimum memeplex is almost always zero because there are usually newborns in the world that have not yet learned the shared memeplex of the population. A rare
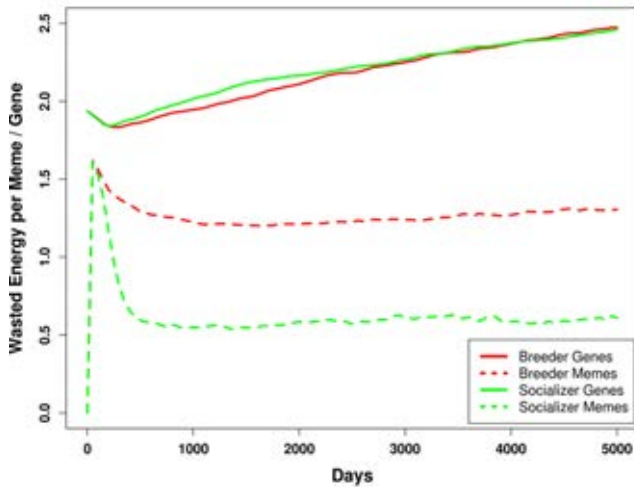
Figure 3: Wasted energy in genes in the genome and memes in the selected memeplex are measured. We show the average wasted energy in the population over time.

Figure 4: Proportion of the selected memeplex devoted to non-gathering actions averaged over all members of the population.

cases when this does not occur is during a colony collapse in which there are no newborns (see below).

This is evidence that cumulative cultural evolution is occurring (Marriott and Chebib, 2016a). This increase over time implies that memeplexes are improved in one generation, shared among the members and passed to the next generation. Improvements made early are preserved in the population from generation to generation. We have tracked this optimization over time as well.

In Fig. 3 the divergences between gene optimization and meme optimization shows that both breeders and socializers can optimize their behavior. For breeders this can only occur by optimizing the local parts of their genome that are copied into the memome and eventually used by the agent. For socializers this means optimizing the shared memeplexes that are passed between agents.

The data show are averages over the population. New born socializers have not had a chance to learn the shared memeplexes and thus have memeplexes similar to the breeders that don't optimize. The most optimized memeplexes in the population have 0 wasted energy after about day 500. Since this is the most optimized the memeplex can get this slows the cumulative evolution. The only role of the social learning after this optimization is to maintain the highly optimized memeplex (or one of its clones) from generation to generation.

Another means of optimizing memeplexes is to eliminate time spent on the non-gathering activities: breeding, learning, and socializing. We see (Fig. 4) that breeders evolve to spend more time engaging in these actions over time as selection pressure against this is weak. Cultural evolution occurs much quicker and so the weak pressure becomes much stronger over the same time.
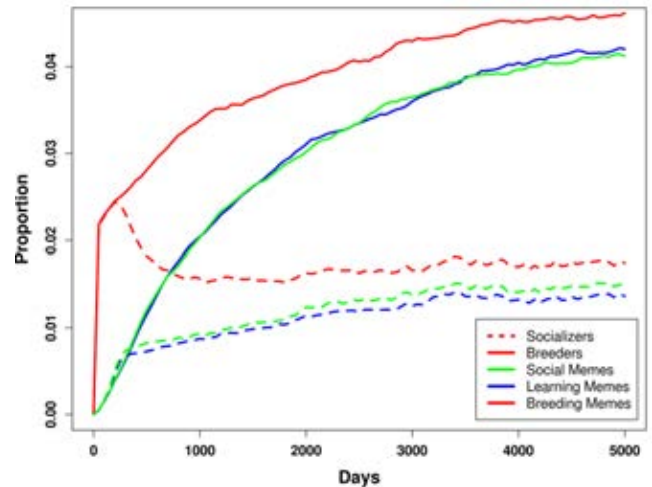
We see that the socializers optimize to spend much less time on these actions over the first 1000 days. The most optimized memeplex would spend no energy breeding, learning or socializing. This is rare though it occurs. A memeplex with no breeding component is quite common and suppresses reproduction in the agent if selected repeatedly (see below). A memeplex with no learning component is not a big detriment since the memeplex is likely already very optimized. A memeplex without a social learning component makes it impossible to spread itself. While more optimal these are rare since they die with the host agent. As a result most optimized memeplexes spend no energy or a very small amount of energy on breeding and usually a little more energy on learning and socializing. The success of the memeplex depends on its ability to optimize and spread itself.

Herding Although we have not conducted a quantitative analysis of herding in our agents, we can analyze the underlying mechanisms supporting the herds we observe. We know from prior work that the herds of the control group are maintained by common descent (Marriott and Chebib, 2015b). Herds in the socializers are also maintained through the common decent of the memeplexes shared among members of the herd. Herds observed in socializers, even if outwardly similar, are being maintained by social mechanisms.

Assortative Mating Assortment of genetically related parents are not significantly affected by the presence of learning and social learning (see Fig. 5). There are some small differences in assortment of phenotypically related parents though it is not clear how to interpret this slight variation (see Fig. 6).

302
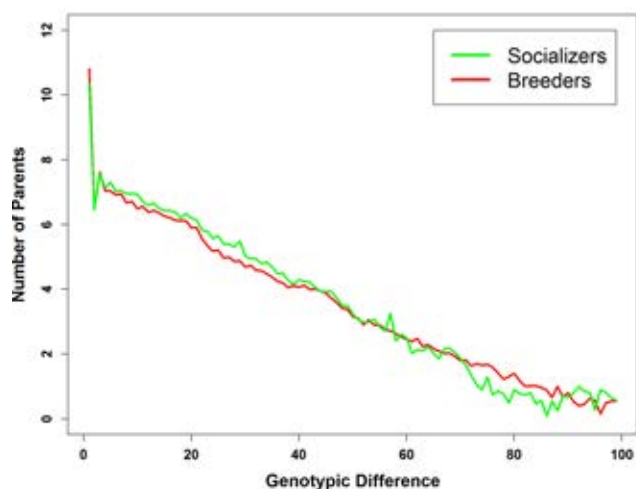
Figure 5: Genetic difference between parents. This data is plotted on a log scale with base 2 and is averaged over the 130 runs.
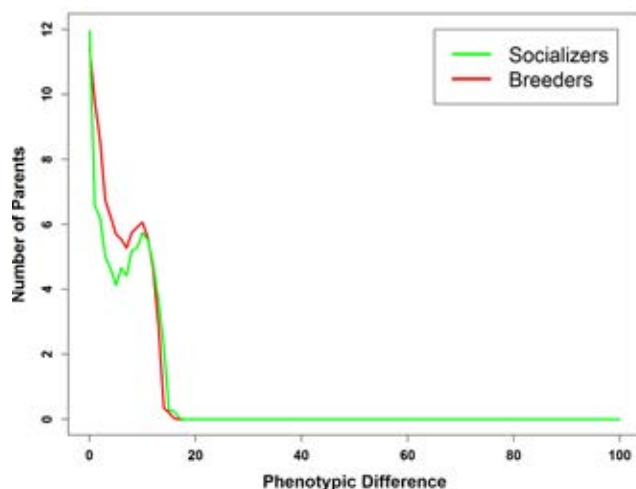
Figure 6: Phenotypic difference between parents. This data is plotted on a log scale with base 2 and is averaged over the 130 runs.

Philopatry When an agent would die we would record it in one of four categories. If the agent died childless then it could not be evaluated for natal philopatry. If the agent had children there were three possibilities: they only bred at their birth site, they sometimes bred at their birth site, or they never bred at their birth site.

Breeders had an average of 45.6% childless agents while socializers had an average of 63.6% childless agents. Of the agents that had children, there were 59.9% of breeder agents with children that never bred at their birth site. In the socializers there were 49.7% of agents with children that never bred at their birth site. Fewer agents are breeding in the socializers but more of them are breeding at their birth
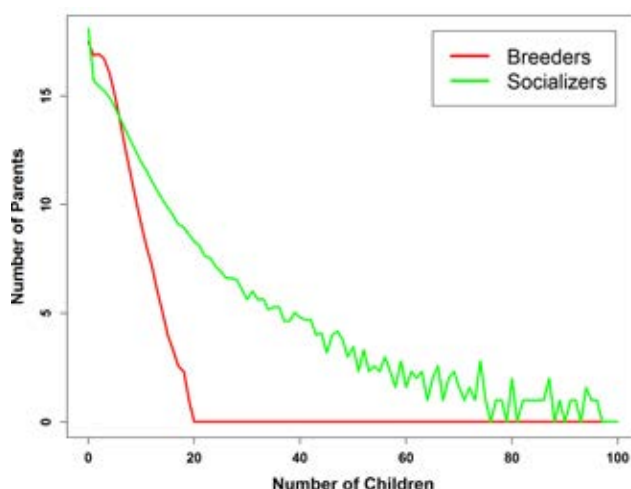
Figure 7: The number of children each agent had. This data is plotted on a log scale with base 2. This data is cumulative over all 130 runs.

site. 52.8% of breeders that bred at their birth site at least once did not breed elsewhere during their life, whereas in socializers 75.3% bred only at their birth sites.

While socializers were more likely to die childless they appear to engage in more natal philopatry than the breeders. They were more likely to breed at their birth site and they were more likely to breed exclusively at their birth site.

Eusociality Recall the conditions of eusociality among animals. Agents must have multi-generational communal cohabitation, mutual care for the young, and reproductive division of labor (and sometimes natal philopatry).

Both breeders and socializers have multi-generational communal cohabitation. We know both display natal philopatry though it is stronger in socializers. So we must evaluate whether our agents have reproductive division of labor and mutual care for the young.

At death we recorded the number of children the agent had during its life. We have plotted this data showing how many agents died with n children (see Fig. 7). This plot is on a logarithmic scale and shows an exponential drop off as number of children increases. Socializers have a shallower decrease as number of children increase. No breeders had more than twenty children. Among socializers some agents, though rare, have more than ninety children. Combining evidence from above with this we see that both fewer agents are engaging in sexual reproduction and those that do are reproducing more. This apparently meets the criteria of division of sexual reproductive labor.

We also gathered data on the age of the eldest agent in the population. We notice that among the socializers there are older agents than among the breeders. To have many children an agent must live long enough to birth each child

and gather the energy required for this activity. This would require an old agent. It would also require an optimized memeplex that spent time breeding every day. If this agent also spread this memeplex to others the efficient breeding culture can be introduced and maintained in the population.

Agents that do not spend energy breeding can save a lot of energy, which can extend their lifespan. Recall that the most optimized memeplexes spend no time breeding so all energy can be stored for a long productive childless life. If this agent also spreads this memeplex to others the childless culture can be introduced and maintained in the population. If the memeplex spread to the whole population then the population will die out (see below).

Among the optimized memeplexes the dominant kinds are those with breeding and those without breeding. If spread each will generate a different kind of culture. In our populations these optimized memeplexes occupy agents in the same population. Not all agents have learned one of these dominant cultures and engage in sub-optimal culture. This is common of younger agents. These different cultures compete for the participation of agents.

We treat the sharing of optimal memeplexes to the young as a type of brood care in our simulation as we have not realized a brood care mechanism in our agents. Using this as a tool to evaluate brood care in our agents we can assess the whether there is mutual care for the young.

Agents in our model do not discriminate when social learning. They can't recognize their parents or young. Though there is a higher probability of two related agents occupying the same locations in the environment. Social learning is also bidirectional. Both agents act as teachers and learners. However, usually only the less experienced agent benefits from this exchange.

Now consider an agent with an optimized memeplex that has a breeding component. After some time this agent produces an offspring. The newborn has a sub-optimal memeplex and will likely travel a different path than the parent. If the offspring is lucky in the next few days the agent will encounter its parent and engage in social learning. Then the offspring may become an optimal breeder as well. This lucky agent provided direct parent-child brood care. This is not the only possibility.

The offspring occupies a population with others. Some may be older siblings, cousins, uncles, etc. while others are more distantly related. When the offspring first gets a chance to socially learn it might learn from one of these other agents (remember they don't discriminate). This results in the exchange of sub-optimal memeplexes but possibly also the enculturation of some sub-optimal culture that brings the offspring away from its parent. We might consider this a kind of mutual brood care.

Finally, let's imagine one of the others in the population has an optimal memeplex that avoids breeding. This agent cannot reproduce and create its own brood. It can only spread its memeplexes to the offspring of agents that breed. Thus, the existence of this culture relies on the care of other agent's offspring. We have strong evidence that these cultures do indeed exist (see below).

Together this evidence suggests that our socializers have emerged a type of eusociality. The have multi-generational cohabitation with mutual care for the young. Like humans they have an interesting culture based division of reproductive labor. The also engage in natal philopatry more often than the breeders.

Colony Collapse When a highly optimized memeplex has no breeding component but does have social learning it can spread into the population as discussed above. The danger of this culture is that if every agent in the population follows it then the population will die out.

This cultural suppression can have catastrophic consequences. A typical run will begin with a handful small colonies of agents in different parts of the random geometric network. When a culture of not breeding emerges and spreads to every agent in one of these colonies the population dies out. In 21 out of 100 socializer runs this led to every agent in the simulation dying before round 5000. This never occurs in breeder or learner runs. Inspection of the memeplexes of agents during a collapse confirms that there are no breeding components and most agents have a large store of energy and a long lifespan.

Conclusion

We were curious how social learning would affect strategies of sexual reproduction in our simulated agents. We did not see significant differences between the assortative behavior of breeders and socializers. We interpret this result as additional evidence that assortment can and probably is maintained in most populations by non-social forces.

Social learning also appeared to enforce a higher rate of natal philopatry. While fewer social learning agents bred, more of them bred at their birth sites and more of them bred exclusively at their birth site. This suggests that sociality and natal philopatry may correlate in natural populations.

Finally after adding social learning to our agents we find that they evolve a culture of eusocial reproduction. Reproductive labor is more concentrated both in a sub-population and occasionally within agents that breed considerably more than others in the population.

All agents that engage in social learning can be considered to engage in brood care by sharing culturally learned information to others. As they don't discriminate when social learning and there is multi-generational cohabitation this brood care occurs between agents of different generations and of different relatedness.

These are the criteria for eusociality applied to animals and in applying these criteria to our agents we can see there is evidence to call them eusocial. It is interesting to us that

our relatively simple social exchange mechanism is strong enough to evolve a eusocial culture in our agents. It is noteworthy that this eusociality is maintained by cultural forces not genetic forces. That is, whether our social agents breed or not is not dependent upon their genetics, but rather on their learned culture.

Acknowledgments

References

Boesch, C. (1994). Cooperative hunting in wild chimpanzees. *Animal Behaviour*, 48(3):653–667.

Burda, H., Honeycutt, R. L., Begall, S., Locker-Grütjen, O., and Scharff, A. (2000). Are naked and common mole-rats eusocial and if so, why? *Behavioral Ecology and Sociobiology*, 47(5):293–303.

Clutton-Brock, T. and Lukas, D. (2012). The evolution of social philopatry and dispersal in female mammals. *Molecular Ecology*, 21(3):472–492.

Creel, S. and Creel, N. M. (1995). Communal hunting and pack size in african wild dogs, lycaon pictus. *Animal Behaviour*, 50(5):1325–1339.

Crespi, B. J. and Yanega, D. (1995). The definition of eusociality. *Behavioral Ecology*, 6(1):109–115.

Franklin, S. (1997). *Artificial minds*. MIT press.

Grünbaum, D. and Okubo, A. (1994). Modelling social animal aggregations. In *Frontiers in mathematical biology*, pages 296–325. Springer.

Jackson, J. V. (1987). Idea for a mind. *ACM SIGART Bulletin*, (101):23–26.

Jiang, Y., Bolnick, D. I., and Kirkpatrick, M. (2013). Assortative mating in animals. *The American Naturalist*, 181(6):E125–E138.

Lovegrove, B. and Wissel, C. (1988). Sociality in molerats. *Oecologia*, 74(4):600–606.

Marriott, C. and Chebib, J. (2014). The effect of social learning on individual learning and evolution. In *The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, pages 736–743. MIT Press.

Marriott, C. and Chebib, J. (2015a). Emergence-focused design in complex system simulation. In *European Conference on Artificial Life*. MIT Press.

Marriott, C. and Chebib, J. (2015b). Finding a mate with no social skills. In *Proceedings of the 2015 conference on Genetic and Evolutionary Computation*. ACM.

Marriott, C. and Chebib, J. (2016a). Divergent cumulative cultural evolution. In *Proceedings of the 2016 Conference on the Synthesis and Simulation of Living Systems*. MIT Press.

Marriott, C. and Chebib, J. (2016b). Modelling the evolution of gene-culture divergence. In *Proceedings of the 2016 Conference on the Synthesis and Simulation of Living Systems*. MIT Press.

Michener, C. D. (1969). Comparative social behavior of bees. *Annual review of entomology*, 14(1):299–342.

Mouret, J.-B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*.

Nakamura, M. (2003). gatherings of social grooming among wild chimpanzees: implications for evolution of sociality. *Journal of human evolution*, 44(1):59–71.

Nowak, M. A., Tarnita, C. E., and Wilson, E. O. (2010). The evolution of eusociality. *Nature*, 466(7310):1057–1062.

Parrish, J. K. and Edelstein-Keshet, L. (1999). Complexity, pattern, and evolutionary trade-offs in animal aggregation. *Science*, 284(5411):99–101.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM Siggraph Computer Graphics*, 21(4):25–34.

Starck, J. M. (1998). *Avian growth and development: evolution within the altricial-precocial spectrum*. Number 8. Oxford University Press on Demand.

Trivers, R. (1972). *Parental investment and sexual selection*, volume 136. Biological Laboratories, Harvard University.

Whiten, A. and Ham, R. (1992). On the nature and evolution of imitation in the animal kingdom: reappraisal of a century of research. *Advances in the Study of Behavior*, 21:239–283.

Wilson, E. O. and Hölldobler, B. (2005). Eusociality: origin and consequences. *Proceedings of the National Academy of Sciences of the United States of America*, 102(38):13367–13371.

Task Allocation in Foraging Robot Swarms: The Role of Information Sharing

Lenka Pitonakova[1,3], Richard Crowder[1] and Seth Bullock[2]

[1]Institute for Complex Systems Simulation and Department of Electronics and Computer Science,
University of Southampton, United Kingdom
[2]Department of Computer Science, University of Bristol, Bristol, United Kingdom
[3]contact@lenkaspace.net

Abstract

Autonomous task allocation is a desirable feature of robot swarms that collect and deliver items in scenarios where congestion, caused by accumulated items or robots, can temporarily interfere with swarm behaviour. In such settings, self-regulation of workforce can prevent unnecessary energy consumption. We explore two types of self-regulation: *non-social*, where robots become idle upon experiencing congestion, and *social*, where robots broadcast information about congestion to their team mates in order to socially inhibit foraging. We show that while both types of self-regulation can lead to improved energy efficiency and increase the amount of resource collected, the speed with which information about congestion flows through a swarm affects the scalability of these algorithms.

Introduction

Congestion is an important factor that can negatively affect the performance of robot swarms (Hoff et al., 2010). Today's robotic systems, utilised in automated warehouses (D'Andrea, 2012), agriculture (Cartade et al., 2012), or in hospitals (Thiel et al., 2009), must maintain effective work schedules for individual robots in order to minimise interference between robots and save energy. Decentralised task allocation, which affords redundancy and scalability, has been proposed as a possible solution (Wawerla and Vaughan, 2010; D'Andrea, 2012) that is likely to become more important in the near future as autonomous robot swarms will increasingly be deployed in unstructured and dynamic environments. In this paper, we explore the problem faced by a robot swarm that collects items from the environment and can cope with congestion by regulating its workforce in a decentralised manner in order to save energy. Furthermore, we explore how the means by which information about congestion is obtained by the robots affects the scalability of the swarm's performance under various foraging conditions.

During foraging, congestion can either result from the size of the robot population or from the structure of the environment. For example, robots might be required to wait until an occupied resource drop-off location becomes accessible (Wawerla and Vaughan, 2010) or they might need to queue in order to leave a crowded drop off location to perform more work (Krieger and Billeter, 2000). An autonomous robot swarm should be able to sense when congestion has become a problem and adjust its workforce accordingly.

We simulate robot swarms that collect items from the environment and drop them off in a central base, from where the items are consumed at a given rate. We explore two types of workforce self-regulation: *non-social*, where robots become idle upon directly experiencing severe congestion, and *social*, where a robot will inhibit the foraging of its team mates by signalling them to become idle when the congestion that it experiences is severe. We show that both types of self-regulation can lead to significant energy savings and thus to a greater number of items collected when the energy available to the robots is limited. More importantly, we demonstrate that the speed with which information about congestion flows through a swarm, either when robots detect congestion themselves or when they exchange information with their team mates, affects the swarm's ability to respond to it appropriately. While social self-regulation results in rapid information flow and can lead to significant performance benefits in certain scenarios, it can also lead to significantly worse performance in others. On the other hand, non-social self-regulation, where information flow is slower, leads to improved energy efficiency across a greater number of foraging conditions, making it more suitable in unknown environments, although it is outperformed by social self-regulation in some cases.

The following sections provide an overview of related work and a description of our simulation and analysis methods. We then compare, across a number of experimental scenarios, the performance of our two types of self-regulated swarms with that of control swarms that do not use self-regulation. We evaluate both the amount of energy needed to collect items and the number of items collected when robot energy is limited. We conclude with a discussion of how our results relate to our previous work on information flow in swarms (Pitonakova et al., 2016) and provide examples of real-world applications where the two types of self-regulated swarms could be used.

Related Work

Route planning systems that optimise robot traffic are often used in controlled warehouse environments with small robot teams (Vivaldini et al., 2010; Mather and Hsieh, 2012). However, such approaches require a centralised controller to guide robot behaviour and are thus unsuitable for large swarms, where computation of optimal solutions becomes infeasible (Dahl et al., 2009). Furthermore, a model of the environment and of the tasks within it, which centralised planning systems rely on, can be difficult to obtain in dynamic or unstructured environments. On the other hand, decentralised decision making, where robots change their behaviour based on limited local information obtained through their sensors, is more suitable for complex tasks of this type (Hoff et al., 2010).

Decentralised robot decision making has been well studied in a number of logistic and foraging applications. For example, unmanned vehicles that need to transport items from one location to another can adjust their work time and decide to recruit others based on the number of items in pick-up locations (Wawerla and Vaughan, 2010). In behaviour-based robotics, a combination of environmental cues, such as the presence of items or other robots nearby, can trigger or inhibit foraging behaviour, leading to self-organised division of labour between robots that are idle and those that collect resources (Jones and Mataric, 2003). Alternatively, 'bucket brigading' robots can form chains of work areas and progressively transport items between two locations (Shell and Mataric, 2006; Pini et al., 2013) and even adapt the size of the work areas based on collisions with other robots in order to improve their performance (Lein and Vaughan, 2009).

The *Response Threshold Model* (RTM) is a self-regulatory mechanism inspired by social insects (Bonabeau et al., 1997) that has been applied in a number of simulated and real-world robot experiments. According to the model, robots alternate between *foraging* and *resting* based on some internal, environmental or social cues in order to optimise their energy consumption. For example, robots can count the number of items stored in the base and only leave to forage when the number is below a specified threshold (Yang et al., 2009). Robots can also evaluate how many items they encountered during foraging and decide to rest if the environment is not rich enough (Labella et al., 2006). By counting the number of collisions with other robots (Liu et al., 2007) or by detecting drops in their own expected performance (Dahl et al., 2009), robots can decide to rest if they estimate that congestion is beyond an acceptable level. Finally, in dynamic environments, where the number of items in the environment changes over time, robots can decrease the energy cost of collecting items by only foraging when enough items are available, estimating the state of the environment in either a centralised (Liu et al., 2007) or decentralised (Dai, 2009) manner.

Our work builds on the Response Threshold Model liter-
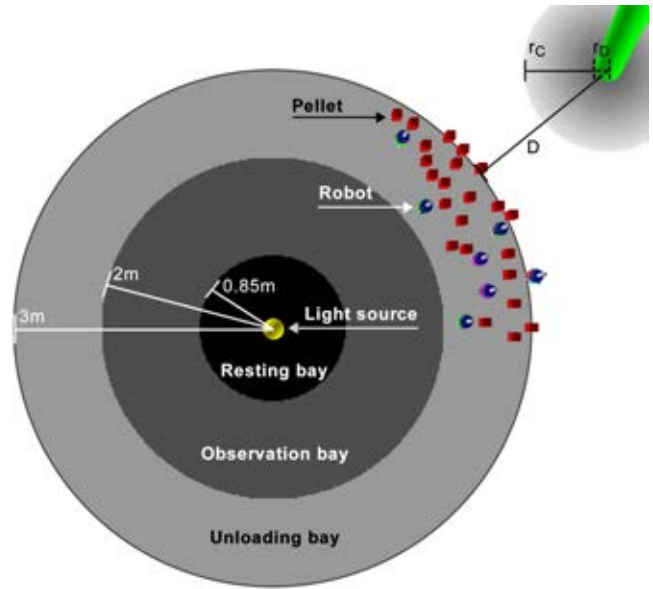
Figure 1: ARGoS simulation screenshot of the base and a deposit Dm away from the base edge. The base consists of a resting bay, an observation bay and an unloading bay. A light source is placed above the centre of the base to guide robot navigation. Pellets collected by the robots temporarily accumulate in the unloading bay, causing congestion.

ature, and in particular on the work of (Liu et al., 2007) and (Dai, 2009), where robots estimated the level of congestion in order to prevent unnecessary energy consumption. However, we apply the RTM in a novel scenario, where successful foragers *recruit* other robots to the worksites that they are exploiting (see also Pitonakova et al., 2014, 2016).

Furthermore, we provide novel insights into the role played by information flow in decentralised congestion estimation. In our non-social model, congestion is estimated by each robot individually, while in the social model, robots communicate their estimates to nearby robots in order to socially inhibit foraging.

Our approach to congestion estimation is inspired by the self-regulatory behaviour of honey bees foraging for nectar (Anderson and Ratnieks, 1999; Gregson et al., 2003). When nectar is abundant, foragers may bring more nectar into the hive than the nectar-receiving bees can cope with. In order to prevent unnecessary foraging, individual foraging bees evaluate how long it takes for their nectar to be unloaded. If unloading is taking too long, a forager will *tremble dance* around the nest, inhibiting other bees from recruiting and thus reducing the number of foragers. Our social RTM uses a similar principle.

Methods

Environment

All our experiments are performed in the ARGoS simulation environment which implements realistic 3D physics and robot models (Pinciroli et al., 2012). The simulation has continuous space and updates 10 times per simulated second. A circular base with a diameter of 3 metres is situated in the centre of the experimental arena. The base is divided into three sections (Figure 1): an interior circular *resting bay* with an annular *observation bay* around it and an annular *unloading bay* around that. A light source, placed above the centre of the base, is used by robots as a reference for navigation towards and away from the base centre (as in, e.g., Krieger and Billeter, 2000; Pini et al., 2013).

Cylindrical resource deposits with radius r_D are placed outside of the base, each containing an unlimited volume of resource. In order to enable robots close to a deposit to move towards it, a colour gradient with radius r_C is present on the floor around each deposit.

We explore two types of scenarios:

- HeapN: $N \leq 4$ deposits distributed evenly around the base at a distance $D = \{5, 7, 9\}$m from the base edge. These deposits represent large heaps of resource (e.g., mineral deposits), with $r_D = 0.5$m and $r_C = 3$m.

- ScatterN: $N \geq 10$ deposits randomly distributed between $D - 5$m and $D + 5$m from the base edge. These deposits are small (e.g., litter on a street), with $r_D = 0.1$m and $r_C = 1$m.

Robots

The simulated MarXbots (Bonani et al., 2010) are circular, differentially steered robots with a diameter of 0.17m that can reach a maximum speed of 5cm/s in our simulation. The robots use infra-red sensors for obstacle avoidance and communication, colour sensors for navigation towards nearby deposits, and a light sensor for phototaxis towards the base (see Pitonakova et al., 2016, for more details). The robots are modelled as finite-state machines and can implement three types of homogeneous swarm: *control swarm*, *non-social* self-regulators, and *social* self-regulators (Figure 2).

Control swarm robots exhibit basic foraging behaviour with no self-regulation. A robot starts with a random orientation and a random position in the observation bay as an *observer*, ready to receive and follow recruitment signals. An observer moves randomly across the observation bay and avoids traveling into the unloading and resting bays. At each time step an observer can become a *scout* with scouting probability $p(S) = 10^{-3}$. A scout leaves the base and uses Lévy movement (Reynolds and Rhodes, 2009) to search for a resource deposit within 20m of the base. The robot updates its estimated location relative to the base using path
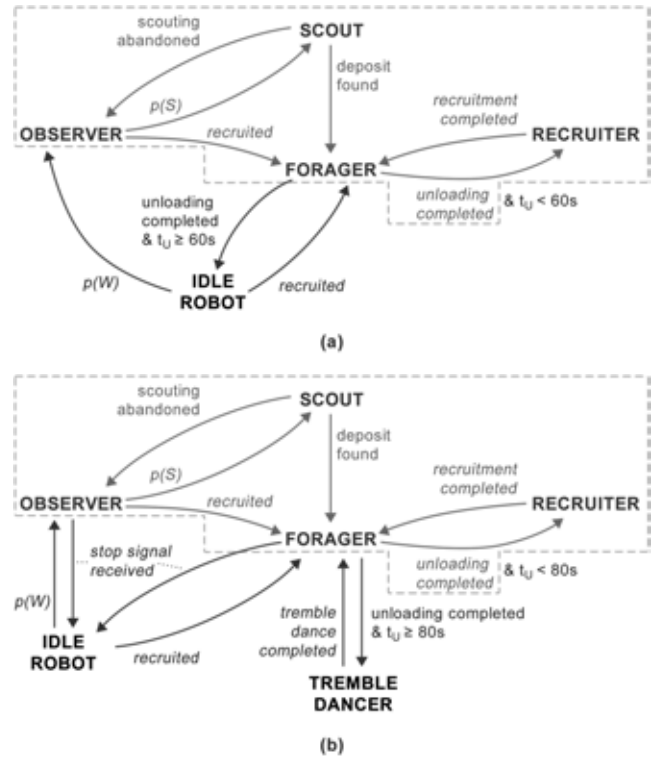
Figure 2: Finite state machine representation of the robot controller in swarms with (a) non-social self-regulation, and (b) social self-regulation. The behaviour of the control swarm controller is displayed in dashed boxes.

integration based on odometry at each time step (e.g., Lemmens et al., 2008; Gutiérrez et al., 2010). When a deposit is found, the robot loads one unit of volume of resource and returns back to the base utilising phototaxis, while keeping track of its position relative to the deposit using odometry. Odometry noise is not modelled. Any scout that cannot find a deposit within 600s returns to the base and becomes an observer.

A laden robot returning to the base drops off its load in the unloading bay in the form of four pellets of size 0.1m^3. The robots cannot push existing pellets around and thus have to avoid them in order to traverse the unloading bay. A new pellet can only be deposited when there is enough free space in front of the robot. Deposited pellets disappear from the simulation (representing their utilisation by a hypothetical unmodelled system of robots or human users) after a period of *unloading bay handling time*, t_H. When $t_H = 1$s, pellets disappear very quickly and do not cause congestion. By increasing the value of t_H, we can experiment with the level of congestion in the simulation, as more accumulated pellets make entering and leaving the base more difficult.

After depositing the pellets, the robot moves further into the base and performs recruitment for 120s, randomly mov-

ing across the base while avoiding re-entering the unloading bay. A recruiter advertises the fact that it has information about a deposit to all observers located within recruitment range of 0.6m. Deposit location is communicated to each observer in a one-to-one fashion by taking into account the local axes of the robots and their alignment relative to each other (Gutiérrez et al., 2010). The recruiter resumes foraging from the same deposit after it completes recruitment.

In *self-regulated swarms*, robots additionally measure their *pellet unloading time*, t_U, i.e., the time between entering the base and leaving the unloading bay. Robots in swarms with *non-social self-regulation* (Figure 2a) proceed to the resting bay and become *idle* after depositing pellets if congestion is severe (i.e., $t_U \geq 60$s). An idle robot consumes a negligible amount of energy (as in, e.g., Wawerla and Vaughan, 2010) and can be woken up and immediately recruited by a recruiter, i.e., by a robot that does not experience severe congestion. In order to avoid deadlocks, an idle robot can also wake up spontaneously with a waking probability $p(W) = 10^{-4}$.

Robots in swarms with *social self-regulation* (Figure 2b) do not become idle after experiencing severe congestion (i.e., when $t_U \geq 80$s), but become *tremble dancers* instead. A tremble dancer travels randomly across the observation bay and broadcasts *stop signals* to all robots within a 0.9m range for 120s, after which it leaves the base to resume foraging without recruiting. Stop signals inhibit foraging as any observer or forager that receives a stop signal moves to the resting bay and becomes idle. Stop signals also inhibit recruitment, as they cause any recruiter in range to cease recruiting and immediately leave the base to forage.

Analysis

We performed 50 simulation runs that lasted 4 simulated hours in Heap1, Heap4, Scatter10 and Scatter25 scenarios and compared the performance of all three swarm types using $N_R = 25$ and $N_R = 50$ robots. Since we are interested in efficient energy usage, we define a performance metric, *energy efficiency*, C, which represents the amount of energy a swarm spends in order to collect a unit of resource:

$$C = \frac{R}{E} \quad (1)$$

where R is the total amount of resource collected by the swarm and E is the total amount of energy expended by the swarm. It is assumed that an idle robot expends 0 units of energy per second and a robot in any other state expends 1 unit of energy per second. Since the control swarm robots are never idle, control swarms spend a total of $N_R \times (4 \times 60 \times 60) = N_R \times 14,400$ units of energy in each 4-hour experiment. We compare C values achieved by the two types of self-regulated swarms with that achieved by the control swarms in order to find out how advantageous self-regulation was in different scenarios.

We also analyse how much resource the swarms collected when energy availability was limited. During this analysis, it is assumed that all robots stop working when the swarm spends $N_R \times E'$ units of energy, where E' is the energy limit per robot. Energy limits may play a role for example in planet exploration, where robots might use a common solar-powered energy repository of a limited capacity.

Simulation Results

In the following sections, we compare the control swarms to each of the two kinds of the self-regulated swarms in terms of their energy efficiency, C, and the amount of resource they collected, R. We show that the self-regulated swarms can achieve better C in scenarios where pellets cause significant congestion. Furthermore, self-regulation leads to a higher amount of resource collected when the total energy available to the swarms is limited in such scenarios. We also discuss cases when self-regulation leads to performance deterioration, especially when social self-regulation is used.

Energy efficiency

In this section we report the performance (in terms of energy efficiency) of different swarm types in each of 48 scenarios: 2 swarm sizes (25 and 50) $\times$ 2 unloading bay handling times (5s and 20s) $\times$ 3 deposit distances (5m, 7m , and 9m) $\times$ 4 deposit distribution types (Heap1, Heap4, Scatter10 and Scatter25). In each case, we report the average performance of 50 self-regulated swarms relative to the average performance of 50 control swarms in the same scenario.

First we will summarise the performance of the control swarms, depicted in Figure 3. Their resource collection performance was more attenuated by congestion when the number of robots was large ($N_R = 50$) and when unloaded pellets did not disappear quickly from the unloading bay ($t_H = 20$s). Congestion was especially problematic in scenarios with a large number of deposits and when deposits were closer to the base. More severe performance deteriora-
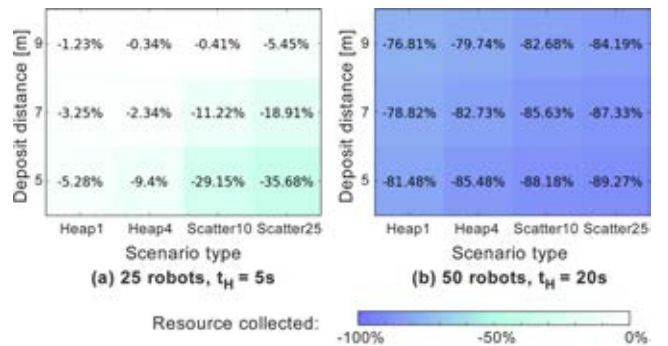
Figure 3: Resource collection performance of control swarms relative to experiments with no congestion (i.e., when $t_H = 1$s).
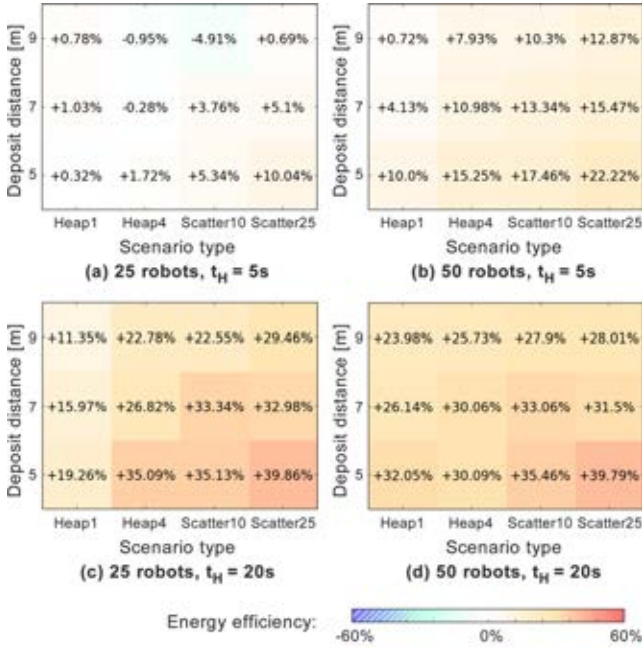
Figure 4: Performance of non-social self-regulated swarms relative to control swarms.
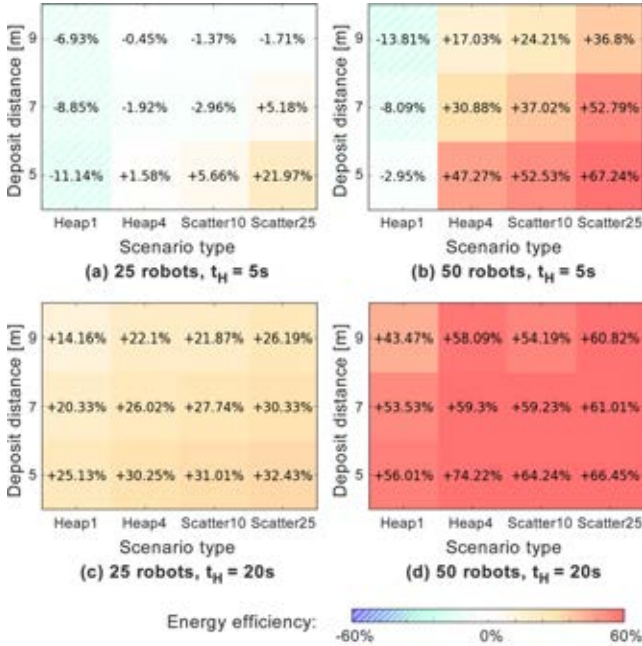
Figure 5: Performance of social self-regulated swarms relative to control swarms.

tion was measured in the Scatter scenarios, where multiple foraging locations were exploited at the same time, causing fast pellet accumulation around the whole unloading bay.

Evaluating the performance of *non-social* self-regulated swarms relative to that of control swarms (Figure 4) indicates that non-social self-regulators tended to enjoy more of an advantage when control swarms were more affected by congestion. Consequently, where congestion was very mild (e.g., a small number of robots foraging for a few heaped deposits distributed far from a base that handles unloaded deposits quickly), the performance of non-social swarms and control swarms was very similar ($\approx \pm 1\%$ difference), and control swarms even enjoyed a 5% advantage in the mildest Scatter10 scenario. However, where congestion tended to be severe (e.g., a large number of robots foraging for many scattered deposits distributed near to a base that handles unloaded deposits slowly), the performance of non-social swarms was considerably greater than that of control swarms (up to $\approx +40\%$ in the most extreme Scatter25 environments).

The performance of *social* self-regulatory swarms relative to the control swarms follows a similar but more complicated pattern (Figure 5). Again, where congestion tended to be severe, the performance of social swarms was better than that of control swarms (up to $\approx +67\%$ in the most extreme Scatter25 environments). Moreover, in these scenarios, social swarms did even better than non-social swarms, achieving an advantage over the control swarms that was often between 20% and 40% larger than that achieved by non-social swarms. Conversely, in scenarios where congestion was very mild, the performance of social swarms was *worse* than that of control swarms and non-social swarms by as much as -14%.

In general, there were two factors that affected the advantage of self-regulation: the amount of congestion in the base and the distribution of deposits in the environment. For instance, self-regulation was most advantageous in Scatter scenarios when deposits were close to the base (i.e., when the control swarms experienced high congestion due to short trips between the base and the deposits), and, more importantly, when foraging effort could be refocussed in a new direction once a particular part of the unloading bay became congested. On the other hand, self-regulation was not as effective in the Heap1 scenarios, where all resources were concentrated in a single location. Robots in the self-regulated swarms could still become idle when pellets accumulated, but recruitment could only take place again when the foraging robots measured a low unloading time, i.e., when enough of the pellets that had been unloaded in the part of the unloading bay between the deposit heap and the resting bay had disappeared. This was a particular problem for the swarms with social self-regulation, where the information about congestion spread quickly through the swarm, causing a majority of the robots to become idle. Unlike in non-social swarms, the number of foraging robots was often very low and it took the social swarms a long time to recover from inactivity.
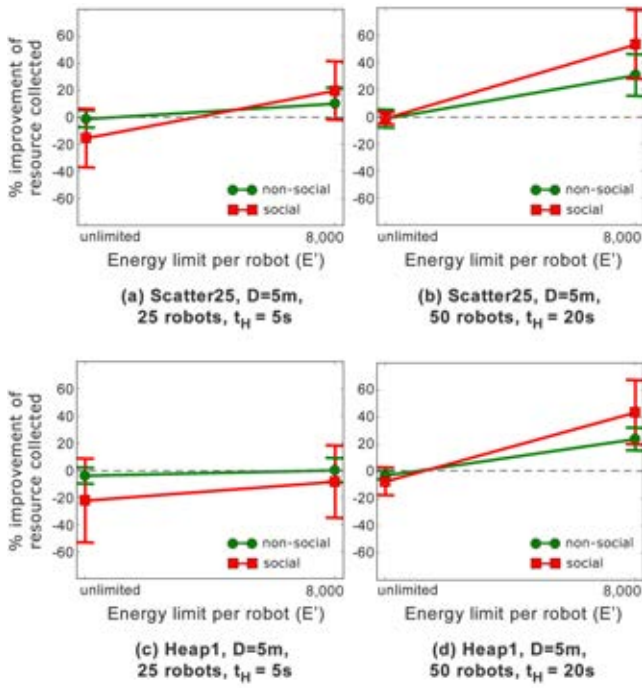
Figure 6: Resource collection performance of self-regulated swarms relative to control swarms under unlimited and limited energy conditions for scenarios with (a, c) mild congestion, and (b, d) severe congestion.

Resource collection

In this section we report the average performance (now in terms of total amount of resource collected) of the self-regulated swarms relative to the average performance of the control swarms. As in the previous section, we consider cases with mild congestion ($N_R = 25$, $t_H = 5s$) and severe congestion ($N_R = 50$, $t_H = 20s$). We first consider experiments where the total energy available to the swarms was unlimited. We then report on experiments with energy limitation, where robots ceased foraging as soon as their swarm had consumed $N_R \times E'$ units of energy, where E' was the energy limit per robot. Figure 6 depicts our results.

Although both types of self-regulated swarms were often more energy efficient than control swarms, when energy was unlimited they did not tend to collect more resource than the control swarms. Non-social self-regulators tended to collect a similar quantity to control swarms, whereas social self-regulators collected less resource than control swarms when congestion was mild (Figure 6a,c)

When swarm energy was limited, non-social self-regulators tended to either collect significantly more resource than control swarms (when congestion was severe), or roughly the same amount as control swarms (when congestions was mild). For instance, when E' was set to 8000 and when a large number of robots foraged from a base that

handled unloaded deposits slowly in the Scatter25 scenario, non-social swarms foraged up to 30% more resource relative to control swarms (Figure 6b). In experiments where congestion was mild, the advantage of non-social swarms was less pronounced. For instance, when a small number of robots foraged from a base that handled unloaded deposits quickly, non-social self-regulated swarms only collected up to 10% more resource than control swarms (Figure 6a).

Social self-regulation again led to more extreme variation in performance when the swarm energy was limited. When congestion was severe, social self-regulators tended to collect significantly more resource than either control swarms or non-social self-regulators (Figure 6b and 6d). Whereas when congestion was mild, they collected roughly the same amount as control swarms and social self-regulators (Figure 6a and 6c). For instance, in Scatter25, social-self-regulators collected approximately 55% more resource than the control swarms when $E' = 8000$ (Figure 6b). On the other hand, in Heap1, where the robots could not spread their foraging effort to other directions once a particular part of the unloading bay became congested, social self-regulators collected on average 10% *less* resource than the control swarms when congestion was mild (Figure 6c). In both cases, variation in performance within a scenario was higher for social swarms.

When the value of E' was higher or lower than 8000, the relative performance of both self-regulated swarms decreased linearly but was never lower than when the swarm energy was unlimited.

Discussion and Conclusions

We have shown that swarms can regulate their foraging activity effectively on the basis of locally perceived levels of congestion. The solution presented in this paper extends previous studies of the Response Threshold Model (RTM) (e.g., Liu et al., 2007; Dahl et al., 2009; Yang et al., 2009), applying it for the first time to foraging swarms that use recruitment and investigating the effect of information sharing during decentralised congestion estimation.

We compared three types of swarms: control swarms with no self-regulation, swarms with non-social self-regulation (where robots become idle when they directly sense severe congestion), and swarms with social self-regulation (where robots instruct their team mates to become idle when they detect severe congestion). The swarms were assessed across a number of experimental scenarios, where we varied the number of deposits (N_D), deposit distance from the base (D), the number of robots (N_R), and the time it took for accumulated material to be consumed at the base (t_H). We evaluated the performance of the swarms in terms of energy efficiency, C, and showed that C can be improved through self-regulation especially in environments where the collected material accumulates in the base quickly (because D is small, or N_R or t_H are large) or where the swarms

can exploit multiple foraging directions simultaneously (i.e., because N_D is large). Additionally, we demonstrated that self-regulated swarms collect more resources than control swarms when the energy supply available to the robots is limited.

There were notable differences in how swarms with non-social and social self-regulation performed in the various experimental scenarios. By comparison with control swarm behaviour, non-social self-regulation led to mediocre performance improvements or equivalent levels of performance in some scenarios. On the other hand, social self-regulation achieved large improvements over control swarms in scenarios where pellets accumulated quickly, but were also outperformed by control swarms in scenarios where congestion was not as severe or where all resources were concentrated in a single location. In our work on information flow in foraging swarms that use recruitment (Pitonakova et al., 2016), we argued that *fast* information flow can lead to pathological states of a whole swarm that prevent the swarm from responding to changes in the environment. Furthermore, we demonstrated that while swarms with fast information flow tend to perform extremely well in a limited number of environments but perform poorly in others, swarms with *slow* information flow tend to perform well across a broad spectrum of scenarios. In this paper, we extend this argument to scenarios involving congestion. Information flow was slower in swarms of non-social self-regulators which relied on their own local perception alone, and it was faster in swarms of social self-regulators which communicated information about congestion to one another. As was the case in (Pitonakova et al., 2016), slow information flow led to behaviour suitable for a larger number of experimental scenarios, while fast information flow caused more extreme variation in performance meaning it was only appropriate in a restricted set of scenarios.

Consequently, robots inspired by our social self-regulated swarms could be applied effectively in appropriate well-defined foraging or logistic tasks, for example to deliver items between various locations in warehouses and hospitals, or to collect crops. In these scenarios, the relevant task parameters (swarm size, processing time of collected items, etc.) are known upfront. However, if we were to employ robot swarms in an unknown or more variable environment, e.g., work sites on different planets or underwater, we would need to take into account the fact that while fast information flow can lead to beneficially fast response times, it can also cause significantly suboptimal performance under certain conditions. In such applications, self-regulation that is more subtle and occurs in a more localised fashion would be more suitable, not because of the ability of the swarms to perform work faster or more efficiently, but because such collective behaviour is more scalable. It might also be advantageous to create an adaptive algorithm, where robots alter their own self-regulatory behaviours (for example their willingness to exchange information with others, their waking up probability, etc.), in order to achieve a level of information flow within the swarm that varies dynamically in a way that is appropriate to the swarm's current environment.

References

Anderson, C. and Ratnieks, F. L. W. (1999). Worker allocation in insect societies: Coordination of nectar foragers and nectar receivers in honey bee (*Apis mellifera*) colonies. *Behavioral Ecology and Sociobiology*, 46(2):73–81.

Bonabeau, E., Sobkowski, A., Theraulaz, G., and Deneubourg, J.-L. (1997). Adaptive task allocation inspired by a model of division of labor in social insects. In Lundh, D., Olsson, B., and Narayanan, A., editors, *Biocomputing and Emergent Computation: Proceedings of BCEC97*, pages 36–45, London. World Scientific Publishing.

Bonani, M., Longchamp, V., Magnenat S., Philippe, R., Burnier, D., Roulet, G., Vaussard, F., Bleuler, H., and Mondada, F. (2010). The MarXbot, a miniature mobile robot opening new perspectives for the collective-robotic research. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010)*, pages 4187 – 4193, Piscataway, NJ. IEEE Press.

Cartade, P., Lenain, R., Thuilot, B., Benet, B., and Berducat, M. (2012). Motion control of a heterogeneous fleet of mobile robots: Formation control for achieving agriculture task. In *Proceedings of the International Conference on Agricultural Engineering (CIGR-AgEng '12)*.

Dahl, R. S., Mataric, M. J., and Sukhatme, G. S. (2009). Multi-robot task allocation through vacancy chain scheduling. *Robotics and Autonomous Systems*, 57:674–687.

Dai, H. (2009). Adaptive control in swarm robotic systems. *The Hilltop Review*, 3(1):54–67.

D'Andrea, R. (2012). Guest editorial: A revolution in the warehouse: A retrospective on Kiva Systems and the grand challenges ahead. *IEEE Transactions on Automation Science and Engineering*, 9(4):638–639.

Gregson, A. M., Hart, A. G., Holcombe, M., and Ratnieks, F. L. (2003). Partial nectar loads as a cause of multiple nectar transfer in the honey bee (*Apis mellifera*): A simulation model. *Journal of Theoretical Biology*, 222(1):1–8.

Gutiérrez, Á., Campo, A., Monasterio-Huelin, F., Magdalena, L., and Dorigo, M. (2010). Collective decision-making based on social odometry. *Neural Computing and Applications*, 19(6):807–823.

Hoff, N., Sagoff, A., Wood, R. J., and Nagpal, R. (2010). Two foraging algorithms for robot swarms using only local communication. In *IEEE International Conference on Robotics and Biomimetics (ROBIO 2010)*, pages 123–130, Piscataway, NJ. IEEE Press.

Jones, C. and Mataric, M. J. (2003). Adaptive division of labor in large-scale minimalist multi-robot systems. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, pages 1969 – 1974, vol. 2, Piscataway, NJ. IEEE Press.

Krieger, M. J. B. and Billeter, J.-B. (2000). The call of duty: Self-organised task allocation in a population of up to twelve mobile robots. *Robotics and Autonomous Systems*, 30(1-2):65–84.

Labella, T. H., Dorigo, M., and Deneubourg, J.-L. (2006). Division of labour in a group of robots inspired by ants' foraging behaviour. *ACM Transactions on Autonomous and Adaptive Systems*, 1(1):4–25.

Lein, A. and Vaughan, R. T. (2009). Adapting to non-uniform resource distributions in robotic swarm foraging through work-site relocation. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009)*, pages 601–606, Piscataway, NJ. IEEE Press.

Lemmens, N., de Jong, S., Tuyls, K., and Nowe, A. (2008). Bee behaviour in multi-agent systems. In Tuyls, K., Nowe, A., Guessoum, Z., and Kudenko, D., editors, *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*, volume 4865 of *Lecture Notes in Computer Science*, pages 145–156. Springer, Berlin.

Liu, W., Winfield, A. F., Sa, J., Chen, J., and Dou, L. (2007). Towards energy optimization: Emergent task allocation in a swarm of foraging robots. *Adaptive Behavior*, 15(3):289–305.

Mather, T. W. and Hsieh, M. A. (2012). Ensemble modeling and control for congestion management in automated warehouses. In *Proceedings of the 2012 IEEE International Conference on Automation Science and Engineering (CASE 2012)*, pages 390–395, Piscataway, NJ. IEEE Press.

Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Caro, G., Ducatelle, F., Birattari, M., Gambardella, L. M.,

and Dorigo, M. (2012). ARGoS: A modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence*, 6(4):271–295.

Pini, G., Brutschy, A., Pinciroli, C., Dorigo, M., and Birattari, M. (2013). Autonomous task partitioning in robot foraging: An approach based on cost estimation. *Adaptive Behavior*, 21(2):118–136.

Pitonakova, L., Crowder, R., and Bullock, S. (2014). Understanding the role of recruitment in collective robot foraging. In Lipson, H., Sayama, H., Rieffel, J., Risi, S., and Doursat, R., editors, *Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14)*, pages 264–271, Cambridge, MA. MIT Press.

Pitonakova, L., Crowder, R., and Bullock, S. (2016). Information flow principles for plasticity in foraging robot swarms. *Swarm Intelligence*, pages 1–31, DOI: 10.1007/s11721–016–0118–1.

Reynolds, A. M. and Rhodes, C. J. (2009). The Lévy flight paradigm: Random search patterns and mechanisms. *Ecology*, 90(4):877–887.

Shell, D. A. and Mataric, M. J. (2006). On foraging strategies for large-scale multi-robot systems. In *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2006)*, pages 2717 – 2723, Piscataway, NJ. IEEE Press.

Thiel, S., Häbe, D., and Block, M. (2009). Co-operative robot teams in a hospital environment. In *Proceedings of the 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS 2009)*, volume 2, pages 843–847, Piscataway, NJ. IEEE Press.

Vivaldini, K., Galdames, J. P. M., B., P. T., Sobral, R. M., Araujo, R. C., Becker, M., and Caurin, G. A. P. (2010). Automatic routing system for intelligent warehouses. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*, pages 93–98, Piscataway, NJ. IEEE Press.

Wawerla, J. and Vaughan, R. T. (2010). A fast and frugal method for team-task allocation in a multi-robot transportation system. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*, pages 1432–1437, Piscataway, NJ. IEEE Press.

Yang, Y., Zhou, C., and Tian, Y. (2009). Swarm robots task allocation based on response threshold model. In *Proceedings of the 4th International Conference on Autonomous Robots and Agents (ICARA 2009)*, pages 171–176, Piscataway, NJ. IEEE Press.

Increasing Reward in Biased Natural Selection Decreases Task Performance

Evert Haasdijk and Floor Eigenhuis
Vrije Universiteit Amsterdam, The Netherlands
e.haasdijk@vu.nl, f.m.eigenhuis@student.vu.nl

Abstract

This paper presents an investigation into a population of robots that evolves through embodied evolution — an evolutionary process that is not centrally controlled, but emerges from robot interactions just as natural evolution does. The robots select their partners randomly, without reference to any assessment of task performance, but the environment is biased to promote task behaviour by awarding additional lifetime to robots that pick up pucks. The experiments show that the robots do learn to pick up pucks in such a setting. Contrary to what one might expect, increasing the amount of additional lifetime awarded decreases task performance for all settings considered. Closer analysis shows that this decrease is in part due to the fact that the increased lifespan decreases the number of opportunities to spread a robot's genome, but that increasing the award level also negatively affects selection pressure when there is opportunity for robots to spread their genome. We conclude that higher rewards overly emphasise one aspect of robot behaviour and in doing so prevent evolution from exploring the behaviour space.

Introduction

Embodied evolution, or more generally on-line evolutionary robotics, subscribes to a view of collectives of robots that are released in uncharted, possibly changing, environments. The robots learn to operate in that environment, of which the particulars are unknown at design time, through evolution (Watson et al., 2002; Haasdijk et al., 2014). Just as in natural evolution, the process is not centrally orchestrated (in contrast to most evolutionary computing research), but evolution emerges from local interactions between the individuals: they survive to meet other individuals and between them decide to procreate, or not. For such systems to be of practical relevance, evolution must serve two purposes. Firstly, the robots must adapt to their environment so that they get the opportunity to procreate. The robots must, for instance, learn to move about to spread their genomes, or they must maintain their energy levels by regularly visiting charging stations. Secondly, the robots must perform some user-defined task: monitoring, patrolling, surveying, mining or harvesting are often considered in these kinds of scenarios (Bellingham and Rajan, 2007).

Although the environment in which robots operate does not specify any crisply defined objective function, it does indirectly circumscribe goals for the population of robots to survive and evolve. This implies environmental selection pressure that drives adaptation to the environment without reference to any user-defined task. There are two ways to augment this selection to also drive towards task performance: firstly, the robots can explicitly assess their task performance and add a second tier of selection (*mate selection*). Secondly, and more commonly in artificial life than in evolutionary robotics research, the environment can be modified so that individuals with appropriate behaviour receive an environmental advantage that increases their chances of reproductive success, e.g. by increasing their lifespan. In the latter case, task performance is not explicitly selected for, but the natural selection process is biased to promote task performance.

A well-known example of an evolutionary system which explicitly selects for task performance –and origin of the term *embodied evolution*– is that by Watson et al. (2002). Watson et al. added selection on the basis of an individual's prowess in a resource gathering task by varying the frequency at which robots would attempt to broadcast their genomes: better task performance increased this frequency. Thus, the task is explicitly defined and the robots assess their own task performance to drive selection. Haasdijk (2015) investigated the interplay of natural, environmental selection and explicit task-based selection, finding that explicit selection for task performance results in substantially higher selection pressure than that imposed indirectly by the environment, causing robots to prefer environmentally detrimental behaviour if that improves their odds in explicit selection.

This paper considers the alternative method of promoting task behaviour: the rules of the environment are modified so that robots that act appropriately benefit through the environment improving their chance of reproductive success. A good example of such *biased natural selection* (Bredeche and Montanier, 2012) can be seen in the Avida system by Adami and Brown (1994). Here, the individuals are programs in a virtual machine that can procreate by making

copies of themselves, just as in Ray's seminal Tierra system (1991). Selection for task performance is then added by increasing the clock rate for individuals that perform particular calculations well: this increases the speed at which they can generate copies and so bestows a reproductive advantage. The famous poisonous food experiment by Todd and Miller (1990) provides another example. In this experiment, the task is to collect resources, and virtual agents that gather ('eat') the right type of resource increase their lifespan, while eating the wrong type of resource decreases lifetime. The difference in lifetime implies a difference in reproductive success: agents that live longer (i.e., agents that eat 'healthy' plants) have more opportunity to procreate. Similar approaches use 'virtual energy' as an indicator of task performance to then increase lifespan and so give reproductive advantage, e.g., Elfwing et al. (2005) and Weel et al. (2012).

The experiments in this paper take a similar approach in a resource gathering scenario: simulated robots can collect pucks that extend their lifetime by certain fraction of their initial lifespan (the *reward level*). The resultant reproductive benefits promote task behaviour and so induce selection pressure towards puck gathering behaviour. The research question we consider in this paper is how the size of the reward influences the course of evolution.

Experimental Set Up

The experiments in this paper are based on the MONEE experiments by Haasdijk (2015) (from which a substantial part of this section is taken), which in turn extend Bredeche et al.'s mEDEA system (2012). In mEDEA, there is no objective to optimise: robots can exchange genetic material that encodes their controllers whenever they come within a certain maximum distance of each other (e.g., in range for infrared communication). A robot's controller is active for a fixed amount of time and when this time expires, it randomly selects one of the received genomes and activates a mutated copy of this genome as the new active controller. Thus, robot controllers procreate by transmitting their genome to eggs, and the more eggs a robot inseminates, the more chances it has for procreation. Because the transmission of genomes is continuous and at close range, the more a robot moves about the arena, the better its chances of producing offspring.

The robot controller lifecycle in our experiments consists of two phases: life and rebirth. The robot controllers have a limited, fixed, lifetime during which they perform their actions; moving about, foraging, et cetera (this lifetime may be extended by picking up pucks as described below). When their lifetime ends, they enter a rebirth phase and become 'eggs': stationary receptacles for genomes that are transmitted by passing live robots. The rebirth phase also lasts a fixed amount of time, and once this has passed, the egg randomly selects parents from the received genomes to create a new controller. The robot then reverts to the 'life' role with
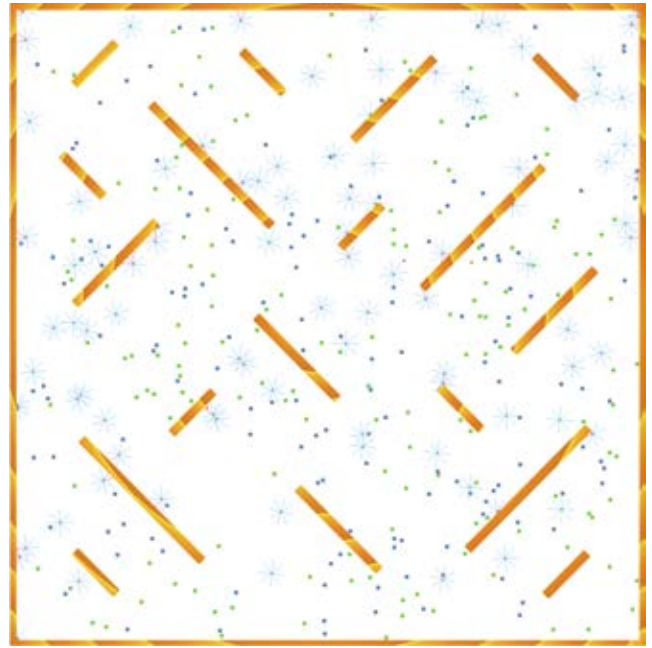
Figure 1: Experiment screenshot. Robots are shown as small circles with sensor beams indicated. Pucks are shown as small green squares (the blue squares show a second puck type that is disregarded in the experiments in this paper). The shaded orange rectangles indicate arena walls and obstacles.

this new controller. The resulting evolutionary process is essentially the same as that in mEDEA: the more ambulant a robot. the higher its reproductive success rate. In contrast to MONEE, there is no further selection criterion because the eggs select the parent genome randomly. In these experiments, the active phase lasted 2,000 time steps, and the egg phase 200 time steps.

Environment and Control

The experiments were conducted in a simple 2D simulator called RoboRobo (Bredeche et al., 2013), simulating 100 e-puck robots in an environment that contains obstacles and pucks. [1] The sides of the square arena are roughly 330 robot body lengths long (1,024 pixels in the simulator), and it contains a number of obstacles (see figure 1) and pucks. The pucks are spread throughout the arena, and they are immediately replaced in a random location when picked up. The robots move around the arena, spreading their genome as they encounter eggs and dying when their time has passed.

Robots can collect pucks simply by driving over them; picking up a puck extends the robot's lifetime by a certain amount. To detect pucks, the robots have 8 special sensors, laid out in the same manner as the standard e-puck infrared

---

[1]Code for the experiments and analysis scripts is available from `https://github.com/ci-group/monee.git`.

sensors: 6 face forward, 2 face to the rear. Each robot is controlled by a single-layer feed forward neural network which controls its left and right wheels. The inputs for the neural network are the robot's puck and obstacle sensors as well as two bias nodes (18 inputs in total). The robot's genome directly encodes the neural network's weights as an array of reals. The robots select a single parent from the received genomes and their current controller is discarded, so there is no crossover. Variation is applied by adding small gaussian perturbations ($\mathcal{N} = 0, \sigma = 0.1$) to the connection weights.

As mentioned, the robots alternate between periods of active puck gathering (life phase) and motionless genome reception (egg phase). The egg phase lasts 200 time steps, the life phase is initialised at 2,000 time steps, but to prevent synchronised cycles among the robots, we add a small random number to each robot's initial lifetime. This desynchronises switching between life and rebirth even though the runs start with all robots in sync at the first time-step of their lifetime.

Biased Natural Selection

MONEE extends mEDEA by having the robots select explicitly for task behaviour instead of selecting randomly from the received genomes. In the current set of experiments, however, this explicit selection is disabled. Instead, we provide reproductive advantage and so promote puck-gathering behaviour by rewarding robots for pucks they pick up: each puck yields an increase in lifetime. The amount of added lifetime is defined as the percentage of lifetime added. A reward level of 0.1, for example, means that a robot's lifetime is extended by 10%, i.e., by 200 time steps. We ran experiments with varying reward levels ranging from 0.05 to 0.8, with 50 replicate runs of for each setting.

The behaviour of collecting pucks and consequently living longer improves the reproductive chances of the robot: the more pucks a robot collects, the longer its lifespan; robots that are skilled at picking up pucks thus live longer and consequently have more opportunities to disperse their genomes by inseminating robots in egg state. Robots with less effective behaviour return to the dormant egg stage sooner, accelerating the distribution of genes that lead to puck collecting behaviour. Through repetition of this process of selection and gene dispersion the entire population's aptitude increases.

Quantifying Selection Pressure

Haasdijk et al. (2014) introduced a measure to quantify selection pressure that calculates the likelihood of random associations between behaviour and number of offspring in a population. This measure is based on the premise that an increasing level of certainty that the relation between behaviour and fecundity is not random indicates a higher selection pressure. If there were no selection pressure, the relationship between behaviour and fecundity would be random,

and contrariwise, if an individual's chances of generating offspring depend on its behaviour, the relationship is systemic. Fisher's exact test (Fisher, 1925) determines the certainty of nonrandom associations between the categories in a contingency table. We construct contingency tables by considering the distance covered, number of pucks collected and offspring count over the lifetime of the robots in the population. We split these individuals into classes with and without offspring and we split them along the median distance travelled or the median number of pucks collected during their lifetime to create two 2×2 contingency tables: one relating offspring and distance travelled and one relating offspring to number of pucks collected. The cells of the contingency tables contain the count of individuals for that cell (e.g., the number of individuals with offspring and below median distance travelled).

Fisher's exact test estimates the likelihood that the two classes in each contingency table (having offspring and above/below median distance travelled or pucks collected, respectively) are associated. The p-values resulting from these tests indicate the probability that there is no relationship between having offspring and having above- or below-median distance travelled or pucks collected. Thus, low p-values indicate high selection pressure and vice versa. Because the p-values are very small, we ease interpretation and comparison by reporting the log-likelihood multiplied by -1.

Results and Analysis

Figure 2 shows the amount of pucks collected over time for a range of reward levels. The reproductive benefit of picking up pucks clearly leads to puck-collecting behaviour: the robots pick up more pucks as evolution runs its course. However, the plot also clearly shows that the number of pucks collected is substantially lower for higher reward levels, and that the decrease in performance is systemic. This seems counter-intuitive: not only could one expect increased reward to provide a stronger incentive, but a longer, possibly even infinite, lifetime also implies that a robot can spend more time collecting pucks, because the egg stage in which a robot remains passive is deferred or eliminated. Figure 3 indicates that larger rewards do lead to longer lifetimes: it shows the number of deaths over time for different reward levels. Increasing the reward level decreases the number of deaths in the population per time interval, and with it the amount of time spent in the passive egg state (remember, each death initiates a 200 time steps egg phase). Thus, the robots spend more time in the active state where they can collect pucks.

To analyse the mechanisms that lead to this surprising effect of increasing reward, consider that when the reward is small, robots need to collect multiple pucks to increase their lifespan substantially. Robots that are not so skilled return to the egg stage quickly, providing receptacles for the genomes
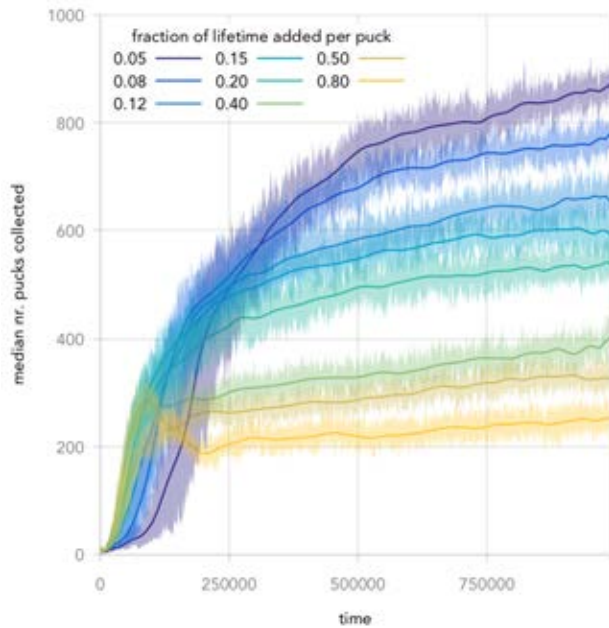
Figure 2: Number of pucks collected per 1,000 time steps vs time for different reward levels. The number of pucks collected initially increases more rapidly for higher reward levels, but lower reward levels ultimately lead to better task performance. The line plots were smoothed to emphasise the trends; the shaded area indicates the 95% confidence interval for the median.

Figure 3: Number of deaths per 1,000 time steps vs time for different reward levels. The number of deaths decreases as the population evolves to pick up pucks. The line plots were smoothed to emphasise the trends; the shaded area indicates the 95% confidence interval for the median.

of more skilled robots. However, the reproductive advantage from puck gathering for skilled robots decreases as the reward level increases. There are two possible causes: firstly, higher rewards allow relatively unskilled but lucky individuals to gain substantial lifetime extensions, allowing them to maintain and spread their genomes for much longer than would be the case for low reward levels. Secondly, fewer robots die at higher reward levels, leaving less opportunity for robots with relatively effective behaviour to pass on their genomes because there are fewer eggs available for insemination. Figure 3 shows that for reward levels 0.4 and higher, the number of deaths decreases very rapidly, and this stalls evolution because no robots become available for insemination.

Figures 4 and 5 provide more detail about the distribution of puck gathering behaviour and longevity over the populations. The plots show a positive correlation between reward and skewness: high rewards lead to populations where a few individuals collect large numbers of pucks and have very long lifetimes, while the majority of individuals perform at a much lower level. The majority gathers fewer pucks for high reward levels so that the median number of pucks collected per individual decreases as the reward increases. High rewards thus do lead to excellent task behaviour in a select

few individuals (high best fitness, in evolutionary algorithm terms). This, however, is not a relevant measure in an on-line setting such as this: here, the performance of each individual in the population counts, and with high rewards there is
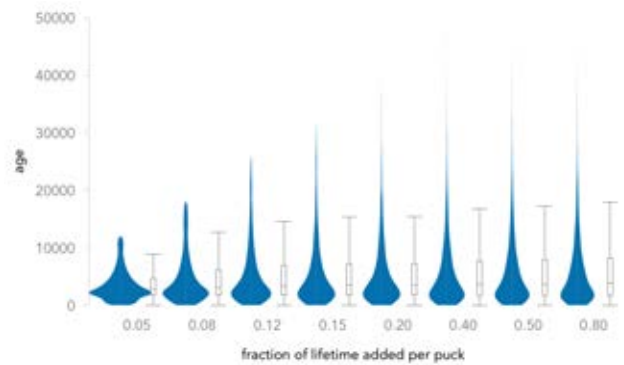
Figure 4: Distribution of age for different reward levels for individuals that died in the last 10,000 time steps of the runs. Each violin plot shows the probability density of age values for a reward level (note that the x axis is not to scale). Next to the violin plots are boxplots showing median and interquartile range. The plots show combined data over all repeats for each setting (individual runs show much the same pattern).
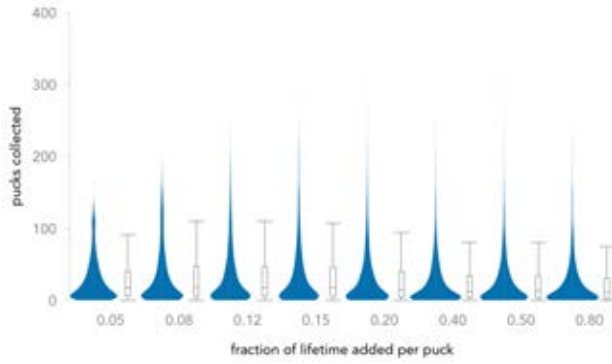
Figure 5: Distribution of number of pucks collected by individuals that died in the last 10,000 time steps of the runs for different reward levels. Each violin plot shows the probability density of values for a reward level (note that the x axis is not to scale). Next to the violin plots are boxplots showing median and inter-quartile range. The plots show combined data over all repeats for each setting (individual runs show much the same pattern).

such a preponderance of very poorly performing individuals that the overall performance of the population becomes ineffectual.

We perform a second series of experiments where more robots die also for higher reward settings by introducing 'random death'. In these experiments, robots can die at every time step with a probability of $1.5 \cdot 10^{-4}$, regardless of the amount of lifetime they actually should have left. Thus, eggs are more readily available for all reward levels. The results of these experiments are displayed in figure 6. The graph shows the number of pucks collected during the last 1,000 time steps at the end of each run for the normal condition (in blue) and the random death (in red) conditions. It shows the value for each run as a small circle and the median for each condition as a × symbol. The plot also shows the result of linear regression model of the relationship between the median number of pucks collected and the log of the reward level.

For lower reward levels the random death condition is not beneficial: the robots actually collect fewer pucks. At these lower reward levels, sufficient eggs are available all the time and more deaths are only counterproductive. Around reward level 0.2, the lines cross, and from this point onward the robots collect more pucks on average when robots die randomly. Note that the clear pattern of robots collecting fewer pucks for higher reward levels persists. This leads us to deduce that the decreasing task performance is also caused by large rewards decreasing selection pressure because even relatively poorly behaving robots receive substantial rewards.

The graphs in figure 7 show how selection pressure develops over time for a number of conditions. There are
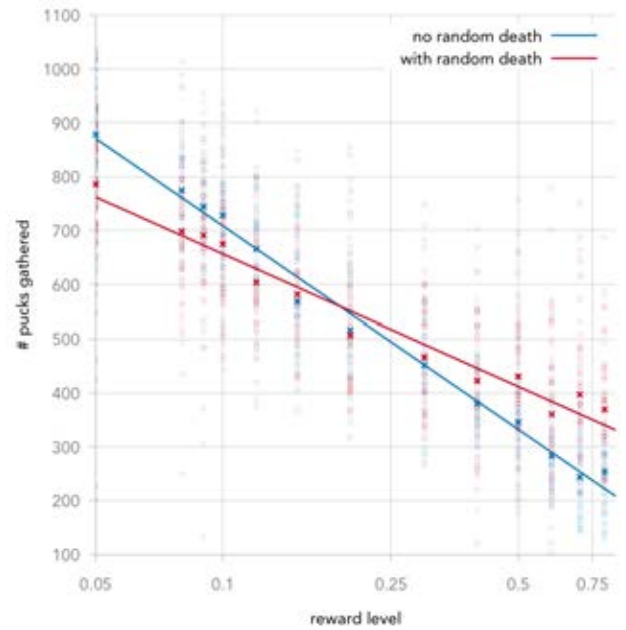
Figure 6: The number of pucks collected in the last 1,000 time steps vs the reward level with and without random death. Note that the horizontal axis is logarithmic. The circles represent individual runs, the × symbols indicate the medians for each reward level setting and the red and blue lines show the result of log linear regression through the medians. For reward levels lower than ca. 0.2, adding random death decreases task performance, for higher values this increases the number of pucks collected.

two components of behaviour that the environment selects for (albeit indirectly). Firstly, robots that move about the environment more have a greater chance of encountering eggs where they can leave copies of their genome. Secondly, picking up pucks increases lifetime and so increase the number of opportunities to spread the genome. Figs 7a and 7b show how the selection pressure deriving from these two components develops over time. For reward levels up to 0.2, selection pressure increases initially as the relevant behaviour spreads through the population. Then, as this behaviour becomes prevalent, the consequent relative reproductive benefit and with it the selection pressure decreases slowly to an intermediate level. Similar trends were also reported by Haasdijk et al. (2014). The selection pressure decreases as the reward level increases, supporting our deduction. For reward levels higher than 0.2, the picture is different: after an initial rise, selection pressure rapidly decreases, then fluctuates to settle at a low value. Selection pressure settles much sooner than it does for low reward levels.

Figure 7c compares selection pressures with and without the random death condition for a reward level of 0.2.
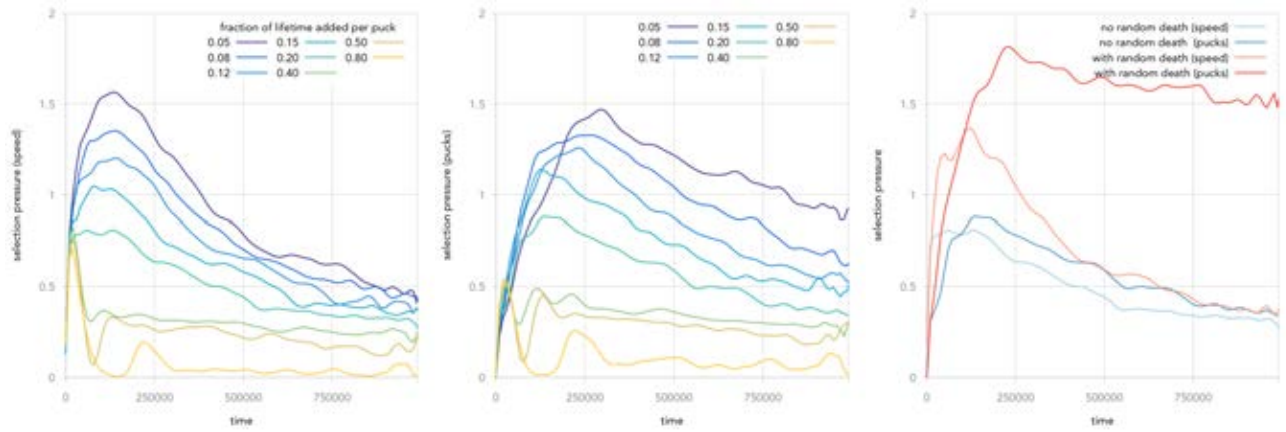
(a) Selection pressure from movement. (b) Selection pressure from number of pucks (c) Selection pressure with and without ran-
collected. dom death for reward level 0.2.

Figure 7: Development of selection pressure over time for different reward levels (7a and 7b) and comparing runs with and without random death. The selection pressure decreases with increasing reward level; for a reward level of 0.8, the selection pressure becomes minimal. Adding random death increases the selection pressure substantially, in particular the selection pressure resulting from puck collecting. Selection pressure is quantified as -1 times the log-likelihood of a random association between number of offspring and speed or number of pucks collected. The plots were smoothed to emphasise the trends.

It shows that increasing the number of deaths in this way substantially increases the selection pressure, in particular regarding the number of pucks selected. Interestingly, the selection pressure becomes much greater and stays much higher than it does without random death for lower reward levels, even though the number of pucks collected remains substantially lower.

The fluctuations for higher reward levels after the initial rise in selection pressure correspond with the fluctuations in Figs 2, 3 and 8. The fluctuations become increasingly pronounced and persistent as the reward level increases. In all cases, these fluctuations occur for reward levels higher than 0.2, which is also the tipping point at which adding random death starts improving puck gathering behaviour. This seems to indicate that this fluctuating trend is related to the lack of available eggs. Because of the high reward levels, there are few available eggs until robots that have poor behaviour but are lucky eventually do start dying off. At that point, there is a period where robots can spread their genome and robots with more appropriate behaviour enjoy some reproductive advantage, increasing selection pressure until there are few eggs available again. The cycle then repeats until the behaviour stabilises.

Higher reward levels do imply a faster increase in the number of pucks collected. The limited increase in lifespan for lower reward levels implies that moving at speed to be able to impregnate many eggs is an important component of reproductive success. This is also borne out by figure 7: initially, the selection pressure from movement is higher than from puck collection. As the median speed of the robots
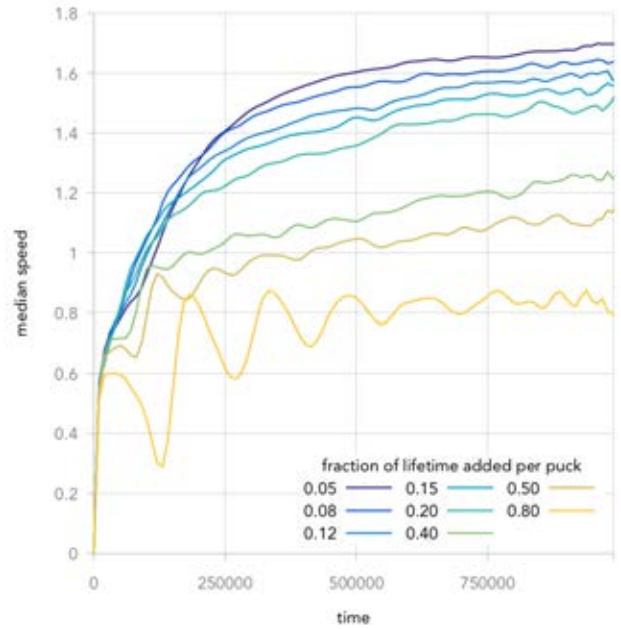
Figure 8: Median robot speed vs time for different reward levels. The development of speed shows the same trend as that of the number of pucks collected in 2: speed increases more rapidly for higher reward levels, but lower levels ultimately lead to higher speeds. The plots were smoothed to emphasise the trends.

319

rises (figure 8), collecting pucks becomes more advantageous as now it is possible to live substantially longer even with low reward levels. At higher reward levels, speed is not necessary to live for a very long time, and consequently still be able to spread one's genome. Figure 7 shows that speed matters only initially, and the benefits of collecting pucks appear sooner than for lower reward levels. Thus, it seems that high reward levels emphasise the puck collecting task too much to the detriment of movement, causing evolution to focus on the quick win of collecting a couple of pucks. More modest rewards cause evolution to explore the benefits of movement equally, and in the end this is beneficial for collecting pucks as well.

Conclusion

In this paper, we investigated an evolutionary system where robots randomly exchange genetic material in an environment that bestows benefits on robots that collect pucks. This benefit takes the form of an increase of the robot's lifespan. We showed that the population of robots does learn to pick up pucks as a result of the natural selection that is biased by the reward of additional lifetime without any further explicit selection.

When considering different reward levels, we saw that, unexpectedly, increasing the reward actually leads to poor performance, although the number of pucks collected initially rises more rapidly. High rewards emphasise one component of behaviour (in this case, collecting pucks) to the detriment of other components of behaviour (in this case, movement). Consequently, evolution focusses too much on the quick win of collecting pucks and neglects movement, becoming stuck in sub-optimal behaviour. If the reward is too big, there is no gradient for evolution to exploit: the benefit of mediocre or even poor performance is so big that there is little incentive to improve behaviour and evolution bogs down.

The most immediate conclusion, then, is: if the goal is for a population of robots to collect as many pucks as possible in a setting with biased natural selection, the reward for puck collecting should be minimal. However, it is tenuous to generalise this precise conclusion to other systems, e.g., where the benefit is awarded by increasing the speed of movement.

More generally put we showed that unduly rewarding behaviour in one aspect limits evolution's capability to explore the behaviour space, and the mechanisms that generate selection pressure must be balanced with care. This resonates with findings of research into explicitly selecting for diverse behaviour, for instance by Lehman and Stanley (2011).

It is a truism that evolution requires death, and in our experiments, the effects of overly focussing on puck collection are exacerbated by the lack of opportunities to spread a robot's genome because the increased lifespan reduces the number of available receptacles. This lack seems to become

particularly pressing when the reward level exceeds the tipping point of increasing lifespan by 20%.

The research presented here is part of an ongoing effort to research the interacting selection processes in embodied evolution, and further investigations, in particular to relate the findings here to experiments with multiple tasks and with explicit selection for task behaviour, are underway.

References

Adami, C. and Brown, C. (1994). Evolutionary learning in the 2d artificial life system avida. *Proceedings of Artificial Life IV*, pages 377–381.

Bellingham, J. G. and Rajan, K. (2007). Robotics in remote and hostile environments. *Science*, 318(5853):1098–1102.

Bredeche, N. and Montanier, J.-M. (2012). Environment-driven open-ended evolution with a population of autonomous robots. In *Evolution of Physical Systems, ALIFE XIII Workshop, online proceedings*, pages 7–14.

Bredeche, N., Montanier, J.-m., Liu, W., and Winfield, A. F. T. (2012). Environment-driven distributed evolutionary adaptation in a population of autonomous robotic agents. *Mathematical and Computer Modelling of Dynamical Systems*, 18(1):101–129.

Bredeche, N., Montanier, J.-M., Weel, B., and Haasdijk, E. (2013). Roborobo! a Fast Robot Simulator for Swarm and Collective Robotics. Technical report, arXiv.org.

Elfwing, S., Uchibe, E., Doya, K., and Christensen, H. I. (2005). Biologically Inspired Embodied Evolution of Survival. In *2005 IEEE Congress on Evolutionary Computation*, volume 3, pages 2210–2216. IEEE Press, Piscataway, NJ.

Fisher, R. A. (1925). Theory of Statistical Estimation. *Mathematical Proceedings of the Cambridge Philosophical Society*, 22(05):700–725.

Haasdijk, E. (2015). Combining Conflicting Environmental and Task Requirements in Evolutionary Robotics. In *Ninth IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, pages 131–137, Boston, MA, USA. IEEE Press.

Haasdijk, E., Bredeche, N., and Eiben, A. E. (2014). Combining Environment-Driven Adaptation and Task-Driven Optimisation in Evolutionary Robotics. *PLoS ONE*, 9(6):e98466.

Lehman, J. and Stanley, K. O. (2011). Abandoning objectives: Evolution through the search for novelty alone. *Evol. Comput.*, 19(2):189–223.

Ray, T. (1991). Is it alive or is it GA? *Proceedings of International Conference on Genetic Algorithms*.

Todd, P. M. and Miller, G. F. (1990). Exploring adaptive agency ii: simulating the evolution of associative learning. In *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*, pages 306–315, Cambridge, MA. The MIT Press.

Watson, R. A., Ficici, S. G., and Pollack, J. B. (2002). Embodied Evolution: Distributing an evolutionary algorithm in a population of robots. *Robotics and Autonomous Systems*, 39(1):1–18.

Weel, B., Hoogendoorn, M., and Eiben, A. E. (2012). Online evolution of controllers for aggregating swarm robots in changing environments. In Coello, C. A. C., Cutello, V., Deb, K., Forrest, S., Nicosia, G., and Pavone, M., editors, *Parallel Problem Solving from Nature - PPSN XII*, pages 245–254, Berlin, Heidelberg. Springer.

Performance Metrics of Collective Coordinated Motion in Flocks

Jorge L. Zapotecatl[1,2,3], Angélica Muñoz-Meléndez[2], and Carlos Gershenson[3,4]

[1]Posgrado en Ciencia e Ingeniería de la Computación,
Universidad Nacional Autónoma de México.
[2]Instituto Nacional de Astrofísica, Óptica y Electrónica
[3]Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas,
Universidad Nacional Autónoma de México.
[4] Centro de Ciencias de la Complejidad, UNAM, México.
email: jzapotecatl@gmail.com, munoz@inaoep.mx, cgg@unam.mx

Abstract

Measurements of coordinated motion in flocks are necessary to evaluate their performance. In this work, a set of quantitative metrics to evaluate the performance of the spatial features exhibited by flocks are introduced and applied to the well-known *boids* of Reynolds. Our metrics are based on quantitative indicators that have been used to evaluate fish schools. These indicators are revisited and extended as a set of three new metrics that can be used to evaluate and design flocks.

Keywords. metrics, flock, behaviors, boids

Introduction

An intriguing collective phenomenon of nature is caused by animals moving as a coordinated unit, for example a flock of birds or a fish school. This phenomenon has attracted interest in various scientific fields such as biology, artificial life and artificial intelligence distributed, because the phenomenon suggests an intelligent organization that transcends the abilities of each individual (Camazine et al., 2003).

The coordinated collective movement has important applications such as virtual reality and computer animation. The flocks models are often used to provide realistic-looking representations of flocks, schools or herds. For instance, they are used in the Half-Life video game to model the flying bird-like creatures or in the film Batman Returns to model bat swarms and armies of penguins (Bajec and Heppner, 2009).

In addition, the flocks models can be used for direct control and stabilization of teams of simple Unmanned Ground Vehicles (UGV) or Micro Aerial Vehicles (MAV) in swarm robotics (Min and Wang, 2011; Saska et al., 2014). Also, they can be used in applications that require the coordinated action of multiple autonomous individuals such as flying robots (drones) used in collective search, agricultural monitoring and event surveillance (Vsrhelyi et al., 2014).

Other interesting applications where flock models has been applied is the automatic programming of Internet multi-channel radio stations and for optimization tasks (Ibáñez et al., 2003; Cui and Shi, 2009).

The measurement and evaluation of the collective performance of autonomous agents is an open issue (Navarro and Matía, 2009). A lot of work remains to be done in the development of indicators that capture important aspects of the collective dynamics of groups of autonomous entities based, for instance, on the goal achievement, the formation of spatial patterns, and the exploitation of resources.

The tune of values for the achievement of coordinated flocks is far from being trivial, because the parameters that influence the behaviors of the members of flock are closely interrelated. Performance metrics are the criteria that determine success in the behavior of a system. Therefore it is necessary to design objective performance metrics that allow discern that flocks behave better.

In this work we propose new metrics in order to capture the global performance, in spatial terms, of the flocks such that they can be used as a benchmark. Our metrics are based in quantitative indicators that have been used to characterize spatial features exhibited by a fish school: extension, polarization and frequency of collisions (Huth and Wissel, 1992; Zheng et al., 2005). These measures are revisited and extended in a set of three new measures: consistency in expansion, consistency in polarization, and quality.

Related Work

A seminal work that aims at reproducing the flock phenomenon is the model of coordinated collective motion proposed by Reynolds (Reynolds, 1987), where individual entities, generically known as *boids*, achieve realistic behaviors by the application of a set of basic rules. This model is used in this work to applied our performance metrics. Therefore, we review in detail this model in the next section.

In the area of biology, Huth and Wissel (Huth and Wissel, 1992) present a simulation of a school of fish. The behavior of each fish are attraction, repulsion, parallel orientation and search. Polarization and extension are proposed as descriptive metrics of a school. The polarization reflects the degree of alignment of the agents headings, if fish belonging to the school are oriented in similar directions a school has a small polarization. The extension reflects the degree

of cohesion that has a school, that is, how far the fishes are between themselves.

In the work of Huepe and Aldana (Huepe and Aldana, 2008) are compared three simple models that reproduce qualitatively the emergent swarming behavior of bird flocks or fish schools, by using of the metrics: polarization, local density and nearest neighbor mean distance. While the polarization (standard order parameters use in the flocks) behave equivalently in all cases, the local density and nearest neighbor mean distance introduced provide a more detailed description that can clearly distinguish the properties of swarming behavior.

In the work of Navarro and Matía (Navarro and Matía, 2009) a set of metrics that measures the performance of collective movement of mobile robots is proposed and discussed. The different metrics proposed cover several aspects of the characteristics of the collective movement including: those related to the area and shape of the group; the movement; and the positioning and orientation of its members.

In the work of Bajec et al. (Bajec et al., 2007) it is presented an artificial animal construction framework that has been obtained as a generalization of the existing bird flocking models. A set of metrics that can measure and judge the flocking behavior of a group of *boids* is presented and the metrics are used in a series of controlled experiments to evaluate the flocking behavior of *boids*.

The *boids* model

This work is based on a model of coordinated collective motion proposed by Reynolds in his classic article (Reynolds, 1987), which is inspired by the movement of flocks and schools. Entities belonging to a formation (birds, fish, etc.) were called by Reynolds generically as *boids*. Each *boid* apply a simple set of steering behaviors such as cohesion, separation and alignment, that govern their movement. A flock is the result of the interaction of each *boid* with its neighbors.

The set B of n *boids* b_i involved in the flock is denoted by formula 1.

$$B = \{b_i, i = 1, 2, \ldots, n\} \qquad (1)$$

Each *boid* b_i has a position vector $\vec{p_i}(t)$ and a velocity vector $\vec{v_i}(t)$ that describe its motion in space in a time t. The force which adjusts the speed of the *boid* is typically an impulse generated by the same, so that impulse is limited in a scalar of maximum force denoted as f_m. In addition, *boids* are restricted to a maximum speed expressed as s_m. This speed limit is imposed by a gating of the velocity vector of the *boid*. In addition, the acceleration acquires a *boid* also depends on the inertia of the body expressed as a scalar quantity mass m_b.

The local space associated with each *boid* b_i is described by $\vec{f_i}(t)$, $\vec{s_i}(t)$ and $\vec{u_i}(t)$ which refer respectively to the

vectors "forward"(x axis), "side"(z axis) and "up"(y axis). Where each *boid* has a local view of its environment called "area of perception"related to a steering behavior. The area of perception is determined by a radius r and a angle θ (field of view) where only neighbors who are in the area of perception are selected for calculating certain steering behavior (see Figure 1).
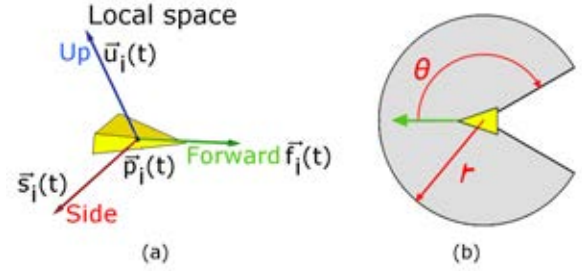
Figure 1: Figure (a) shows the local space of a *boid*. Figure (b) shows the area of perception of a *boid*.

The set of *boids* b_j perceived by the *boid* b_i is denoted P_i and is calculated as follows:

Step 1. Calculate distance, distance d_{ij} is determined from the *boid* b_i and *boid* b_j,

$$d_{ij} = \|\vec{p_j}(t) - \vec{p_i}(t)\| \qquad (2)$$

Step 2. Calculate angle, the angle θ_{ij} is determined between the vector "forward"of *boid* b_i and the unit vector in the direction of the position of the *boid* b_j,

$$\theta_{ij} = \frac{180}{\pi} \arccos \left[\vec{f_i}(t) \cdot \left[\frac{\vec{p_j}(t) - \vec{p_i}(t)}{\|\vec{p_j}(t) - \vec{p_i}(t)\|} \right]^T \right] \qquad (3)$$

Step 3. Calculate neighbors, determine if the distance d_{ij} is less than the radius r of *boid* b_i and angle θ_{ij} is within the angle θ of *boid* b_i, then the *boid* b_j belongs to its local neighborhood,

$$P_i = \{b_j \in B; \forall b_j : d_{ij} < r \wedge \theta_{ij} < \theta, j = 1, 2, \ldots, m\} \quad (4)$$

where m is the number of *boids* perceived by the *boid* b_i on its radius of steering behavior.

The steering behaviors of a *boid* b_i executed at time t are cohesion, $\vec{c_i}(t)$; separation, $\vec{s_i}(t)$; and alignment, $\vec{a_i}(t)$ where the areas of perception associated with these steering behaviors are determined by the radii and angles r_c, θ_c; r_s, θ_s; and r_a, θ_a respectively. In addition, the steering behaviors cohesion, separation and alignment are associated with the sets of *boids* b_j perceived C_i, S_i and A_i, respectively. The calculation of neighbors *boids* belonging to said sets is performed as in steps 1, 2 and 3 above.

The purpose of cohesion behavior is to move a *boid* towards the center of a group perceived within its neighborhood. If this steering behavior were uniquely applied, the formation would be gathered together in a region. The formula of cohesion, $\vec{c}_i(t)$ is expressed in (5), where m_c is the cardinality of the set C_i.

$$\vec{c}_i(t) = \left(\frac{1}{m_c} \sum_{\forall b_j \in C_i} \vec{p}_j(t) \right) - \vec{p}_i(t) \tag{5}$$

The purpose of separation behavior is to move a *boid* to avoid a collision with their neighbors and prevents agglomeration of formation. If only this behavior is applied, the formation dissipate. The formula of separation, $\vec{s}_i(t)$ is expressed in 6.

$$\vec{s}_i(t) = - \sum_{\forall b_j \in S_i} (\vec{p}_j(t) - \vec{p}_i(t)) \tag{6}$$

The purpose of alignment behavior is to move a *boid* in the same direction as their neighbors. The alignment behavior acts as a first heuristic to avoid collision, because when all *boids* of a formation move at the same velocity the risk of collision between them is reduced. The formula of alignment, $\vec{a}_i(t)$ is expressed in (7), where m_a is the cardinality of the set A_i.

$$\vec{a}_i(t) = \left(\frac{1}{m_a} \sum_{\forall b_j \in A_i} \vec{v}_j(t) \right) - \vec{v}_i(t) \tag{7}$$

When each *boid* b_i applies the basic behaviors of cohesion $\vec{c}_i(t)$, separation $\vec{s}_i(t)$, and alignment $\vec{a}_i(t)$ in combination, as result the formation is held together and moves coordinately. The formula for flocking, $\vec{fk}_i(t)$ is expressed in (8).

$$\vec{fk}_i(t) = \alpha \vec{c}_i(t) + \beta \vec{s}_i(t) + \delta \vec{a}_i(t) \tag{8}$$

Where each behavior is multiplied by the weights α, β, and δ and the range of values of these weights is $[0, \infty)$. The net force is determined by the force $\vec{fk}_i(t)$ and limited by the maximum force f_m.

$$\vec{F}_i(t) = \begin{cases} f_m \frac{\vec{fk}_i(t)}{\|\vec{fk}_i(t)\|} & \text{if } \left\| \vec{fk}_i(t) \right\| > f_m \\ \vec{fk}_i(t) & \text{otherwise} \end{cases} \tag{9}$$

Acceleration is equal to net force divided by the mass of the *boid* and is expressed in (10)

$$\vec{ac}_i(t) = \frac{1}{m_b} \vec{F}_i(t) \tag{10}$$

The new position of *boid* b_i at time t is calculated from its velocity $\vec{v}_i(t)$ and its previous position at time $t - \Delta t$, as presented in formulas (11) and (12).

$$\vec{v}_i(t) = \begin{cases} r_m \frac{\vec{v}_i(t-\Delta t)+a\vec{c}_i(t)}{\|\vec{v}_i(t-\Delta t)+a\vec{c}_i(t)\|} & \text{if } \|\vec{v}_i(t - \Delta t) + a\vec{c}_i(t)\| > s_m \\ \vec{v}_i(t - \Delta t) + a\vec{c}_i(t) & \text{otherwise} \end{cases} \tag{11}$$

$$\vec{p}_i(t) = \vec{p}_i(t - \Delta t) + \vec{v}_i(t) \tag{12}$$

In addition, it is necessary to limit the magnitude of change in orientation of the *boids* to prevent that *boids* look nervous due to abrupt changes and apply a behavior to avoid the walls. For the sake of space, the details of the implementation of these features is available online in the project code based on the library OpenSteer (`https://github.com/Zapotecatl/MetricsBoids`).

Performance Metrics

In this work the performance of the *boids* was evaluated in terms of extension, polarization and frequency of collision (Huth and Wissel, 1992)(Zheng et al., 2005) as quantitative indicators to characterize a formation, these metrics are revisited and extended. In this work we propose the metrics of consistency in extension, consistency in polarization, and quality, all of them are quantitative performance metrics about the collective dynamics of a formation.

The extension reflects the degree of cohesion of the flock and is determined by the average distance between one *boid* and the center of the flock. In this work the extension is given in terms of centimeters. The minimum value of the extension is $0cm$, a value that represents the situation where all *boids* are gathered together in one point in the environment. The maximum value of the extension depends on the shape and size of the pond, the number of *boids* and their distribution in the pond. The center of the flock, a value required to calculate its extension, is denoted as $cen(t)$, and it is expressed in (13).

$$\vec{cen}(t) = \frac{1}{n} \sum_{i=1}^{n} \vec{p}_i(t) \tag{13}$$

The extension of the flock at time t is denoted as $ext(t)$ and it is calculated by applying (14).

$$ext(t) = \frac{1}{n} \sum_{i=1}^{n} \|\vec{cen}(t) - \vec{p}_i(t)\| \tag{14}$$

The polarization is defined as the average of the angular deviation of each *boid* with respect to the average orientation of the entire group and expresses the degree of alignment of the *boids* headings, if *boid* belonging to the flock are oriented in similar directions a flock has a small polarization. Polarization holds a value in the range $[0°, 90°]$, where $0°$ represents a flock with an optimal parallel orientation where *boids* are perfectly aligned, and $90°$ represents a flock with the highest degree of "confusion" where *boids* are

completely unaligned. The average orientation of the group is denoted as $\vec{\mu}_p(t)$ and it is expressed in (15).

$$\vec{\mu}_p(t) = \frac{1}{n} \sum_{i=1}^{n} \vec{f}_i(t) \qquad (15)$$

The angle between the vectors $\vec{f}_i(t)$ and $\vec{\mu}_p(t)$ is represented by the symbol $\angle$ and expressed in (16), and the polarization of the flock at time t is denoted as $pol(t)$ and it is calculated from the values obtained in (16) as expressed in formula (17).

$$\angle(\vec{f}_i(t), \vec{\mu}_p(t)) = \frac{180}{\pi} \arccos\left(\vec{f}_i(t) \cdot \left[\frac{\vec{\mu}_p(t)}{\|\vec{\mu}_p(t)\|}\right]\right) \quad (16)$$

$$pol(t) = \frac{1}{n} \sum_{i=1}^{n} \angle(\vec{f}_i(t), \vec{\mu}_p(t)) \qquad (17)$$

The frequency of collision represents the degree of conflict among *boids* and is defined as the average of the number of *boids* in the collision state. The frequency of collision holds a value in the range $[0, 1]$, where 0 represents the ideal scenario where no collision occurs, whereas 1 represents the worst scenario where all agents collide with each other. When a *boid* b_i is in the collision state, the value of $c_i(t)$ is 1, otherwise its value is 0. The formula to calculate the collision state of *boid* b_i is expressed in (18), where r_b is the radius of the body of the *boid* .

$$c_i(t) = \begin{cases} 1 & \text{if } \|\vec{p}_j - \vec{p}_i\| < r_b, j = 1, ..., n, i \neq j \\ 0 & \text{otherwise} \end{cases}$$
$$(18)$$

The number of *boid* in collision state is expressed in (19).

$$col(t) = \sum_{i=1}^{n} c_i(t) \qquad (19)$$

The frequency of collision of the flock at time t is denoted $fcol(t)$, as expressed in (20).

$$fcol(t) = \frac{1}{n} col(t) \qquad (20)$$

Proposed Metrics

In order to capture the global performance of a flock considering the different aspects of its behavior that are represented by the combination of previous metrics, we propose three new metrics: 1) consistency in the extension, 2) consistency in the polarization, and 3) quality. These metrics express the relationship between the extension, polarization and the frequency of collision.

The **consistency in extension** aims at balancing the radius separation in a formation. Minimizing the values of extension and frequency of collision affects the flock since

the cohesion of groups is maintained by the combination of these metrics. The extension of a flock decreases when *boids* get close to each other, which is considered a positive action. However, if they get to close to each other, collisions might happen, which is considered a negative action. Therefore a careful balance of these metrics is necessary. The consistency in the extension evaluated at time t is denoted as $cns_{exp}(t)$ and it is expressed in (21).

$$cns_{ext}(t) = 1 - \frac{\sum_{i=1}^{m} \|\vec{cen}(t) - \vec{p}_i(t)\| + k \cdot col(t)}{max_e \cdot n} \quad (21)$$

where k is a distance penalty that holds a value in the range $[0, max_e]$ and m to denote the number of *boids* that do not collide with each other. The constant max_e represents the maximum extension that depends on the shape and size of the pond, the number of *boids* and their distribution in the pond. In this work we consider the value of max_e as half of the distance between two extreme points of the pond, which normalizes the second term in (21). Finally the complement is calculated. The consistency in the extension holds a value in the range $[0, 1]$, where 0 indicates the worst consistency and 1 the optimum consistency.

Expression (21) captures the necessary balance between the radius separating *boids* and the frequency of collision. Each *boid* adds a proportional contribution of its distance towards the center of the flock to the group consistency. However if the *boid* collides this contribution is overridden and summarized as a constant in the penalties.

if $k = max_e$ the integrity of the *boids* is highly weighted. A flock is evaluated with a value of $cns_{exp} = 0$ in a situation where all its members collide, whereas consistency in extension it approached to 1 when *boids* not collide. On the other hand, $cns_{exp} = 1$ can only be achieved if $k = 0$ and all the members of flock collide.

The **consistency in polarization** aims at balancing the orientation of a flock. Minimizing the values of polarization and frequency of collision affects the flock since the *boids*' goal is to achieve coordinated motion of uniformly aligned *boids* that do not collide to each other. The polarization of a flock decreases when *boids* are similarly oriented, which is considered a positive action when moving together. However the *boids* may need to change their orientation, but if such a change is not performed promptly *boids* may collide with each other. Therefore a careful balance of these measures is necessary. The consistency in the polarization evaluated at time t is denoted as $cns_{pol}(t)$ and it is expressed in (22).

$$cns_{pol}(t) = 1 - \frac{\sum_{i=1}^{m} \angle(\vec{f}_i(t), \vec{\mu}_p(t)) + \rho \cdot col(t)}{180 \cdot n} \quad (22)$$

where ρ is an angle penalty that holds a value in the range $[0°, 180°]$ and m to denote the number of *boids* that do not

collide with each other. The constant $180°$ represents the maximum polarization, which normalizes the second term in (22). Finally the complement is calculated. The consistency in the polarization holds a value in the range of $[0, 1]$, where 0 indicates the worst consistency and 1 the optimum consistency.

Expression (22) reflects, on its side, the necessary balance between the angular orientation of the *boids* and the frequency of collision.

Each *boid* adds a proportional contribution of its angular deviation with respect to the average group orientation. Similar to the consistency in extension, if the *boid* collides this contribution is overridden and summarized as a constant in the penalties to obtain the consistency in polarization.

Again, in this measure if $\rho = 180°$ the integrity of the *boids* is highly weighted. A value of $cns_{pol} = 0$ is achieved when the group is highly polarized and all its members collide. On the other hand, $cns_{pol} = 1$ when the group is perfectly aligned.

The **quality** aims at establishing a criterion to combine the results in the consistency in both extension and polarization, in such a way that we can evaluate the global performance of a flock. The quality is weighted by the factors σ and γ which determine, respectively, the influence of consistency in the extension and polarization on the final result. The quality is expressed in (23).

$$qlty(t) = \sigma \cdot cns_{ext}(t) + \gamma \cdot cns_{pol}(t) \qquad (23)$$

where $0 \leq \sigma \leq 1$, $0 \leq \gamma \leq 1$ and $\sigma + \gamma = 1$

Note that for each one of the previous metrics, both the punctual metric during the simulation steps, as well as the resulting metric in one simulation run are calculated. The later is calculated as the average of the corresponding measure during the simulation steps. For instance, the **total quality** is expressed in (24).

$$qlty = \frac{1}{l} \sum_{t=1}^{l} qlty(t) \qquad (24)$$

where the number of iterations performed during the simulation is denoted by l.

Results

The *boids* model was implemented and run on a computer with i5 processor and 16 GB of RAM equipped with a Nvidia Geforce GT 730 graphics board under Linux. The Open Source 3D Graphics Engine (OGRE) was used and an object-oriented programming approach was applied for our experiments. The library OpenSteer enabled us an accurate replication of *boids* to conduct experiments on fair and common basis (Reynolds, 1999), the source code of this project is available online (`https://github.com/Zapotecatl/MetricsBoids`).

The shape of the pond that we used in our simulations is a cuboid with dimensions in width, height and depth, denoted as W, H and D, respectively (see Figure 2). The *boids* were separately, initially distributed in random positions where the maximum distance that separates a *boid* from the central point of the pond is $50cm$, in such a way that they can be perceived by each other. The *boids* avoid colliding with the walls of the pond so periodic boundary conditions are not applied.
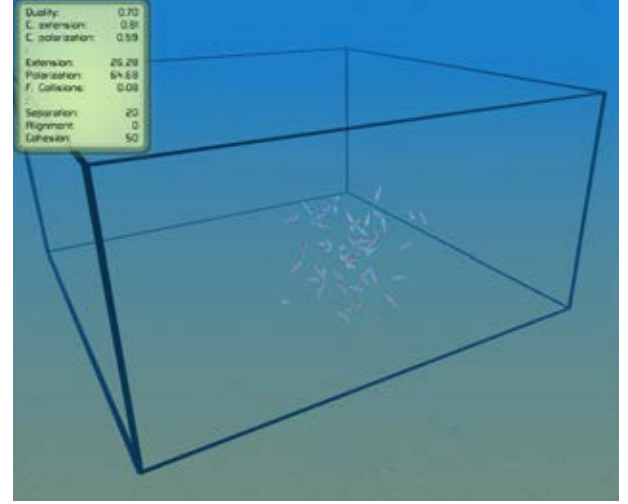
Figure 2: The shape of the pond used in our simulations is a cuboid. The *boids* can collide with the walls of the pond so periodic boundary conditions are not applied.

The 3D environment allows qualitatively visualize the result of our metrics (see Figure 3). The videos on how the groups look in relation to metrics are available online (`https://vimeo.com/user49682258/videos`).
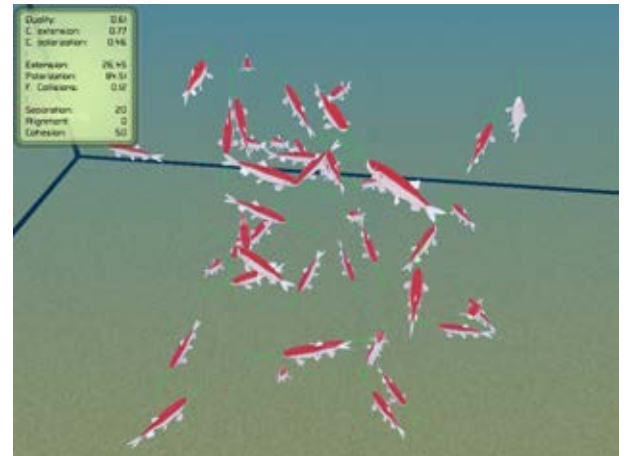
Figure 3: The figure shows a flock evaluated by the quality metric with a value of 0.61.

Extensive experiments were conducted to evaluate in

terms of previous metrics the performance of *boids* model. The experiments consisted in varying the radii r_s and r_a that determines the area of perception associated with the behaviors of separation and alignment in order to estimate the performance of the flocks under different configurations. Cohesion radius $r_c = 50cm$ for all experiments. For each experiment configuration the simulation was repeated 15 times for obtaining representative averages. The parameters used in our simulation are shown in the table 1.

Parameter	Value	Description
W	*300cm*	*Size of pond in width*
H	*150cm*	*Size of pond in height*
D	*300cm*	*Size of pond in depth*
l	*100000*	*Number of iterations*
n	*50*	*Number of boids*
r_b	*5cm*	*Radius of the body*
f_m	*50cm/s*	*Maximum force*
s_m	*25cm/s*	*Maximum speed*
m_b	*1*	*Mass*
r_c	*50cm*	*Cohesion radius*
r_s	*(0-50)cm*	*Separation radius*
r_a	*(0-50)cm*	*Alignment radius*
θ_c	*360°*	*Cohesion field of view*
θ_s	*360°*	*Separation field of view*
θ_a	*360°*	*Alignment field of view*
α	*25*	*Cohesion weight*
β	*25*	*Separation weight*
δ	*25*	*Alignment weight*
k	*225cm*	*Penalty distance*
ρ	*180°*	*Penalty angle*
σ	*0.5*	*C. extension weight*
γ	*0.5*	*C. polarization weight*

Table 1: Parameters used in our simulation with the *boids* model.

Figure 4 shows that the shorter the radius of separation and bigger the radius of alignment decreases the total extension. That is explained by the fact that, as long as the alignment radius is increased, the *boids* are motivated to establish the formation.

Figure 5 shows that when *boids* are assigned a big radius of alignment the total polarization is small. That is explained by the fact that, as long as the alignment radius is increased, the *boids* are motivated to go in the same direction.

Figure 6 shows that the bigger the separation radius, the smaller the total frequency of collision. That is explained by the fact that the *boids* are usually dispersed when they apply a big separation radius and for that the possibility of collision is reduced.

It is worth recalling that the quality metric provides a value that combines the results of the consistency in the extension and consistency in the polarization previously introduced. Figure 7 shows that the maximum value of total quality is in the order of $\cong 0.93$ resulting from different configurations with which we can conclude that, If the integrity of the members of the flock is highly weighted, the flock is evaluated with a **good quality under these scenarios**: (1)
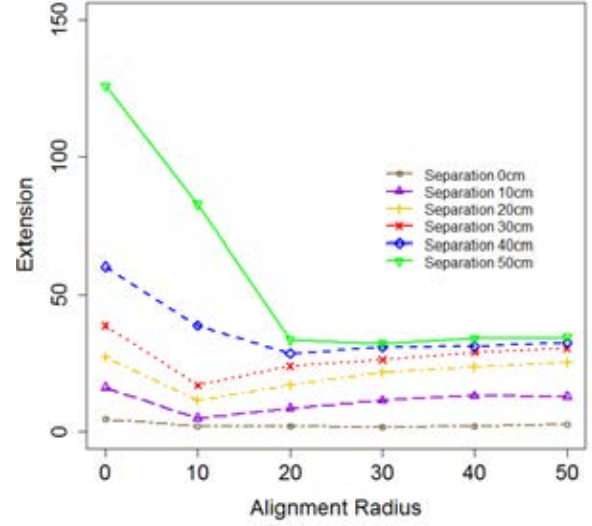
Figure 4: Extension, the minimum value of total extension is in the order of $1.64cm$ with $r_c = 50cm$, $r_s = 0cm$, and $r_a = 30cm$.
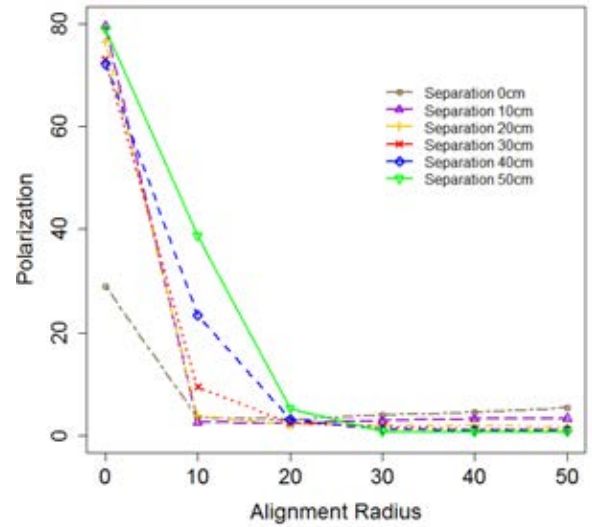
Figure 5: Polarization, the minimum value of total polarization is in the order of $0.79°$ with $r_c = 50cm$, $r_s = 50cm$, and $r_a = 50cm$.

The value of the separation radius is in a range that enables collision avoidance. This radius is in the range $[20, 50]$, (2) The value of the alignment radius is greater than or equal to the radius of separation, and (3) The value of the alignment radius is less than or equal to the radius of cohesion, as illustrated in Figure 7.

Figure 8 shows a configuration of radii of perception r_c,
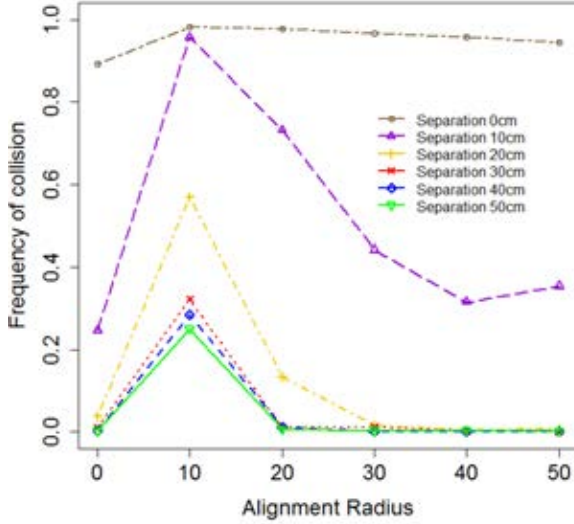
Figure 6: Frequency of collision, the minimum value of total frequency of collision is in the order of 0.0001 with $r_c = 50cm$, $r_s = 50cm$, and $r_a = 50cm$.
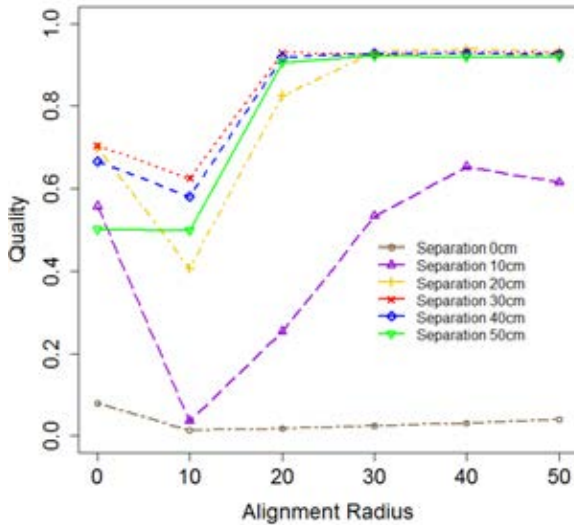
Figure 7: Quality, the maximum value of total quality is in the order of $\cong 0.93$ resulting from different configurations.

r_s and r_a of the flock that received the highest evaluation based on the quality metric.

Based on previous results we confirm that the combined application of the behaviors of cohesion and separation contribute to keep the flocking *boids* gathered without colliding. The alignment behavior on its side is useful to achieve coordinated motion of the *boids* and also avoid collisions.
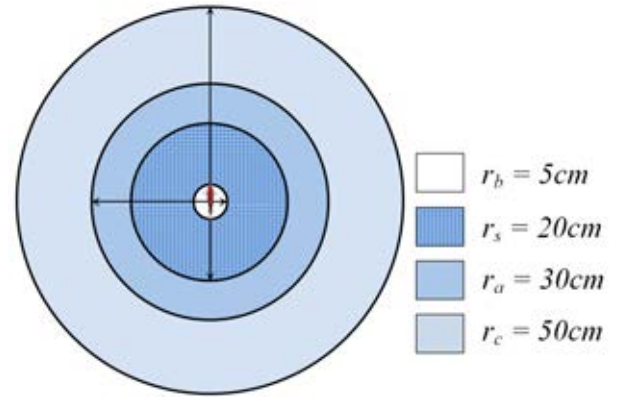
Figure 8: A configuration of areas of perception that received the highest evaluation based on the quality measure are: $r_c = 50cm$, $r_s = 20cm$ and $r_a = 30cm$.

The tune of values for the achievement of coordinated flocks is far from being trivial, because the parameters that influence the behaviors of the *boids* are closely interrelated. A small separation radius, for instance, brings *boids* to gather in compressed and "well-welded" groups, that are however quite prone to collide among them. In contrast, a big separation radius results in *boids* that are comfortably separated from each other and which are more prone to detaching from the flock.

Concluding Remarks and Future Work

The proposed metrics represent in such a way the dynamics of the system and they contribute to the analysis of the phenomenon of collective coordinated motion, in terms of generic global parameters and the relationships among these parameters. Therefore, since a flock refers to a group that shows a class of polarized, non-colliding and aggregate motion (Reynolds, 1987), the metrics allow to establish a general benchmark for the evaluation of models of the type flock; not only for *boids* model.

The performance metrics proposed in this research might be applied not only for qualifying some aspects of group behavior but also for tuning the behavior of groups of artificial agents. In effect, parameters such as radius of perception, field of view, and weight of behaviors for producing flocks able to reach the highest quality can be certainly estimated. Therefore, the metrics proposed in this work can be used for the design of flocks.

For example, is possible to apply a methodology that uses genetic algorithms for evolutionary development of a flock (Wood and Ackland, 2007; Olson et al., 2013). The fitness function apply quality metrics to punish flocks where members collide too and reward flocks where members do not collide. Resulting in the optimization of the behavior of agents so that they form flock and simultaneously not collide. The parameters to deliver the optimal results can be

applied in a animated flock or in autonomous flying robot (Virgh et al., 2014).

As future work, we are also working on the refinement of performance metrics that enable a more representative evaluation of the dynamics of the flock than current metrics. Note, for instance, that using current metrics a flock can be evaluated similarly when *boids* are fully dispersed that when they form dispersed subgroups in the environment. These situations should be properly identified.

Acknowledgements

References

Bajec, I. L. and Heppner, F. H. (2009). Organized flight in birds. *Animal Behaviour*, 78(4):777 – 789.

Bajec, I. L., Zimic, N., and Mraz, M. (2007). The computational beauty of flocking: boids revisited. *Mathematical and Computer Modelling of Dynamical Systems*, 13(4):331–347.

Camazine, S., Deneubourg, J., Franks, N. R., Sneyd, J., Theraulaz, G., and Bonabeau, E. (2003). *Self-Organization in Biological Systems*, chapter Fish Schooling, pages 167–188. Princeton University Press.

Cui, Z. and Shi, Z. (2009). Boid particle swarm optimisation. *International Journal of Innovative Computing and Applications*, 2(2):77 – 85.

Huepe, C. and Aldana, M. (2008). New tools for characterizing swarming systems: A comparison of minimal models. *Physica A*, 387(12):2809–2822.

Huth, A. and Wissel, C. (1992). The simulation of the movement of fish schools. *Journal of Theoretical Biology*, 156(3):365–385.

Ibáñez, J., Gómez-Skarmeta, A. F., and Blat, J. (2003). Dj-boids: Emergent collective behavior as multichannel radio station programming. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*, IUI '03, pages 248–250, New York, NY, USA. ACM.

Min, H. and Wang, Z. (2011). Design and analysis of group escape behavior for distributed autonomous mobile robots. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 6128–6135.

Navarro, I. and Matía, F. (2009). A proposal of a set of metrics for collective movement of robots. In *Workshop on Good Experimental Methodology in Robotics, Robotics Science and Systems*, pages 1–6.

Olson, R. S., Hintze, A., Dyer, F. C., Knoester, D. B., and Adami, C. (2013). Predator confusion is sufficient to evolve swarming behaviour. *Journal of The Royal Society Interface*, 10(85).

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 21(4):25–34.

Reynolds, C. W. (1999). Steering behaviors for autonomous characters. In *Game Developers Conference*, pages 763–782.

Saska, M., Vakula, J., and Preucil, L. (2014). Swarms of micro aerial vehicles stabilized under a visual relative localization. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 3570–3575.

Virgh, C., Vsrhelyi, G., Tarcai, N., Szrnyi, T., Somorjai, G., Nepusz, T., and Vicsek, T. (2014). Flocking algorithm for autonomous flying robots. *Bioinspiration Biomimetics*, 9(2):025012.

Vsrhelyi, G., Virgh, C., Somorjai, G., Tarcai, N., Szrnyi, T., Nepusz, T., and Vicsek, T. (2014). Outdoor flocking and formation flight with autonomous aerial robots. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 3866–3873.

Wood, A. J. and Ackland, G. J. (2007). Evolving the selfish herd: emergence of distinct aggregating strategies in an individual-based model. *Proceedings of the Royal Society of London B: Biological Sciences*, 274(1618):1637–1642.

Zheng, M., Kashimori, Y., Hoshino, O., Fujita, K., and Kambara, T. (2005). Behavior pattern (innate action) of individuals in fish schools generating efficient collective evasion from predation. *Journal of Theoretical Biology*, 235:153–67.

FSTaxis Algorithm: Bio-Inspired Emergent Gradient Taxis

Joshua Cherian Varughese[1,2], Ronald Thenius[1], Franz Wotawa [2] and Thomas Schmickl[1]

[1]Artificial Life Lab, Department of Zoology, Karl-Franzens-Universität Graz
[2]Institute for Software Technology, Techische Universität Graz
joshua.varughese@uni-graz.at

Abstract

This article presents a novel bio-inspired emergent gradient taxis principle for robot swarms. The underlying communication method was inspired by slime mold and fireflies. Nature showcases a number of simple organisms which can display complex behavior in various aspects of their lives such as signaling, foraging, mating etc. Such decentralized behaviors at the organism level gives rise to an emergent intelligence such as in bees, slime mold, fireflies etc. Chemo taxis and photo taxis are known to be abilities exhibited by simple organisms without elaborate sensory and actuation capabilities. Our novel algorithm combines the underlying principles of slime mold and fireflies to achieve gradient taxis purely based on neighbor-to-neighbor communication. In this article, we present a model of the algorithm and test the algorithm in a multiagent simulation environment.

Introduction

Swarm robotics research has shown that complex problems can be solved in unconventional ways (Schmickl and Hamann, 2011)(Schmickl et al., 2008)(Bjerknes et al., 2007). Swarm intelligence uses simple agents following simple rules to solve complex problems. Without the advantages of swarm intelligence, such complex problems could only be solved at a relatively high amount of computational and economic resources. In project subCULTron (subCULTron, 2015), we aim to develop an underwater society of learning, adapting, self-sustaining robots which can be used for various applications. Given the challenges of robotic systems underwater such as limited availability of classical communication systems, limited mobility etc., there need to be stable but simple solutions. In nature, there are many such behaviors such as chemical gradient taxis (chemo taxis) in slime mold (Webb, 1998) or thermal taxis in bees (Grodzicki and Caputa, 2005). It is challenging that nature solves these problems with the minimum resources. Gradient taxis is an example of an algorithm that will be used in subCULTron for gradient ascent or descent. Like many swarm researchers in the past (Schmickl and Crailsheim, 2007) (Nakagaki, 2001), we draw inspiration from nature to solve the gradient ascent for robots in subCULTron.

Many studies in the past have been done on application of swarm based algorithms in robotics. Swarm behavior is based on decentralized underlying rules at the organism level giving rise to an emergent intelligence such as in bees (Bodi et al., 2015) (Kernbach et al., 2009), slime mold (Nakagaki et al., 2004), fireflies (Buck and Buck, 1966) etc. The aggregation and maze solving capabilities of slime mold have been extensively researched (Nakagaki, 2001). Slime mold swarms have been used to solve mazes (Nakagaki et al., 2000) and this capability has been tested in real world scenarios such as the Tokyo rail transport system (Nakagaki et al., 2004). Similarly, fireflies and their ability of phase synchronized pinging has been of interest to the research world for a long period of time (Buck and Buck, 1966). Many of such behaviors has found applications in engineering and computer science. For example, Yang (2009) has taken inspiration from fireflies to solve multi-modal optimization problems and such efforts have shown promising results.

In this paper, we present a novel method which combines communication behavior from slime mold as well as fireflies for gradient ascent. Various kinds of gradient functions to a swarm of agents for testing the algorithm. The following sections will first formulate the algorithm, describe the testing scenarios, methods and discuss the results in that order. Our algorithm is a fine example of how emergent solutions can be used for tasks without using complex computation, large amount of memory and with minimum power consumption. There exist classical approaches for gradient ascent and multi-modal optimization such as the steepest gradient descent (Arfken, 1985), Particle Swarm Optimization (Kennedy and Eberhart, 1995) etc. Another possible solution for gradient related problems is Simultaneous Localization And Mapping (SLAM)(Bazeille and Filliat, 2011) where an agent constructs a map of an unknown environment while simultaneously keeping track of its own location and the gradient value. Then a global observer is able to guide the agent finally to the maximum or minimum gradient value. However, these solutions require high amount of computation power which is economically and computation-

wise unfeasible for a swarm of underwater robots such as that in subCULTron.

The objectives of this paper are as follows:

1. Formulate a novel emergent gradient taxis algorithm inspired by slime mold and fireflies.

2. Test the algorithm in various types of gradients.

3. Validate the algorithm by investigating boundary conditions.

4. Discuss strengths and weaknesses of the algorithm and compare it with an existing emergent gradient taxis algorithm (SwarmTaxis Bjerknes et al. (2007)).

Biological Inspiration

As previously mentioned, the presented algorithm takes inspiration from slime mold and fireflies. Therefore, it is of merit to look into the aspects of their biological behavior that we draw inspiration from. This section briefly discusses communication strategies used by slime mold and fireflies which will support the formulation of the algorithm.

Slime Mold

Slime mold (*Dictyostelium Discoideum*), is a free living diploid life form. It has been subject of much study in the past due to its ability to survive harsh environments by taking advantage of group behavior. Slime mold, during its life cycle, aggregates with other cells to from a multicellular organism. Each organism starts its life as a unicellular amoeba, but during starvation they aggregate to form a multi cellular fruiting body. Chisholm and Firtel (2004) divide its life cycle as follows: Aggregation, Streaming, Slug, Culmination, Fruiting body. The algorithm presented in this paper will deal mainly with the aggregation phase and hence will look it in detail.

When there are ample food sources, cells grow and divide in a matter of three to four hours (Siegert and Weijer, 1992). On the other hand, if there is a scarcity of food, significant cooperation between the cells begin, thereby kicking off the aggregation phase. During this time, some cells (centers) release Cyclic Adenosine Monophosphate (cAMP) into the environment to induce a chemical concentration spike around them (Siegert and Weijer, 1992). cAMP concentration diffuses very quickly into the environment and therefore the chemical spike is short-lived. This chemical spike enables these centers to recruit other cells present around them. When surrounding cells perceive this chemical signal, they move towards areas of high cAMP concentration and release cAMP themselves, thereby relaying the signal. This in turn, attracts other cells towards the centers. One cell is able to release cAMP at an interval of 12-15 seconds (Alcantara and Monk, 1974); during this interval, individual cells are insensitive to cAMP pulses. This interval can

be understood as the refractory phase of the amoeba. The signal relaying mechanism described above forms the basis for spatiotemporal patterns known as scroll waves (Siegert and Weijer, 1992). The refractory phase is responsible for these scroll waves as it prevents the signaling organism from perceiving its own signal that was relayed earlier. The emergence of scroll waves enable the amoeba to move towards the recruiting centers for successful aggregation.

Fireflies

Fireflies are a family of insects that are capable of producing bio-luminescence to attract a mate or a prey (Buck and Buck, 1966). The brightness of the bio-luminescent light depends on the amount of luciferin, a light emitting compound, available with the firefly (de Oliveira et al., 2011). Bio-luminescence of various families of fireflies has been a subject of elaborate study in the past (Buck and Buck, 1966). Apart from being able to blink, fireflies are known to behave in cooperation with other fireflies. It is a spectacular sight to see thousands of fireflies light up in unison on a tree lighting it up entirely. This uniform blinking is in order for the swarm to have higher chance of attracting mates or prey (Buck and Buck, 1966). The luminescence of the blinking swarm is much more than that of an individual firefly.

Such synchronicity is a result of a simple mechanism by which initially the individual fireflies blink randomly and when it perceives a blink in its surrounding, it blinks again and then resets its own frequency to match the other (Camazine et al., 2001). It takes time for the fireflies to achieve complete synchronization. This is analogous to a phase couple oscillator which adjusts its phase to match it to that of the faster one in the vicinity. This trait emerges into a pseudo synchronized blinking pattern while the frequency of blinking will be influenced by the fastest blinking insect.

FSTaxis algorithm

As per the objectives listed in the introduction, a novel gradient taxis algorithm is hereby presented, namely, the Firefly Slime mold Taxis(FSTaxis) algorithm. As its name suggests, this algorithm draws inspiration from biological systems introduced in the Section "Bio Inspiration". The FSTaxis algorithm makes use of the communication strategy of slime mold and the phase coupled oscillation aspect in fireflies. The behavior of agents in the FSTaxis algorithm can be broadly classified into *Ping behavior* and *Motion behavior*. The following sections will explain the working of these behavior modes. The sequential flow of instructions of the FSTaxis algorithm can be found on the following page. Hereafter, a "ping" is referred to the single bit communication which each agent broadcasts. The agents are equipped with sensors to determine the direction of incoming pings and the environmental factor of interest value at its own location.
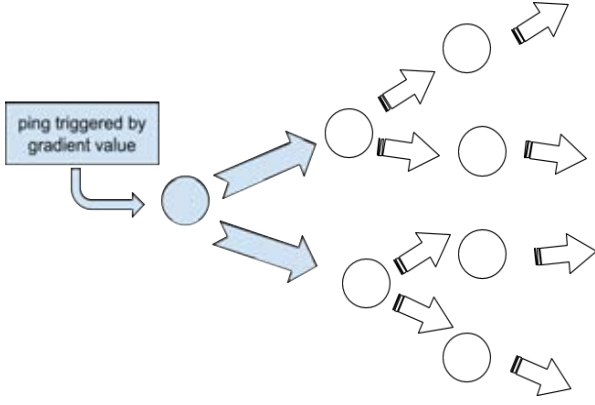
Figure 1: When the intrinsic cycle length of each agent counts out, a ping is broadcasted, surrounding agents captures the ping and relays it on. The blue circles show original agent whose internal clock triggered. The white agents in the figure relay the pings

.

The pings can be perceived by other robots within a very limited sensor radius, s_r. Also, the agents are able to move around in the environment with limited speed v_a.

Ping behavior

Each agent has three communication states: "pinging", "refractory" and "inactive" as shown in the state transition diagram Figure 2. By default all agents are set to inactive mode and each of them have an internal countdown timer whose initial value is associated with its position in the environmental gradient. In the inactive mode, the agent only checks for incoming pings. When an agent receives a ping, it broadcasts a ping for a period of time, say t_p. During t_p, the agent is said to be pinging and after t_p, the agent enters the refractory mode. During refractory time, the agent ignores all incoming pings. After the refractory time t_r, the agent sets itself back to inactive mode. The cycle continues if the agent receives another ping.

Each agent has an inherent cycle time determined by the environmental gradient at its position. As shown in Figure 1, if the internal timer of any agent counts to zero before a ping is received, the agents broadcast a ping and sets its own ping frequency, f_p, by associating it with the gradient value at its position, g_p. That "original" ping is further relayed by the neighboring agents as per the ping behavior explained above. The agent that triggers the original ping (the agent whose f_p counted to zero) is referred to as the "leader" in the upcoming sections of this paper.

In order to provide scaling of pinging frequencies to meaningful values, two preset maximum and minimum are selected for the gradient under consideration. Let these values be g_{max} and g_{min}. Equation 1 shows the relation between ping frequency of agents and inherent cycle time of agents. The selection of constants, α and ω, are dependent

upon t_p, t_r and the boundary values of the gradient under consideration. Here, α and ω should be selected so that the agents continue pinging in the entire range of gradient values possible. For example, if t_p is equal to f_p for any agent, it will continuously ping without ever entering refractory phase. Therefore, it is necessary that α and ω are adjusted to scale ping frequencies to meaningful values. In this paper, α and ω are selected only to demonstrate the gradient ascent ability of the FSTaxis algorithm. Since it does not depend upon the type of gradient the values of the constants will be the same throughout this paper as shown in Table 1.

$$f_p = \alpha + \frac{((g_p - g_{min})}{(g_{max} - g_{min})} * \omega \qquad (1)$$

Motion behavior

Motion behavior in the agents is overseen by ping behavior. An agent in inactive mode does not move. As shown in Figure 2, motion is initiated in the active mode. When any agent receives a ping it sets itself to active mode, sets its own heading towards the received ping and starts moving to cover a fixed distance, β at velocity v_a. A ping can only be perceived within the limited sensor range, s_r, of the robot, therefore it limits the number of agents that are able to affect any particular agent. In the scenario described above, it is possible that each agent receives multiple pings from different directions, h_n, where n is the number of agents pinging. In such as case, the agent will calculate the mean heading, h_{mean}, and set its heading towards this mean.

If an agent's internal clock triggers, an "original" ping based on the environmental value is broadcast; then, this agent labels itself the leader and does not move in that particular cycle.

When a swarm of agents execute the FSTaxis algorithm as per description above, scroll waves of pings similar to that in slime mold (as mentioned in section "Slime mold") propagates through the swarm. Since the internal timer of the robot at highest gradient value will count to zero first, the direction of the wave will be from the higher to lower gradients. During their inactive cycles, the agents will move towards the mean direction of incoming pings. Since the "leader" broadcasts the original ping and does not move, the agents will gather around the leader. When the agents are in their new position, their internal clock takes the values of the environmental value (gradient value). Whichever agent's internal clock triggers first becomes the leader and the swarm then gathers around this agent. This repeated choosing of leaders and gathering around the leader will draw the swarm towards areas with higher gradient value and in essence emerges into a gradient ascent.

Method

To demonstrate the gradient ascent capability, a linear gradient, a hyper ellipsoid gradient and noisy variants of these

Algorithm 1 The FSTaxis algorithm

repeat
    procedure PING BEHAVIOR(t_p, t_r,v_a,t_f)
        for all agents **do**
            if pingmode = refractory mode **then**
                count down t_r
                if $t_r = 0$ **then**
                    set state $\leftarrow$ inactive mode
                    set $t_f \leftarrow 1/f_p$ - $equation 1$
                    set leader status $\leftarrow$ "OFF"
                end if
            end if
            if pingmode = active state **then**
                count down t_p
                if $t_p = 0$ **then**
                    set state $\leftarrow$ refractory mode
                end if
            end if
            if pingmode = inactive mode **then**
                if any ping received? **then**
                    set state $\leftarrow$ active mode
                    set move agent $\leftarrow$ "ON"
                end if
            end if
            count down t_f
            if $t_f = 0$ **then**
                set state $\leftarrow$ active mode
                set leader status $\leftarrow$ "ON"
            end if
         end for
        end procedure
until $forever$

repeat
    procedure MOVEMENT(move agent, leader status)
        for all agents with movement = "ON" and leader status ! = "ON" **do**
            while $distance covered < \beta$ **do**
                Create empty list, l
                for $i \leftarrow 1, no : of pings received$ **do**
                    append list $l \leftarrow h_i$
                end for
                calculate h_{mean} of list, l
                set agent heading $h_a \leftarrow h_{mean}$
                move agent with fixed velocity v_a
            end while
            set move agent $\leftarrow$ "OFF"
        end for
        end procedure
until $forever$
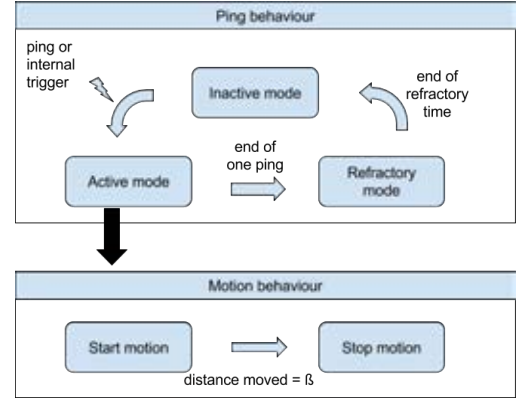
Figure 2: Figure shows the state transition diagram of the FSTaxis algorithm. The algorithm has two behaviors - ping behavior and motion behavior. In ping behavior, there are three states: active, refractory and inactive. An agent is in active state when it receives a ping from a surrounding agent or when its own internal clock triggers. After the ping duration, the agent transitions into a refractory mode. After the refractory time, the agent transitions into the inactive mode. The active mode triggers motion behavior and the agent takes a preset distance towards the ping it received.

gradients are used as test functions. Since depth is of relevance in project subCULTron, it is considered to be the physical quantity of interest in the paper. As mentioned before, the frequencies are scaled according to the equation 1 and Table 1 shows all the constants used in this experiment. The simulation environment used is Netlogo 4.3.1 (Wilensky, 1999). In Netlogo, the test area is divided into "patches" (spatial units) and the agents are called "turtles". For the purpose of this experiment, depth is the physical quantity associated with each patch. The sensor radius of each of the agents are measured in patches and in this experiment it is taken to be 3 patches since it is a reasonable range for underwater communication.

Constants								
	ω	t_p	t_r	s_r	β	α	g_{max}	g_{min}
Value	0.1	5	5	3	1.5	0.008	50	5
Units	-	s	s	p[1]	p[1]	-	m	m

Table 1: Table showing all constants used in the FSTaxis algorithm.

Linear gradient

This section aims to demonstrate the basic implicit capability of the FSTaxis algorithm to traverse gradient. The equation for gradient is that of a line, as shown in equation 2

$$f(x) = x \qquad (2)$$

---

[1] unit p in Table 1 represents number of patches in netlogo
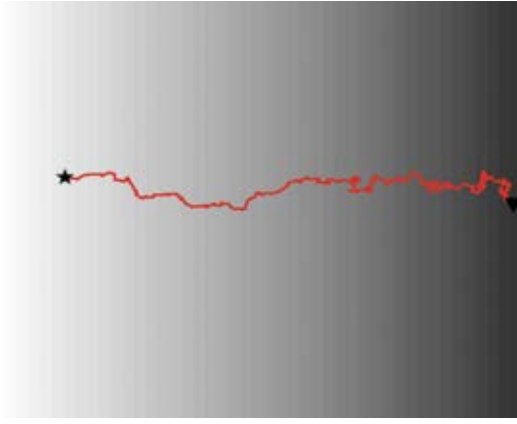
Figure 3: A linear gradient was presented to the FSTaxis algorithm. The grayscale shows the gradient value and the goal is the darkest area; the star symbol represents the starting point and the inverted triangle marks where the swarm converged.

Figure 3 shows the simulation environment in Netlogo setup with a linear gradient. The color scaling represents the gradient value of the environment and therefore the goal of the gradient ascent algorithm will be to go towards the dark colored areas. The red line represents the trajectory of the mean position of the swarm from starting point, represented by the star, to convergence, represented by the inverted triangle. The swarm is said to converge when its mean position oscillates around the area with the maximum gradient value. The trajectory shown is the result of one of the exemplary run from the 100 runs conducted with this test gradient. 100% of the runs resulted in convergence to maximum gradient value.

Hyper ellipsoid gradient

This section presents an environment for testing the FSTaxis algorithm in a relatively more complex gradient, a three dimensional axis parallel hyper ellipsoid. The axis parallel hyper ellipsoid, represented by its standard equation 3, is a convex, continuous function and multiple modal function.

$$f(x) = \sum_{i=1}^{2} x_i^2, where -5.12 < x_i < 5.12 \qquad (3)$$

For the hyper ellipsoid gradient, there are four goals at the corners of the arena with the highest gradient value. The area of the goal (corners of the ellipsoid) is merely 0.23% of the total area of the arena. Therefore, random chances of the swarm converging to the goal is minimal. Figure 4 shows FSTaxis algorithm tested with a hyper ellipsoid gradient, the thick red line shows the trajectory of the swarm. The black star marks the starting point and the inverted triangle shows the area of convergence. The trajectory shown is the result of one of the exemplary run from the 100 runs conducted with this test gradient. 100% of the runs resulted in convergence to maximum gradient value.

Figure 4: An axis parallel hyper ellipsoid gradient was presented to a swarm executing FSTaxis algorithm. The grayscale shows the gradient value and the goal is the darkest area; the star symbol represents the starting point and the inverted triangle marks where the swarm converged.
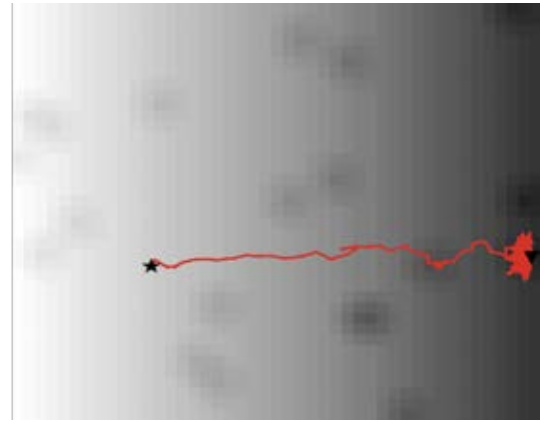
Figure 5: A linear gradient with numerous local maxima was presented to the FSTaxis algorithm. The grayscale shows the gradient value and the goal is the darkest area; the star symbol represents the starting point and the inverted triangle marks where the swarm converged.

Noisy Gradients

In order to test the ability of the FSTaxis algorithm to overcome small local optima, 20 randomly generated obstructions or "hills" have been introduced to the smooth gradient. These obstacles attracts them to stay at these local optima if sufficient exploration is not introduced. Figure 6 and 5 shows the result of a random successful attempt out of the 10,000 iterations of FSTaxis algorithm run with noisy hyper ellipsoid gradient and linear gradient respectively. The experiments with noisy gradients were tested with varying steepness of local optima and spread of each optima. The number of obstructions were kept constant at 20. For each obstruction spread ranging from 1 to 10 and steepness ranging from 0.25 to 2.5 times the normal gradient, 100 iterations were run to observe the convergence to the goal.

334
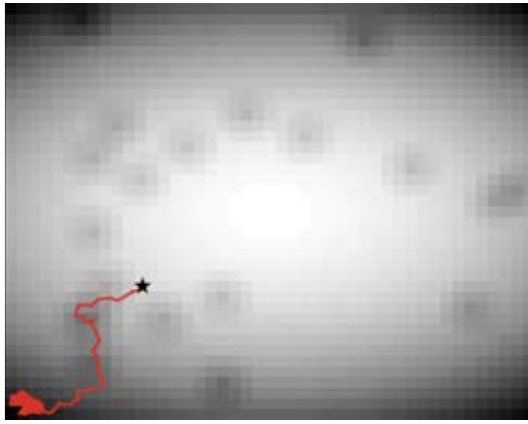
Figure 6: A axis parallel hyper ellipsoid gradient with numerous local maxima was presented to the FSTaxis algorithm. The grayscale shows the gradient value and the goal is the darkest area; the star symbol represents the starting point and the inverted triangle marks where the swarm converged.

Results

The FSTaxis algorithm, as described in the section "Methods", successfully traverses the linear gradient as well as the hyper ellipsoid gradient in all 100 iterations conducted. Figures 3 and 4 show clearly the ability of the algorithm for gradient ascent.

Performance in noisy gradients

When the agents executing FSTaxis algorithm (see subsection "Noisy Gradients") are presented with obstructions in the gradient, they are able to overcome local optima introduced. It can be seen that the agents are eager to climb gradients as seen in 5 and 6. As individual agents move towards the leader, they overshoot the leader (agent whose internal clock triggered a ping) and cross out of the hill to escape the local optima. Figure 7 is a graph relating between the size and steepness of local optima (number of patches) to the percentage runs that converged to the goal. It is seen that when size of local optima is under three patches (for 20 obstacles in the arena) 100% of runs converge to the goal. As size and spread of local optima rises, the rate of convergence decreases.

Performance with multiple gradients

Figure 8 shows the region of convergence relative to the starting point of the swarm for 100 iterations. The X-axis shows the quadrant in the arena where the swarm started and the Y-axis shows the region of convergence. Numbers 1, 2, 3, 4 refer to the quadrants as referred to in the cartesian coordinate system. It is seen that in 95% of the runs, the swarm converges to the goal nearest to it. The 5 % error is attributed to the fact that, when the swarm starts at the midpoint between two gradients, it has to choose which gradient
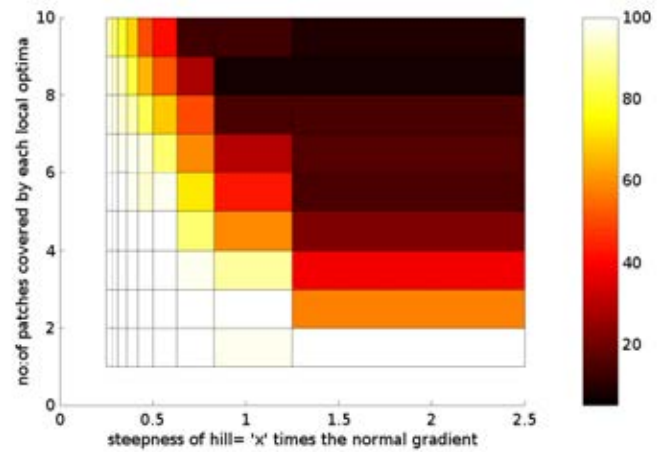
Figure 7: The color map shows the percentage convergence in presence of noise. Each gradient was introduced with 20 obstructions or local optima. The X-axis represents the steepness of each of these obstructions with respect to the normal gradient. The Y–axis represents the area covered by each of these 20 obstructions(measured in patches). Each patch is 0.05% of the entire arena. For each set of hillsize and steepness, 100 iterations were run. White coloured areas show 100% convergence and shades towards red and black coloured areas shows lower convergence as per color map provided.

to ascent. This decision is taken randomly since it depends on which agent's internal clock triggers first.

Discussion

In contrast to many classical solutions of gradient ascent, the FSTaxis algorithm is efficient, simple and requires only simple hardware. Since agents do not actively compare its current gradient value and the previous gradient value, gradient ascent is purely an emergent trait. The FSTaxis algorithm uses no evaluation function and acts purely based on local knowledge. Any agent has to be merely informed about the presence of other agents in its sensor radius. Therefore, the impositions on the agent is to sense the gradient value at current position, adjusting the agent's own ping behavior accordingly and broadcast a 1-bit communication to make its presence known. These qualities make FSTaxis algorithm a choice solution for real world gradient taxis when resources are sparse. Many gradient taxis solutions have been proposed in the past for single agents. Some examples include the standard hill climber (Davis, 1991), helical klinotaxis (Long et al., 2004) etc. While these solutions work well for individual robots, they are not designed for a group of robots. There are a few swarm algorithms for emergent gradient taxis that has been proposed in the past like swarmtaxis(Bjerknes et al., 2007), Artificial Homeostatic Hormone System (Schmickl et al., 2010) etc. The swarmtaxis algorithm is an emergent gradient taxis solution like the FSTaxis algorithm and is based on a single ping broadcast communication. Therefore, it is of merit to compare the FSTaxis
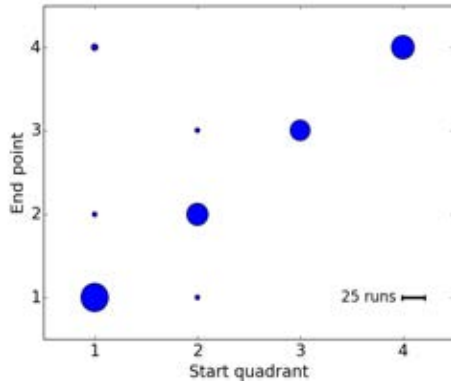
Figure 8: Bubble plot showing the region of convergence relative to the starting point of the swarm for 100 iterations. The reference line at the right lower corner shows the diameter of a 25 run bubble. The X-axis shows the quadrant in the arena where the swarm started and the Y-axis shows the region of convergence. Numbers 1, 2, 3, 4 refer to the quadrants as referred to in the cartesian coordinate system. It is seen that in 95% of the runs, the swarm converges to the goal nearest to it.

algorithm with the swarmtaxis algorithm.

Bjerknes et al. (2007) presented the "swarmtaxis" algorithm which is also an emergent solution for gradient taxis. It is worthwhile to mention swarmtaxis as the FSTaxis algorithm drew inspiration from this work. The swarmtaxis algorithm works by creating a frontier of agents which are facing the source (light) terming them "illuminated". The illuminated agents cast a shadow on agents behind them terming them as "shadowed" agents. The swarmtaxis algorithm works based on the illuminated robots having a higher sensing distance than the shadowed robots and hence, they move away from the shadowed robots. In essence, they move towards the light source.

The swarmtaxis algorithm guarantees that the swarm will converge to the source. It is a stable way to ascent the light gradient however it is not suitable for use in swarm robotics as it imposes various limitations on the kind of gradient it can ascent. For example, the swarmtaxis algorithm assumes that each robot has the ability to physically occlude another from the source. This assumption holds well when light source is at the same level as the agents and not otherwise. FSTaxis algorithm overcomes this limitation by being dependent upon the local gradient value. Therefore, a need for occlusion never arises.

If there are two light sources, the swarmtaxis algorithm will create two frontiers and will not consider the brighter of the sources to move to. Although the swarmtaxis algorithm makes it redundant to have a local gradient value sensor, it is at the cost of not being able to handle multiple sources. The FSTaxis algorithm, on the other hand, is able to handle multiple sources or maxima at the cost of using a local gradient value sensor.

As per the objectives of the paper, boundary conditions for FSTaxis algorithm were investigated. It has been shown that the FSTaxis algorithm is able to work with multiple local optima. As seen in Figure 7, in presence of local optima that are steep enough, the FSTaxis algorithm is likely to get stuck in the local maxima. For problems such as multimodal optimization, for example solving a rotated hyper ellipse, the FSTaxis algorithm does not guarantee convergence to the global maxima.

Conclusion

From this paper, it is demonstrated that FSTaxis algorithm is a feasible solution for gradient ascent in swarm robotics. It is especially attractive because it requires only a single bit communication between the agents.

As discussed previously, the FSTaxis algorithm does not guarantee a solution for multi-modal gradients. In the future, extensions of this algorithm can be formulated for use in multi-modal optimization.

The FSTaxis algorithm works based on sharing of spatial gradient related information via frequency of pings. This paper is a successful demonstration of information exchange without explicitly sending data. In the future, more resources can be dedicated towards how more information regarding gradients can be shared and how the agents can use this to change their behavior.

For relating the current position with its own ping frequency frequency, agents executing the FSTaxis algorithm scales the gradient value as per equation 1. This provides a way to tweak the equation depending on the gradient that is of interest. Moving forward, it would be useful to have a single equation that can be used globally without a need to manually scale the parameters.

Acknowledgments

References

Alcantara, F. and Monk, M. (1974). Signal propagation during aggregation in the slime mould dictyostelium discoideum. *Microbiology*, 85(2):321–334.

Arfken, G. (1985). The method of steepest descents. *Mathematical methods for physicists*, 3:428–436.

Bazeille, S. and Filliat, D. (2011). Incremental topo-metric slam using vision and robot odometry. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4067–4073. IEEE.

Bjerknes, J. D., Winfield, A., and Melhuish, C. (2007). An analysis of emergent taxis in a wireless connected

swarm of mobile robots. In *IEEE Swarm Intelligence Symposium*, pages 45–52, Los Alamitos, CA. IEEE Press.

Bodi, M., Möslinger, C., Thenius, R., and Schmickl, T. (2015). {BEECLUST} used for exploration tasks in autonomous underwater vehicles. *IFAC-PapersOnLine*, 48(1):819 – 824. 8th Vienna International Conferenceon Mathematical ModellingMATHMOD 2015.

Buck, J. and Buck, E. (1966). Biology of synchronous flashing of fireflies. *Nature*, 211:562–564.

Camazine, S., Denenbourg, J. L., Franks, N. R., Sneyd, J., Theraulaz, G., and Bonabeau, E. (2001). Synchronized flashing among fireflies. pages 143–166. Princton University Press, Princton.

Chisholm, R. L. and Firtel, R. A. (2004). Insights into morphogenesis from a simple developmental system. *Nature reviews Molecular cell biology*, 5(7):531–541.

Davis, L. (1991). Bit-climbing, representational bias, and test suite design. In *ICGA*, pages 18–23.

de Oliveira, D. R., Lopes, H. S., and Parpinelli, R. S. (2011). *Bioluminescent swarm optimization algorithm*. IN-TECH Open Access Publisher.

Grodzicki, P. and Caputa, M. (2005). Social versus individual behaviour: a comparative approach to thermal behaviour of the honeybee (*Apis mellifera* l.) and the american cockroach (*Periplaneta americana* l.). *Journal of Insect Physiology*, 51(3):315 – 322.

Kennedy, J. and Eberhart, R. C. (1995). Particle swarm optimization. In *IEEE International Conference on Neural Networks*, Los Alamitos, CA. IEEE Press.

Kernbach, S., Thenius, R., Kornienko, O., and Schmickl, T. (2009). Re-embodiment of honeybee aggregation behavior in an artificial micro-robotic swarm. *Adaptive Behavior*, 17:237–259.

Long, J. H., Lammert, A. C., Pell, C. A., Kemp, M., Strother, J. A., Crenshaw, H. C., and McHenry, M. J. (2004). A navigational primitive: biorobotic implementation of cycloptic helical klinotaxis in planar motion. *IEEE Journal of Oceanic Engineering*, 29(3):795–806.

Nakagaki, T. (2001). Smart behavior of true slime mold in a labyrinth. *Research in Microbiology*, 152:767–770.

Nakagaki, T., Yamada, H., and Hara, M. (2004). Smart network solutions in an amoeboid organism. *Biophysical Chemistry*, 1:1–5.

Nakagaki, T., Yamada, H., and Toth, A. (2000). Intelligence: Maze-solving by an amoeboid organism. *Nature*, 407:470.

Schmickl, T. and Crailsheim, K. (2007). A navigation algorithm for swarm robotics inspired by slime mold aggregation. In Şahin, E., Spears, W. M., and Winfield, A. F. T., editors, *Swarm Robotics - Second SAB 2006 International Workshop*, Lecture Notes of Computer Science, pages 1–13. Springer-Verlag, Berlin, Heidelberg, New York.

Schmickl, T. and Hamann, H. (2011). BEECLUST: A swarm algorithm derived from honeybees. In Xiao, Y., editor, *Bio-inspired Computing and Communication Networks*. CRC Press.

Schmickl, T., Hamann, H., Stradner, J., Mayet, R., and Crailsheim, K. (2010). Complex taxis-behaviour in a novel bio-inspired robot controller. In *Proc. of the ALife XII Conference*, pages 648–655. MIT Press.

Schmickl, T., Thenius, R., Möslinger, C., Radspieler, G., Kernbach, S., and Crailsheim, K. (2008). Get in touch: Cooperative decision making based on robot-to-robot collisions. *Autonomous Agents and Multi-Agent Systems*, 18(1):133–155.

Siegert, F. and Weijer, C. J. (1992). Three-dimensional scroll waves organize dictyostelium slugs. *PNAS*, 89(14):6433–6437.

subCULTron (2015). Submarine cultures perform long-term robotic exploration of unconventional environmental niches. http://www.subcultron.eu/.

Webb, B. (1998). Robots, crickets and ants: models of neural control of chemotaxis and phonotaxis. *Neural Networks*, 11:1479–1496.

Wilensky, U. (1999). Netlogo. *Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL.*

Yang, X.-S. (2009). Firefly algorithms for multimodal optimization. In *Stochastic algorithms: foundations and applications*, pages 169–178. Springer.

Understanding fission-fusion dynamics in social animals through agent-based modelling

Gabriel Ramos-Fernández[1,2], Denis Boyer[2,3], Octavio Miramontes[2,3]

[1]CIIDIR Unidad Oaxaca, Instituto Politécnico Nacional
[2]Centro de Ciencias de la Complejidad, Universidad Nacional Autónoma de México
[3]Departamento de Sistemas Complejos, Instituto de Física, Universidad Nacional Autónoma de México
ramosfer@alumni.upenn.edu

Abstract

This paper describes the way in which we have employed agent-based models to understand fission-fusion dynamics (FFD), a collective pattern of behavior in many social animals. Groups with a high degree of FFD split into subgroups that vary in size, cohesion and composition, often within short temporal scales. These dynamics are thought to be more complex than those of other species with cohesive, stable groups, leading to hypotheses about the origin of social intelligence. Also, a flexible grouping pattern is supposed to be an adaptive solution to the temporal and spatial variation in feeding resources. We have used models where relatively simple agents forage in realistic, heterogenous environments and have shown that, for intermediate levels of heterogeneity in the size of food patches, agents form subgroups that vary in size and composition in a similar fashion as they do in species with a high degree of FFD. We have also explored the idea that by splitting in subgroups that vary in size, animals can exploit a heterogeneous environment with ephemeral food sources more efficiently than cohesive groups. Agent-based models have provided ways to test hypotheses and develop predictions about social and ecological dynamics.

Fission-fusion dynamics (FFD) is a property of many groups of animals that split in subgroups of variable size, composition and degree of cohesion (Kummer, 1971; Aureli, et. al. 2008). Species that show this property, to different degree, include, among mammals: baboons (*Papio* spp), chimpanzees (*Pan* spp), spider monkeys (*Ateles* spp), African elephants (*Loxodonta africana*), hyenas (*Crocuta crocuta*), dolphins (*Tursiops truncatus*), bats (*Myotis bechsteinii*) and several species of birds (Aureli, et al. 2008; Silk, et al. 2014).

A proposed adaptive function of FFD is that it allows for an efficient exploitation of resources that are distributed heterogenously in time and space, adjusting the size of the foraging units to the local density of resources (Kummer, 1971; Aureli, et.al. 2008). While there have been some partial tests of this idea, the complexity of environmental variables and the different ways in which a particular behavior could be adaptive have provided contradictory results (Chapman, et al. 1995; Newton-Fisher, et. al. 2000; Symington 1988).

In terms of the mechanistic basis of FFD at the level of individual behavior, it has been proposed that, because species with FFD confront a greater diversity of social situations than species with cohesive groups, they should be subject to selection for higher cognitive abilities. These would underlie processes of information sharing or withholding and special social interactions that allow for individuals to cope with the constant fissioning and fusioning of subgroups (Aureli, et. al. 2008).

We have used agent-based models to complement field observations of the social behavior of spider monkeys (*Ateles geoffroyi;* Ramos-Fernández, et. al. 2003) in order to understand their movement and grouping patterns. We extended an agent-based model initially aimed at understanding the movement trajectories of a single forager in heterogenous environments (Boyer, et al. 2006), by incorporating several foragers (Ramos-Fernandez et al. 2006). Our goal was to understand the minimum conditions that would give rise to a fission-fusion grouping pattern among foragers. We set up an environment where discrete food patches vary in size according to an inverse power-law (as real trees do in many tropical and temperate forests: Enquist and Niklas, 2001) with some large patches and many small patches. Here, a set of foragers moves according to a local optimality rule, maximizing the size of the next visited patch and minimizing the distance traveled to it. In each iteration of the simulation, a forager takes a step or reduces the food content of a patch by one unit. In addition, foragers do not come back to a previously visited patch.

Even though the model does not specify any interaction among foragers, it does have an implicit fission-fusion mechanism: when two foragers coincide in the same patch, they form a temporary aggregation that can continue as they forage together in other patches, whereas they can also split due to their previous history of visits. Figure 1 shows a summary of the different situations observed in the model: for certain values of the parameter that controls patch size heterogeneity, the foraging trajectories and the size of the temporary aggregations are similar to those described in field studies of spider monkeys (Symington 1988; Ramos-Fernandez & Ayala-Orozco 2003). Particularly, intermediate values of patch size heterogeneity led to the longest foraging trajectories and the largest aggregations. This is because foragers traveled long distances to reach large patches that were neither rare nor scarce, coinciding with others more often at these large patches (Ramos-Fernandez et al. 2006).

These results show that an important ecological influence on FFD could be the relative abundance of patches of different size, in contrast with the usual measures of overall food abundance or average patch size. This can be taken as a prediction for field studies in which the size of visited patches can be measured.

We then developed a simpler version of this model, aimed at testing the idea that groups with FFD could be more
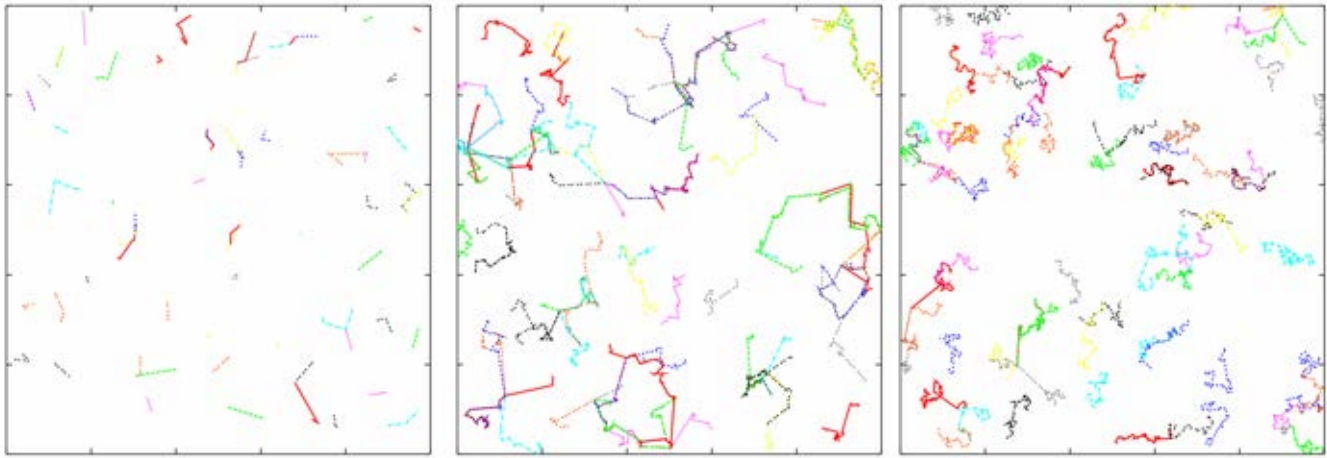
Figure 1. Movement trajectories described by foragers in the agent-based model by Ramos-Fernández et al. (2006). Each line (in different color) represents a different forager. Starting from randomly assigned positions, foragers move to the nearest and largest food patch available (points not visible in the figure). The left panel corresponds to a situation with maximum heterogeneity in patch size and thus a comparatively large proportion of large patches. In this situation, foragers find a large patch very close and the simulation "freezes" with very little interaction between foragers. On the contrary, the panel on the right represents a situation with minimum heterogeneity in patch size, and thus very few large patches. Here, foragers simply visit the nearest patch, describing long trajectories with many changes in direction. They may coincide with others but they mostly forage locally. The situation in the middle panel represents an intermediate level of heterogeneity in patch size, with some large patches that are often worth visiting even when they are far. The trajectories described by foragers are longer, with a combination of small and long "steps" between patches. This is the situation when foragers formed the largest subgroups with others with whom they coincided in these large patches.

efficient than cohesive groups at exploiting ephemeral and unpredictable patches, such as tropical trees, which have short fructification periods, each species with fruit at different times of year (Rathcke and Lacey, 1985). In this environment, food patches have a randomly assigned amount of food, which is present for a randomly chosen period of time. Foragers search for food using a correlated random walk, and they can be either cohesive (all agents move together) or separate (all agents move independently, as subgroups of spider monkeys do: Ramos-Fernández et al. 2011). The foraging efficiency is calculated as the number of food units obtained over the distance traveled by each forager. In order to control for the effect of group size on the foraging efficiency, the same number of foragers are present in both conditions. Preliminary results show that the efficiency of separate foragers is in fact greater than that of the cohesive foragers.

We have successfully used agent-based models to explore the minimum, simplest conditions that could produce a collective pattern out of local interactions between agents and their environment. Also, these models have served to develop predictions to be tested with further fieldwork and to test hypotheses about the adaptive function of FFD.

References

Aureli F., Schaffner C.M., Boesch C., Bearder S.K., Call J., Chapman C.A., Connor R., Di Fiore A., Dunbar R.I.M., Henzi S.P., Holekamp K., Korstjens A.H., Layton R., Lee P., Lehmann J., Manson J.H., Ramos-Fernández G., Strier K.B. and van Schaik C.P. (2008). Fission-Fusion Dynamics: New Research Frameworks. *Current Anthropology* 49: 627-654.

Boyer D., Ramos-Fernández G., Miramontes O., Mateos J.L., Cocho G., Larralde H. Ramos H. and Rojas F. (2006). Scale-free foraging by primates emerges from their interaction with a complex environment. *Proceedings of the Royal Society of London Series B: Biological Sciences* 273:1743-1750.

Chapman, C., Chapman, L. and Wrangham, R. (1995). Ecological constraints on group size: an analysis of spider monkey and chimpanzee subgroups. *Behavioral Ecology and Sociobiology* 36:59–70.

Enquist, B.J., and Niklas, K.J. (2001) Invariant scaling relations across tree- dominated communities. *Nature* 410:655–660.

Kummer, H., (1971). *Primate societies: Group techniques of ecological adaptation.* Aldine, Chicago, IL.

Newton-fisher, N.E., Reynolds, V. and Plumptre, A.J.. (2000). Food Supply and Chimpanzee (*Pan troglodytes schweinfurthii*) Party Size in the Budongo Forest Reserve , Uganda. *International Journal of Primatology* 21:613–628.

Ramos-Fernández, G. and Ayala-Orozco, B. (2003). Population size and habitat use of spider monkeys in Punta Laguna, Mexico. In L. Marsh, editor, *Primates in Fragments: Ecology and Conservation,* pages 191–210. Kluwer/Plenum Press, New York.

Ramos-Fernández, G., Vick, L.G., Aureli, F., Schaffner, C. and Taub, D.M. (2003). Behavioral ecology and conservation status of spider monkeys in the *Otoch ma'ax yetel kooh* protected area. *Neotropical Primates* 11:157-160.

Ramos-Fernández, G., Boyer, D. and Gómez, V. (2006). A complex social structure with fission–fusion properties can emerge from a simple foraging model. *Behavioral Ecology and Sociobiology* 60:536–549.

Ramos-Fernández, G., Pinacho-Guendulain, B., Pérez, A.D. and Boyer, D. (2011). No Evidence of Coordination Between Different Subgroups in the Fission-Fusion Society of Spider Monkeys (*Ateles geoffroyi*). *International Journal of Primatology* 32:1367–1382.

Rathcke, B., and Lacey, E. P. (1985). Phenological patterns of terrestrial plants. *Annual Review of Ecology and Systematics* 179-214.

Silk, M. J., Croft, D. P., Tregenza, T., and Bearhop, S. (2014). The importance of fission–fusion social group dynamics in birds. *Ibis*, 156:701-715.

Symington, M.M. (1988). Food Competition and Foraging Party Size in the Black Spider Monkey (*Ateles paniscus chamek*). *Behaviour* 105:117–134.

Shape matters in cooperation

Dusan Misevic[1], Antoine Frénoy[1], Ariel B. Lindner[1] and François Taddei[1]

[1]Center for Research and Interdisciplinarity, INSERM U1001, Medicine Faculty, site Cochin Port-Royal, University Paris Descartes, Sorbonne Paris Cité, 24, rue du Faubourg Saint Jacques, 75014 Paris, France
dule@alife.org

Why do organisms cooperate with each other? This seemingly simple question has motivated a staggering amount of theoretical and experimental research. When cooperating with others carries a direct fitness cost for the individual, natural selection should act against such behavior. However, cooperation is widespread across natural systems, from birds and bees to bacteria. Past research has identified many factors favoring or disfavoring the evolution of cooperation. For example, we know that properties of public good molecules (Misevic et al. 2012) affect the evolution of cooperation. Here we summarize the results and provide additional discussion about the implications of our recently published work on the importance a previously overlooked factor, the population shape (Misevic et al. 2015).

In the past, we have studied different aspects of cooperation using a well established *in silico* system, Aevol (Frénoy et al. 2013) In Aevol, digital organisms with double-stranded binary genomes and complex genetic architecture mutate, compete, and evolve over thousands of generations. The ancestral organism is not a cooperator, but populations may evolve to secrete different amounts of public good. The secretor pays a cost proportional to the amount of public good it produces, and all neighboring individuals benefit equally form the secreted molecules. The secreted molecules diffuse to neighboring cells and degrade over time. Generations are synchronous, with nine individuals in the classical Moore neighborhood competing to populate the next generation. For the purpose of the study on the population shape, we have also introduced two new, simpler systems, Aevol-lite and CAevol. Aevol-lite has all the properties and simulation mechanics of Aevol, but instead of binary strings and non-trivial genotype to phenotype to fitness mapping, each Aevol-lite individual is represented by a single number, a binary gene identifying whether it secretes a public good or not. CAevol is a further simplification, a system without public good, in which individuals with pure cooperate/defect strategies play a classical Prisoner's dilemma.

Aevol individuals live on a quadrilateral grid with periodic boundary. All the locations on the grid are always full. We studied two different populations shapes, a bulky 100x100 torus (akin to a fat doughnut) or a slender 4x2500 one (akin to a slender bicycle tire). It is interesting to note that we noticed the effect by accident, while studying a different population property and inadvertently modifying the shape as well. We corrected the mistake in the code, but the result remained: more secretion evolved in bulky than in slender populations (Figure 1A). This evolutionary outcome was highly robust and not affected by changes in any of the other cooperation parameters, such as cost and benefit of secretion, or the diffusion and degradation rates.

The result was not intuitive and did not lend itself to an obvious explanation. We expected slender populations to facilitate separation between secretor and non-secretors, leading to fewer interactions, thus selecting for cooperation. In order to explain the prevalence of cooperation in populations of different shape, we closely examined the dynamics of cooperating patches, sub-populations. Additionally, we decided to simplify our model system, moving from Aevol onto first Aevol-lite and finally CAevol. This allowed us to rule out system idiosyncrasies, or plain bugs in the code, and test the generality of the results. Indeed, bulky populations evolved more cooperation no matter what particular simulation system or mode of cooperation (public good v. Prisoners' dilemma) we used (Figure 1).

Studying cooperation in Aevol-lite and CAevol, where there are only two types of individuals, made it much easier to visualize the population over time. We suspected that populations are not clonal, consisting of identical individuals, but instead a diverse assembly of cooperators and non-cooperators in a dynamic equilibrium. Indeed, by plotting Aevol-Lite populations we saw exactly that: populations are a collection of expanding and shrinking patches of cooperators and non-cooperators, constantly taking over one another. And it is in the dynamics of these patches that we found the answer for the population shape effect.
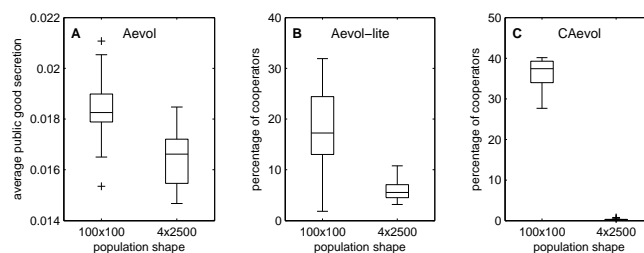
Figure 1. Average cooperation in bulky (100x100) and slender (4x2500) populations in (A) Aevol, (B) Aevol-lite, and (C) CAevol (from Misevic et al. 2015). Cooperation after 50,000 generations is quantified as the average amount of the public good secreted (A) or the percentage of cooperators (B and C). Line marks the median, the box edges are the 25th (q1) and the 75th (q3) percentile, the whiskers the most extreme data points still smaller than $q3 + w(q3 - q1)$ and larger than $q1 - w(q3 - q3)$, where $w = 1.5$.

In slender populations, the population shape constrained the expansion of cooperator patches. This can clearly be seen in Figure 2B, zooming in on a part of a single population over time, where after the first few generations, the cooperator patch (shown in gray) can increase only by eight individuals, four on the left and four on the right. In contrast, in the bulky population, such a patch can expand on all four sides, potentially by many more individuals in each generation. In Figure 2A, which focuses on a section of a bulky population over time, we exactly see such fast expansion. In both cases, the patches are eventually invaded from within by non-cooperators, which arise as mutants. However, before the cooperator patch gets completely overrun, it has a chance to expand to a greater size in bulky than in slender population. And that is precisely the reason for more cooperation in bulky than in slender populations: in all cases cooperator patches arise continuously, only to be taken over by no-cooperators, but in bulky populations they grow bigger, resulting in more cooperators present in the population at any given point in time. We quantified this difference by exhaustively simulating a single cooperator patch from its inception, through its ultimate demise. The results confirmed our verbal analysis of patch dynamics: no matter what the rate of mutation from cooperator to non-cooperators, over the entire lifetime of a patch, more cooperators existed in the bulky than slender population. This patch analysis was done in Aevol-lite, and was further confirmed in Aevol. By using Markov Cluster Algorithm we identified and measured the number and size of clusters formed by individuals based on the amount of public good secretion. We found that those clusters were smaller and more numerous in slender that in bulky populations, exactly in line with what we saw in Aevol-lite.

After hundreds of populations, millions of generations, and billions of individuals, we confirmed that shape does matter for cooperation and were able to explain the effect through the analysis of within-population patch dynamics. But what does our result mean for simulations of cooperation or for the study of evolution of cooperation in general? After all, population shape, as we defined it here, seems to be a rather peculiar parameter. However, it is not as obscure as it seems to be, since we can certainly think of cooperating populations living in complex 3D structures (soil, human lung), 2D plans (petri dish), or even effectively 1D (filamentous cyanobacteria). While we have not extended the simulation to different dimensions, our analysis indicates that the results would hold: more cooperation in populations of higher dimensions. Moreover, for some time there has been a push to consider different, more complex population structures, namely graphs (Ohtsuki et al. 2006). Our results make a strong argument that such treatment is useful and potentially necessary when considering the evolution and maintenance of cooperation, *in silico* or *in vivo*. The conclusions from the abstract bulky and slender populations directly extend to populations in which individuals have different number and strength of cooperative interactions, suggesting that graphs with higher connectivity will promote cooperation.

Finally, the heterogeneous and complex map of interactions between actors has already been studied in epidemiology (Salathé et al. 2010). A recent study using a microbial system established a connection between cooperation and information transfer (Dimitriu et al. 2014),

which allows us to make a connection between two fields here. In cooperation as well as epidemiology, all properties relating to the interactions between individuals, including the population shape and structure, should be considered because they may constrain and alter the maintenance and spread of (potentially infections) cooperative trait.
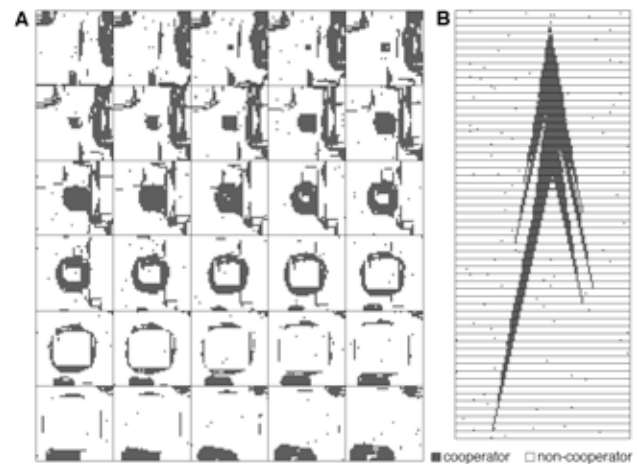
Figure 2. Example of a full lifecycle of a cooperator patch in (A) a bulky and (B) a slender Aevol-lite population (adapted from (Misevic et al. 2015)). Each square is either a cooperator (dark) or a non-cooperator (white cell). Each square in panel (A) and rectangle in panel (B) represents a snapshot of a population region from a single generation, with generations increasing from left to right, top to bottom.

References

Dimitriu T, Lotton C, Benard-Capelle J, Misevic D, Brown SP, Lindner AB, and Taddei F. 2014. Genetic information transfer promotes cooperation in bacteria. Proceedings of the National Academy of Sciences of the United States of America 111:11103-11108.

Frénoy A, Taddei F, and Misevic D. 2013. Genetic architecture promotes the evolution and maintenance of cooperation. PLoS Computational Biology 9:e1003339.

Misevic D, Frénoy A, Lindner A, and Taddei F. 2015. Shape matters: lifecycle of cooperative patches promotes cooperation in bulky populations. Evolution 69:788-802.

Misevic D, Frénoy A, Parsons DP, and Taddei F. 2012. Effects of public good properties on the evolution of cooperation. Pp. 218-225 *in* C Adami, DM Bryson, C Ofria, and RT Pennock, eds. Proceedings of Artificial Life 13. MIT Press.

Ohtsuki H, Hauert C, Lieberman E, and Nowak MA. 2006. A simple rule for the evolution of cooperation on graphs and social networks. Nature 441:502-505.

Salathé M, Kazandjieva M, Lee JW, Levis P, Feldman MW, and Jones JH. 2010. A high-resolution human contact network for infectious disease transmission. Proceedings of the National Academy of Sciences of the United States of America 107:22020-22025.

Development

Developmental Encodings Promote the Emergence of Hierarchical Modularity

Jessica Lowell[1] and Jordan Pollack[1]

[1]DEMO Lab, Departmentl of Computer Science, Brandeis University, Waltham, MA 02453
jessiehl@cs.brandeis.edu

Abstract

While it has been observed (Hornby et al., 2001) that developmental encodings in evolved systems may promote modularity, there has been little quantitative study of this phenomenon. There has also been little study of the factors driving the emergence of hierarchical modularity - modularity on multiple levels, in which the modules found at a finer-grained level can serve as elements in a coarser-grained network that is also modular - despite the fact that most fields with an interest in modularity, including biology and engineering, define hierarchy as an important aspect of modularity. We examine the effect of developmental encodings on the emergence of multiple levels of modularity through the lens of two developmental systems, GRNEAT and GENRE, and find evidence that developmental encodings promote this emergence of modular hierarchy.

Introduction

Below, we examine interactions between development and hierarchical modularity in artificial systems. Modularity, the organization of a system into a hierarchical system of interacting subparts, is observed in many systems both natural and engineered (Koza, 1992; Simon, 1996; Hartwell et al., 1999), and has become important as evolutionary systems are used in increasingly complex applications. Simulations of development, the process by which a mature phenotype is constructed from an organism's genetic code, have been used in computational studies both in conjunction with and distinct from simulations of evolution. We briefly discuss modularity in evolution, followed by an overview of artificial development.

Modularity

Biological systems, including biological networks such as neural networks and bacterial metabolic networks, and other kinds of biological systems such as tissues (which are assembled from cells), tend to be modular. The definition of modularity is somewhat vague - though generally referring to the degree to which a system is composed of separable, recombinable components - and can be used differently in different fields and subfields. Bolker (2000) attempted to

define a list of characteristics of modularity that would be appropriate across different subfields and levels of study in biology, including greater internal integration of modules as compared to external integration, the ability to delineate modules from their surroundings, and module performance that is greater than the sum of its parts. Schilling (2002) found that a variety of fields, including technology, psychology, biology, American studies, and mathematics, define hierarchical nesting as an aspect of modularity. It is worth noting that the hierarchical aspect of modularity, the emergence of which we explore in this paper, has not traditionally been examined in simulated evolution studies, despite its importance in how most fields define modularity. We chose to focus on hierarchy because of this gap in the literature, and because development is such a key factor in the formation of many hierarchically modular biological systems, such as organisms.

Evolutionary algorithms tend to produce nonmodular solutions - though there are some exceptions, as in the coevolutionary algorithm of Juille and Pollack (1996), which used genetic programming to produce modular solutions to the intertwined spirals problem. These nonmodular solutions are often connected in complicated ways and perform better on the task for which they are optimized than the more modular solutions designed by human designers (Thompson, 2012) (Vassilev et al., 2000). However, while this tendency against modularity can produce well-performing solutions for simple problems, it makes it difficult for evolved systems to solve complex problems (Kashtan and Alon, 2005). While this issue can be addressed by building the encapsulation of modules into algorithms, this does not illuminate how modularity evolves in nature, and it means possibly missing out on some design benefit that comes with modularity emerging rather than being hard-coded. In addition, allowing modularity to emerge through an iterative process may allow for nonmodular, high-performing species of solutions to develop modularity over time while preserving their strong performance.

In recent years, there have been several studies examining the emergence of modularity in both natural and sim-

ulated evolution. (Lipson et al., 2002) found, in a study of minimal substrate modularization, that modular separation is logarithmically proportional to rates of environmental variation, and suggested using variable rather than fixed fitness criteria for the evolutionary design of engineered systems. This hypothesis was supported by the work of (Kashtan and Alon, 2005) in computational evolution studies, and (Kashtan et al., 2007; Parter et al., 2007) in natural evolution studies, which found that modularity evolves in response to varying environments (called modularly varying goals) in which individuals perform varying tasks that are decomposable into common subtasks. The requirement that subtasks be performed in sequence, as a chain, has also been found to promote the evolution of modularity (Calcott, 2014). Another possible explanation for modularity's evolution was proposed by (Clune et al., 2013), which suggested that modular networks evolve in response to a small decrease in fitness for each connection in the network - a connection cost - representing the energy cost of forming a link in a physical network. A similar energy cost imposed on the NEAT neuroevolution algorithm, on a problem in which some solutions that evolve are modular, has been found to increase consistency in modularity emergence (Lowell and Pollack, 2014).

Artificial Development

Artificial development, also known as artificial embryology, is an area of artificial life that models biological processes of development, in which there are layers of abstraction between a genotype and a phenotype. The phenotype begins with a seed or embryo and progresses toward maturity according to a set of rules or interactions. The individuals on which evolutionary or other forces are acting are these processes by which the embryo develops. Developmental systems and other forms of indirect encodings of solutions can be contrasted with direct encodings, in which each component of the phenotype is made explicit in the genotype. As the problems being solved by evolutionary computation have grown in complexity, scalability has become an important aspect of the design of new evolutionary computation techniques, and certain properties of developmental systems, such as compact genotypes, lend themselves well to scalability (Bentley and Kumar, 1999; Hornby and Pollack, 2001a), which motivated much early research on artificial development (Tufte, 2008). A system that uses artificial development may be called a generative or developmental system.

Often, these developmental encodings are based on biological developmental processes and principles. Doursat (2009) used lower-level developmental processes such as cell division/differentiation and morphogen gradients to create a self-patterning "organic canvas," and (Miller and Banzhaf, 2003) created a model for the programming of a cell, using cell division and simulated chemical environ-
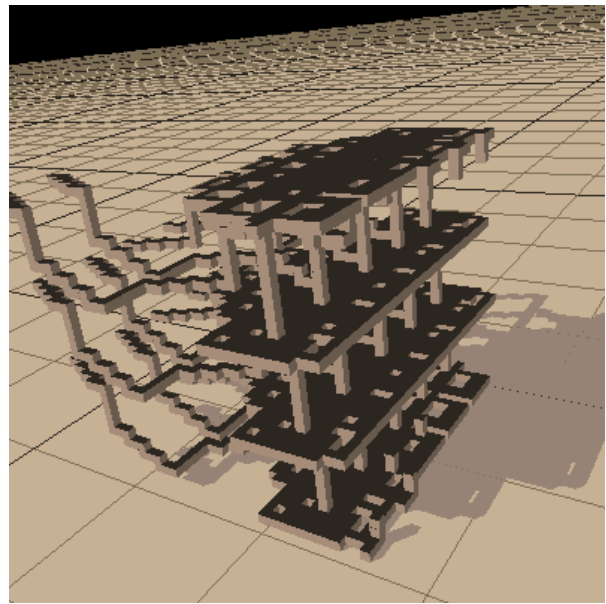
Figure 1: Visualization of an example brick "table" structure produced by GENRE.

ments, that was able to recreate a French flag and other patterns. Other approaches have involved the use of simulated gene regulatory networks (Guo et al., 2009), the exploitation of biological principles of degeneracy (Whitacre et al., 2010), and the evolution of grammars to generate programs or expressions in a given language (O'Neill and Ryan, 2001).

Below, we provide a brief overview of the two different generative systems used in this study.

GENRE

GENRE (Hornby and Pollack, 2001b) is a developmental system that was designed to create more complex virtual creatures than had been created using earlier artificial life techniques. It took a grammatical approach, evolving Lindenmayer systems (L-systems), (Lindenmayer, 1968), parallel grammatical rewriting rules originally designed to model plant growth, that took in parameters and would generate creatures with hundreds of components. The L-systems were applied iteratively to rewrite strings of commands through which to construct creatures or other structures, such that complex strings were constructed from simple ones. Hornby analogized the parallel nature of the rules, and the repetitive structures that they tended to produce, to concurrent cell division. The system outperformed a nongenerative system on performance, creature size, and natural look, when applied to the design of mobile robots and block-based "table" structures. Hornby et al. 2001 observed that the generated robots appeared to exhibit modular properties, but did not attempt to quantify this. An example GENRE-
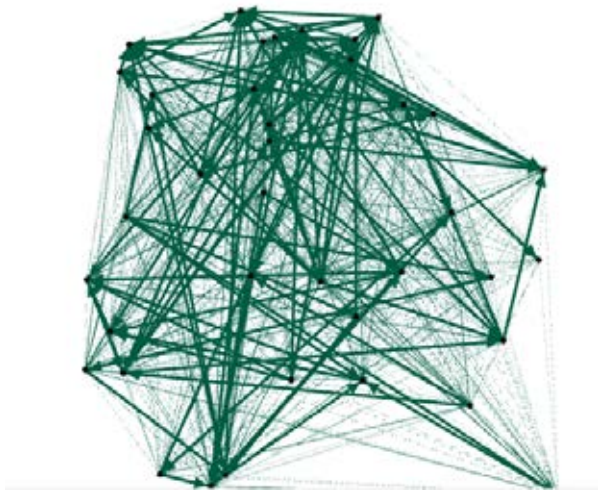
Figure 2: Visualization of an example artificial gene regulatory network (GRN) produced by GRNEAT.

Figure 3: Illustration of two intertwining spirals, which must be distinguished from each other in the intertwining spirals problem.

produced structure can be seen in Fig.1.

GRNEAT

GRNEAT (Cussat-Blanc et al., 2015) evolves artificial gene regulatory networks, or GRNs (Banzhaf, 2003), which are simplified models of the genetic regulatory networks seen in biological systems, and used to control various kinds of agents. The GRNEAT algorithm evolves lists of proteins, which are then developed into network models with matrices of enhancing and inhibiting weights between nodes, mimicking the developmental module function of biological GRNs. The protein lists are used to initialize a GRN, which then updates its weights by calculating interactions between the proteins. It is based on NEAT (Stanley and Miikkulainen, 2002), a well-known algorithm for evolving neural networks, and retains NEAT's major distinguishing features: initialization with small networks, a crossover operator that preserves subnetworks during GRN recombination, and the use of speciation to give growth opportunity to potentially promising innovations. However, a key difference is that artificial GRNs are inherently a developmental encoding, as biological gene regulatory networks are, while NEAT is a direct encoding algorithm. An example GRNEAT network, visualized in Gephi (Bastian et al., 2009), can be seen in Fig.2

Methods

To test the effects of development on hierarchical modularity, we used the GENRE algorithm, which uses L-systems to model parallel cell division, and the GRNEAT algorithm, which evolves protein lists for construction of artificial gene regulatory networks in a manner mimicking NEAT's neuroevolution methods, both of which are described briefly

above. We chose to compare GRNEAT to NEAT because, as stated above, GRNs are developmental by nature, so we could not simply compare a developmental GRN encoding to a nondevelopmental one. We ran GENRE on the brick-table-building problem that was one of its original test problems in (Hornby et al., 2001), which rewards individuals for minimizing the number of bricks and maximixing height, surface area, volume, and stability, and compared the results to those obtained by a non-developmental evolutionary algorithm that is packaged with GENRE for the purpose of running comparisons. We ran the GRNEAT evolutionary process on the problem of distinguishing two intertwined spirals, also called the intertwined spirals problem (Lang, 1988), which is illustrated in Fig.3, with fitness being measured by error ranging from -1 to 0, and compared the results to those obtained by running both feedforward and recurrent versions of the NEAT4J open source Java implementation of NEAT (Simmerson, 2006) on the same problem.

Key parameters for GENRE and for GRNEAT/NEAT are listed in Table 1 and Table 2 respectively. While the comparisons between GRNEAT and NEAT were done primarily using simulations of 250 generations, we also did a set of runs of GRNEAT that were only 10 generations, to see whether any modularity that existed was actually emerging over time or was present early in the simulation. We did not do 10-generation runs for NEAT because it had made almost no progress at solving the intertwining spirals problem after only 10 generations. In the NEAT4J implementation

346

of NEAT, there is an option to allow or disallow recurrent neural networks. We decided to allow recurrency for a more even comparison, as GRNEAT produces recurrent networks. To prevent either the GRNs or neural networks simply memorizing a sequence of outputs rather than learning a mapping from coordinates to spiral ID, in both GRNEAT and NEAT, we used a fresh copy of the pre-initialized network for each new input.

Problem Version	Trials	Generations	Num R, P, C
GENRE	10	100	10, 2 , 2
Nongenerative	10	100	1, NA, NA

Table 1: Key parameters in GENRE experiments. R is the number of production rules, P is the number of parameters per rule, C is the number of condition-successor pairs per rule.

Problem Version	Trials	Generations	pC, pM, PopSize
GRNEAT	20	10	0.25, 0.75, 500
GRNEAT	20	250	0.25, 0.75, 500
NEAT	20	250	0.25, 0.75, 500

Table 2: Key parameters in GRNEAT experiments. pC is probability of crossover, pM is probability of mutation, PopSize is Population Size.

In order to look at the quantitative modularity of GENRE's and its non-developmental counterpart's brick table structures, we needed to represent the structures as networks. In order to do that, we defined each brick as a node, and each case of a face of one brick touching a face of another brick as a link. The link structure was binary, with all links being represented in the adjacency matrix as having a value of 1, and all other elements of the adjacency matrix having a value of zero. This was not necessary for GRNEAT/NEAT, as both GRNs and neural networks are already represented as networks, with links having non-binary weights. Since GRNEAT produces both a matrix of enhancement weights and a matrix of inhibition weights, we combined them into a single weight matrix by subtracting the inhibition factors from the enhancement factors.

Many artificial life and theoretical biology studies of modularity use the metric Q, defined by the approach of (Newman and Girvan, 2004). This approach determines Q by looking at the percentage of edges in the network that connect nodes in the same module, and substracts the expected value for that percentage in a network with the same number of modules but random connections. The modules are defined by a previous part of the algorithm that splits the network into the modules that would maximize Q. Mathematically, the equation for Q in the Newman-Girvan algorithm is:

$$Q = \sum_{s=1}^{k} \left[\frac{l_s}{L} - \left(\frac{d_s}{2L} \right)^2 \right] \quad (1)$$

where L is the number of edges, K is the number of modules, d_s is the sum of degrees of nodes in module s, and l_s is the number of edges in that module.

This method is very useful for examining a single layer of modularity in binary networks (i.e. networks where there is either a connection between two nodes or there is not). However, the weights of links between nodes in GRNs can vary by several orders of magnitude. Both GRNs and recurrent neural networks may benefit from a modularity metric that can account for directedness. And the Newman-Girvan approach only looks for one layer of modularity, rather than for hierarchical modularity. Accordingly, we used the "Louvain method," which was designed for speed, maximization of community detection, and the detection of hierarchical levels of modularity, to determine Q (Blondel et al., 2008). In the Louvain method, each node in the network is initially assigned to its own module, and the modularity Q is calculated according to the following equation for a weighted graph:

$$Q = \frac{1}{2m} \sum ij \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) \quad (2)$$

where A_{ij} represents edge weight between nodes i and j, m represents half the sum of the graph's edge weights, δ is a delta function, c_i and c_j are node communities, and k_i and k_j are the sums of the weights of all edges attached to node i and node j respectively.

Then, for each node, the algorithm calculates the change in modularity, the equation for which depends on whether the version of the algorithm for directed or undirected graphs is being used, for moving that node into the module of each of its neighbors. Once this change is calculated for all modules that the node is connected to, the node is moved into the module that would result in the greatest modularity increase (or left in place if no modularity increase is possible). If no increase is possible, the first level of Q is equal to the current modularity of the network. Subsequent, hierarchical levels of Q are calculated the same way in subequent phases of the algorithm, by using the modules from the previous level as nodes in a new network. For our study, we used Antoine Scherrer's MATLAB implementation of the Louvain method (Scherrer, 2008).

Because modularity can be positive or negative (where negative modularity means that there is less internal integration among modules than one would expect to see in a random graph), we defined a level of modularity as occurring when the Louvain algorithm produces a positive modularity

Figure 4: The mean number of levels of hierarchical modularity produced by a developmental network-evolving system, GRNEAT, was higher than that produced by the non-developmental one on which it is based, NEAT. The 95% CIs for GRNEAT and NEAT were 1.85-2.05 and 1.09-1.51. Number of trials N = 20, p < 0.0001

Figure 5: a) A GRNEAT-produced GRN with the links lightened for easier viewing. The nodes are colored according to the 8 first-level modules found by the first phase of the Louvain algorithm, which found that this level of modularity had $Q = 0.3018$. b) A single module of the GRN. c) The network of the next level of hierarchy, with each of its node representing, and color-coded as, a module from the previous level. d) The same next-level network, with $Q = 0.3266$, with the nodes colored in five new colors according to the 5 second-level modules found by the second phase of the Louvain algorithm.

value, with an allocation of nodes into modules that (if there are lower levels) involves combining some or all modules from the next-lowest level. To determine whether the numbers of hierarchical levels of Louvain modularity were equal in our sets of results, we used Welch's t-test, a variant of the traditional Student's t-test that is robust to non-normality in data and difference in variance between samples. We did not test for differences in the actual Q values of the lowest or other levels, as they were tangential to the question of hierarchy. In practice, Q values for specific levels were between 0.12 and 0.52 for both GRNEAT and NEAT (with most being between 0.2 and 0.4, indicating moderate amounts of single-level modularity), and between 0.3 and 0.72 for both GENRE and its direct encoding counterpart.

Results and Discussion

Our first comparisons were between a set of 20 trials of GRNEAT on the intertwined sprials problem and 20 trials of NEAT with recurrency allowed on the intertwined sprials problem, with mutation probability = 0.75 and crossover probability = 0.25, across 250 generations. In Fig.4, we can see that the best solutions produced in the GRNEAT trials had a mean number of levels of modularity of 1.95, while those produced in the NEAT trials had a mean number of levels of modularity of 1.3, a full 33% lower. This difference in the means was statisically significant ($p < 0.0001$). The emergence of multiple levels of modularity in one GRN is shown in Fig.5.

We wanted to examine whether the increased levels of modularity seen in GRNEAT were something that was emerging rather than something hard-coded into all GRNs.

Because the average fitness of the NEAT neural networks after 250 generations was notably worse than that of the GRNEAT GRNs, and the networks notably smaller (see Fig.6), we also wanted to compare the NEAT networks to GRNs of more similar fitness and size. Accordingly, we compared the 20 250-generation GRNEAT trials to 20 10-generation trials (Fig.7, and, as the 10-generation GRNEAT GRNs were similar in size and fitness to the NEAT neural networks, we compared the 10-generation GRNEAT trials to the NEAT trials (Fig.8)

We can see from these figures that the mean hierarchical modularity of GRNEAT-produced GRNs (along with the size) has increased by nearly a third (a mean 1.5 levels of modularity vs 1.95 levels) between the 10th and 250th generations. We can also see tentative evidence (with a p value that is low but not statistically significant) that GRNEAT GRNs already have greater hierarchical modularity after 10 generations than NEAT recurrent neural networks have after 250, despite being nearly the same size, which is suggestive that this increased hierarchy is not solely a function of network size.

While the most obvious difference between GRNEAT and NEAT is the artificial development aspect, it is possible that there is some other factor influencing the development of hierarchical modularity. Therefore, we looked at a different developmental system using a very different
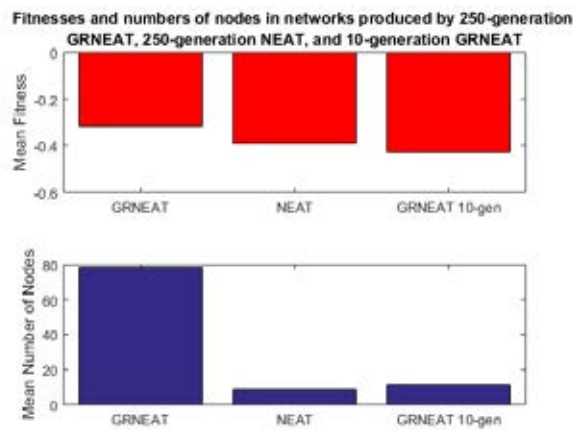
Figure 6: 250 generations of GRNEAT produced top solutions with greater mean fitness than 250 generations of NEAT. 250 generations of NEAT still had greater fitness than 10 generations of GRNEAT.

Figure 7: Networks produced by GRNEAT after 250 generations had greater mean levels of hierarchical modularity than networks produced by GRNEAT after 10 generations. The 95% CIs for GRNEAT and 10-generation GRNEAT were 1.85-2.05 and 1.28-1.72. Number of trials N = 20, p = 0.0014

mechanism of development, GENRE, and compared it to the direct encoding algorithm packaged with its implementation on the GENRE homepage (Hornby, 2001), as discussed in the Methods section. One useful aspect of looking at GENRE's brick-table structures is that the default fitness function for these structures encourages minimizing the number of blocks while maximizing other structural criteria. While, because of the nature of the block structures, the GENRE network representations were much larger than the GRNEAT or NEAT representations, they were actually smaller than those of the direct encoding alternative (640 blocks vs 768 blocks). Therefore, the possibility of network size being a major contributor to the different levels of hierarchy produced by a developmental vs a direct encoding is addressed.

As can be seen in Fig.9, there is a statistically significant (p = 0.0246) difference between the levels of hierarchical modularity produced by GENRE as compared to its nondevelopmental alternative, where the GENRE-produced structures had an average of 4.7 levels, and the others had an average of 4.2.

Notably, in both cases, the number of levels was far higher than for GRNEAT or NEAT, regardless of development, and the network sizes were much larger, which suggests that network size may play some role in the number of levels of hierarchical modularity. However, the fact that GENRE structures have more levels than do a nondevelopmental algorithm optimizing for the same fitness function, in the same number of generations, despite being 17% smaller, provides further evidence that the developmental encoding is doing some of the work in the emergence of this hierarchy.

Conclusion and Future Work

This work opens up two different areas of study. One is the study of the effects of developmental encodings on the emergence of modularity. This is an underexplored area, with the potential to contribute to our understading of the emergence of modularity in general. To our knowledge, this is the first time that developmental encoding effects on the emergence of modularity have been quantified. Another is the study of hierarchical modularity. While modularity is an active area of research, as we outlined earlier in this paper, the hierarchical aspect of modularity has been ignored in modularity-emergence studies despite the importance of hierarchy in biological and other understandings of modularity. This paper takes a first step toward remedying this oversight.

Our results suggest several more specific avenues for future work. It may be useful to compare other developmental systems to similar nondevelopmental ones, to see whether the same effect is observed. Even though GENRE and GRNEAT use very different mechanisms for encoding development, increasing the number of systems studied may provide more evidence that the effect seen here is mechanism-independent. Another possibility would be to adjust different parameters within the developmental systems, to see if other factors can be identified that promote the emergence of hierarchical modularity. Finally, it would be interesting to study the effects of development in biological systems, as was done in (Kashtan et al., 2007; Parter et al., 2007) to determine the effect of varying goals on modularity.
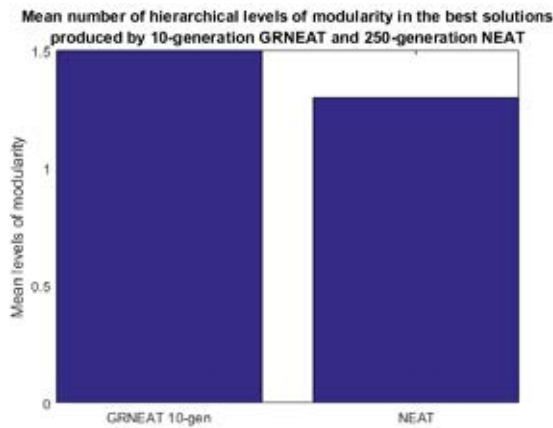
Figure 8: The mean number of levels of hierarchical modularity produced by only 10 generations of GRNEAT was higher than that produced by 250 generations of NEAT. The 95% CIs for 10-generation GRNEAT and NEAT were 1.28-1.72 and 1.09-1.51. Number of trials N = 20, p = 0.2
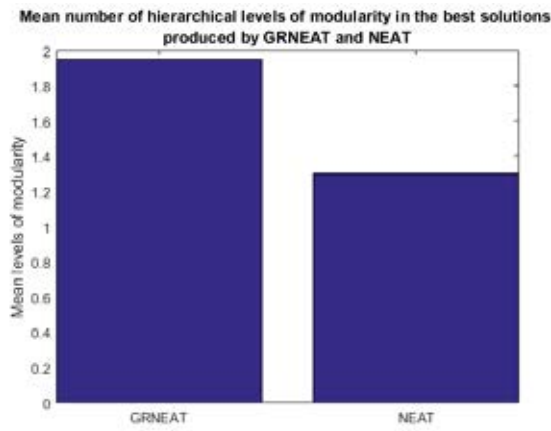

Figure 9: The mean number of levels of hierarchical modularity produced by a developmental brick-table-evolving algorithm, GENRE, was higher than that produced by the nondevelopmental algorithm with which it is packaged. The 95% CIs for GENRE and the nondevelopmental algorithm were 4.4-5.0 and 3.94-4.46. Number of trials N = 10, p = 0.0246

Acknowledgments.

References

Banzhaf, W. (2003). On the dynamics of an artificial regulatory network. In *Advances in Artificial Life*, pages 217–227. Springer.

Bastian, M., Heymann, S., Jacomy, M., et al. (2009). Gephi: an open source software for exploring and manipulating networks. *ICWSM*, 8:361–362.

Bentley, P. J. and Kumar, S. (1999). Three ways to grow designs: A comparison of embryogenies for an evolutionary design problem. In *GECCO*, volume 99, pages 35–43.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Bolker, J. A. (2000). Modularity in development and why it matters to evo-devo. *American Zoologist*, 40(5):770–776.

Calcott, B. (2014). Chaining distinct tasks drives the evolution of modularity. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 701–702.

Clune, J., Mouret, J.-B., and Lipson, H. (2013). The evolutionary origins of modularity. *Proceedings of the Royal Society of London B: Biological Sciences*, 280(1755):20122863.

Cussat-Blanc, S., Harrington, K., and Pollack, J. (2015). Gene regulatory network evolution through augmenting topologies. *Evolutionary Computation, IEEE Transactions on*, 19(6):823–837.

Doursat, R. (2009). Organically grown architectures: Creating decentralized, autonomous systems by embryomorphic engineering. In *Organic Computing*, pages 167–199. Springer.

Guo, H., Meng, Y., and Jin, Y. (2009). A cellular mechanism for multi-robot construction via evolutionary multi-objective optimization of a gene regulatory network. *BioSystems*, 98(3):193–203.

Hartwell, L. H., Hopfield, J. J., Leibler, S., and Murray, A. W. (1999). From molecular to modular cell biology. *Nature*, 402:C47–C52.

Hornby, G. S. (2001). Genre homepage. online, 2001. http://www.demo.cs.brandeis.edu/pr/evo_design/evo_design.html#genre_source.

Hornby, G. S., Lipson, H., and Pollack, J. B. (2001). Evolution of generative design systems for modular physical robots. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 4146–4151. IEEE.

Hornby, G. S. and Pollack, J. B. (2001a). The advantages of generative grammatical encodings for physical design. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, pages 600–607. IEEE.

Hornby, G. S. and Pollack, J. B. (2001b). Evolving l-systems to generate virtual creatures. *Computers & Graphics*, 25(6):1041–1048.

Juille, H. and Pollack, J. B. (1996). Co-evolving intertwined spirals. In *in Proceedings of the Fifth Annual Conference on Evolutionary Programming*. Citeseer.

Kashtan, N. and Alon, U. (2005). Spontaneous evolution of modularity and network motifs. *Proceedings of the National Academy of Sciences of the United States of America*, 102(39):13773–13778.

Kashtan, N., Noor, E., and Alon, U. (2007). Varying environments can speed up evolution. *Proceedings of the National Academy of Sciences*, 104(34):13711–13716.

Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press.

Lang, K. J. (1988). Learning to tell two spirals apart. In *Proc. of 1988 Connectionist Models Summer School*.

Lindenmayer, A. (1968). Mathematical models for cellular interactions in development ii. simple and branching filaments with two-sided inputs. *Journal of Theoretical Biology*, 18(3):300–315.

Lipson, H., Pollack, J. B., Suh, N. P., and Wainwright, P. (2002). On the origin of modular variation. *Evolution*, 56(8):1549–1556.

Lowell, J. and Pollack, J. (2014). The effect of connection cost on modularity in evolved neural networks. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 726–733.

Miller, J. F. and Banzhaf, W. (2003). Evolving the program for a cell: from french flags to boolean circuits. *On Growth, Form and Computers*, pages 278–301.

Newman, M. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113.

O'Neill, M. and Ryan, C. (2001). Grammatical evolution. *IEEE Transactions on Evolutionary Computation*, 5(4):349–358.

Parter, M., Kashtan, N., and Alon, U. (2007). Environmental variability and modularity of bacterial metabolic networks. *BMC evolutionary biology*, 7(1):169.

Scherrer, A. (2008). Matlab louvain implementation. online, 2008.

Schilling, M. A. (2002). Modularity in multiple disciplines. *Managing in the modular age: Architectures, networks and organizations*, pages 203–214.

Simmerson, M. (2006). Neat4j homepage. online, 2006.

Simon, H. A. (1996). *The sciences of the artificial*. MIT press.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127.

Thompson, A. (2012). *Hardware Evolution: Automatic design of electronic circuits in reconfigurable hardware by artificial evolution*. Springer Science & Business Media.

Tufte, G. (2008). Phenotypic, developmental and computational resources: scaling in artificial development. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 859–866. ACM.

Vassilev, V. K., Job, D., and Miller, J. F. (2000). Towards the automatic design of more efficient digital circuits. In *Evolvable Hardware, 2000. Proceedings. The Second NASA/DoD Workshop on*, pages 151–160. IEEE.

Whitacre, J. M., Rohlfshagen, P., Bender, A., and Yao, X. (2010). The role of degenerate robustness in the evolvability of multiagent systems in dynamic environments. In *Parallel Problem Solving from Nature, PPSN XI*, pages 284–293. Springer.

Dynamic Structure Discovery and Repair for 3D Cell Assemblages

Giordano B. S. Ferreira[1], Max Smiley[3], Matthias Scheutz[1] and Mike Levin[2]

[1]Department of Computer Science, Tufts University, Medford MA, USA
[2]Department of Biology, Tufts University, Medford MA, USA
[3]Microsoft Research, Seattle WA, USA

Abstract

Many organisms can regenerate their bodies, but it is currently unclear how they accomplish this feat. In this paper, we introduce a cell-to-cell communication mechanism that allows a 3D arrangement of cells to discover its structure and maintain it in the light of random cell death, even at very high death rates. We report results from simulations of an agent-based model that demonstrate the effectiveness of the proposed approach for Planarian worm-like shapes, but the proposed model is general and applies to any shape.

Introduction

Biological organisms have the ability to regenerate themselves (Birnbaum and Alvarado, 2008), i.e., they are able to detect and reproduce damaged cells that make up their morphological structure. In some cases, whole body parts (e.g., limbs, tail, etc.) can be regenerated and the question arises how this information is encoded and where it is stored (Friston et al., 2015; Pezzulo and Levin, 2015)? While current orthodoxy would still point to genetic encodings and thus morphological information being stored in and recovered from gene expressions, there is converging evidence that this might not be so, at least not in all cases (see the next section). Some of the evidence (reviewed in Lobo et al. (2014)) comes from studies where morphological changes performed on organisms were regenerated after they were lesioned (e.g., damage to deer antlers can result in ectopic growths at the same spot of injury and these growth persist through several subsequence shedding and regenerations of the deer's antlers (Bubenik, 1990)). Since there were no opportunities for genes to encode those initial morphological changes, the information must have been stored elsewhere. But if morphological information is not stored genetically, what other mechanisms could be accountable for representing the morphological structure of an organism?

In this paper, we propose a dynamic messaging mechanism that while not yet mapped on biological substrate can functionally explain how morphological information can be obtained, stored, and used to repair structural damages to organisms. Specifically, different from genetic encodings where information is local to each cell, statically encoded in the gene and thus retrievable only locally, the proposed mechanism is distributed, dynamic, and integrates information across cells. Hence, it is able to detect when cells are missing in a structure and start a regrowth process that generates exactly and only the missing parts. We will demonstrate the operation of the mechanism using an agent-based model of cell-to-cell communication and prototypical 3D organismal shape of a flatworm and show that for various rates of random cell destruction (e.g., due to radiation) the organism is able to maintain its structure. In concluding, we discuss next steps for further simulations and validations of the employed principle.

Background and Previous Work

One of the major questions facing biology and biomedicine is how groups of cells cooperate to build and maintain complex anatomical structures. In many animals, this occurs over long time-spans, counteracting aging, carcinogenic transformation, and tissue abrasion. An understanding of the information structures and algorithms that keep cells orchestrated towards maintaining a large-scale bodyplan would be very important for regenerative medicine, aging research, and degenerative disease, as well as having basic implications for understanding pattern regulation in evolution (Ingber and Levin, 2007).regenerative biology is the Planarian flatworm – a complex bilaterian organism that regenerates amputated pieces, and continuously maintains its bodyplan despite significant turnover and remodeling (Oviedo et al., 2003). While progress is being made with models of gene regulation (Lobo and Levin, 2015), we still seek testable models of cellular communication that explain pattern memory (Tosenberger et al., 2015). Deriving generative, fully-specified models of pattern regulation in this kind of model species is an essential goal for converting molecular-genetic insights into actionable strategies for manipulating growth and form in regenerative medicine, birth defects, cancer, and synthetic bioengineering (Doursat et al., 2013).

The problem of structural maintenance has been ap-

proached by the artificial life community using genetic algorithms, agent-based models and cellular automata to model the behavior of how a single cell could multiply and generate a whole tissue, and after some time, this tissue could maintain its shape against some external or internal perturbations. Andersen et al. (2009), for example, used a genetic algorithm to evolve a gene regulatory network which controls the behaviors of cells. The authors put specific shapes that they want to create in their fitness function, thus the GA could find a network (i.e., genotype), which, starting using a single cell, generates that specific shape (i.e., phenotype). They concluded that different networks can lead to the same phenotype, and more interestingly, the shape become capable of healing wounds even though this process was not encoded in the fitness function.

In Gerlee et al. (2011), the authors use a genetic algorithm to evolve a 3-dimensional cellular automaton that creates and maintains a mono-layer tissue structure. First, the authors wanted to show that a cellular automaton could evolve from a network containing just one cell, to a 2-dimensional structure similar to how epithelial cells are organized in most of the organs of the body. Further, the authors put some external and internal perturbations to verify if the model was capable of returning to its original structure. Similar to our proposed model, each cell is a discrete agent which interacts with its neighborhood, and in their paper, depending on its concentration of oxygen and a generic growth factor. However, in that model the network will grow indefinitely until it reaches a pre-defined area.

Basanta et al. (2008), also used a 3D cellular automaton to model the interaction among neighbors cells and used a genetic algorithm to find a good genotype to perform the procedure. In this work, a 3D shape is created based on the cellular automaton's rule coded in the genes of all cells, and at some point, some genotypes achieve a state of homeostasis. After that, lesions were performed on the shape, and some genotypes were able to regenerate their structures. The authors verified that the organisms which perform best in this "wound recovery" were the ones which had a specific direction by which the cells evolved in the tissue creation.

Overall, past approaches (to the extent that we could find) used some kind of genetic encoding to define how interactions among cells should take place. Thus, cells behaviors depend on their neighborhood and are encoded in the genotype. Our proposed model, on the other hand, does not rely on any genetic encoding, because the behavior of the cells depends on the messages they receive. The critical advantage is that our model does not have any local storage of shape data nor does it rely on it; rather, it can dynamically learn and maintain new morphologies using the same underlying mechanism.

The Communication Model

We start by first presenting the idea of the proposed cell-to-cell communication mechanism, followed by the detailed agent-based model implementing it based on the now customary ODD (Overview, Design concepts, Details) protocol (Grimm et al., 2010).

Discovery and Regeneration

The purpose of the agent-based model is to investigate possible structure monitoring and regeneration process for 3D cell structures, possibly resembling organismal bodies such as the Planarian flatworm. Specifically, we intend to propose mechanisms for such 3D cell structures to *dynamically discover* their morphology and then maintain it indefinitely in the light of random damages happening to parts of it such the damages that occur as part of natural aging. The basic idea is that cells can send messages to other cells or forward messages they receive from other cells which contain information about the path they traveled. This information can then be checked as a packet travels through the body's cells and if a cell along the way is missing, it must have been damaged and thus needs to be repaired. To illustrate how this works, consider the 2D arrangement of cells in Fig. 1.

The packet originating at the bottom right cell during "discovery" (where packets are randomly generated and only those are kept whose paths actually reflect paths that can be taken) consists of three segments of variable length – (0,4), (1,2), and (0,2) – where the first number in each pair indicates a direction (with 0 being West, 1 being SW, 2 being SE, and so on) and the second indicates the distance measured in cells (4 means four cells across). Thus, this packet structure can specify arbitrary paths with up to two directional changes to cover cells in a 2D arrangement.

If that packet were now to retrace its path back to its origin in a lesioned structure and thus could not find the fourth cell in a row as predicted by its (0,4) segment, this detection failure could be used by the cell where the packet got stuck to grow a new cell in the missing position (as the old cell residing there must have died). The regrowth now allows the packet to complete the first segment of its path and the same process of regeneration repeats itself for the second and third segments up until all missing cells along the path have been recovered. Note that not all missing cells were regenerated, only those discovered by the particular packet along its path. For the other missing cells to be regenerated, additional packets with paths going through them would be needed.

The 3D Spatial Agent-Based Model

The proposed ABM model has just one type of agent representing the cells of the organism. Each agent has some attributes that describe them at a given time. Each agent i is defined by an unique identity number i_{id} and its location on the organism's body $< i_x, i_y, i_z >$.

Example packet generated during Discovery stage

Packet: (0,4), (1,2), (0,2)

Direction 0
Distance 4

Direction 1
Distance 2
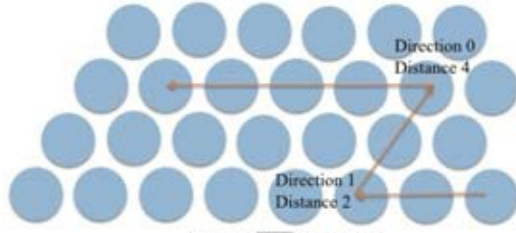
Damage to Structure

Example packet backtracking during Repair stage

Packet: (0,4), (1,2), (0,2)

Direction 0
Distance 4

Direction 1
Distance 2

Direction 0
Distance 2

Packet: (0,4), (1,2), (0,2)

Direction 0
Distance 4

Direction 1
Distance 2

Direction 0
Distance 2

Packet: (1,2), (0,2)

Direction 0
Distance 4

Direction 1
Distance 2

Direction 0
Distance 2

Packet: (0,2)

Direction 0
Distance 4

Direction 1
Distance 2
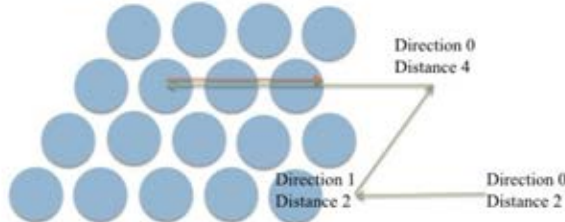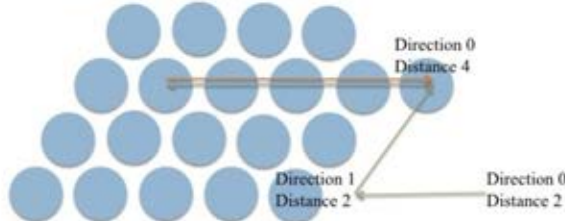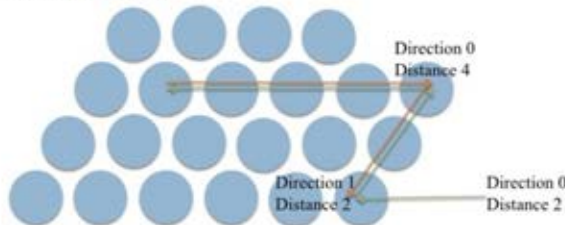
Direction 0
Distance 2
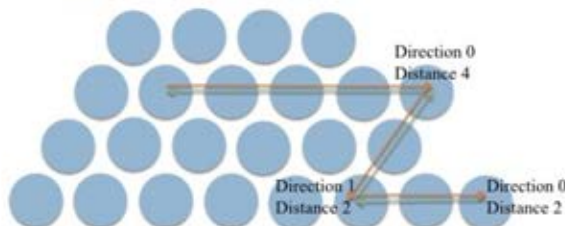
Figure 1: Example of cell discovery, damage detection, and repair.

The specific shape of the evaluation organism, a Planarian worm, is a 3D structure called rhombic dodecahedral honey-

comb. One can imagine each cell as an hexagon with three other hexagons stacked above it and three other hexagons behind. Therefore, each cell is a rhombic dodecahedron hence it has at most 12 neighbor cells which is stored in a list $i_{Neighbors}$.

Cells hold and send packets to its neighbors. A packet β contains a list of vectors β_V, of distance and direction that describe the path that a packet has traveled across the cell network. The packets are organized in temporal order, with the most recent vector at the top of the list. Thus, each cell contains one list containing the packets received from its neighbors during a cycle $i_{ReceivedPackets}$ and a list of packets the cell is holding $i_{HeldPackets}$.

Each vector v has an integer $v_{Distance}$ representing the number of cells the vector traveled through, $v_{Direction}$ representing one of the twelve directions in which the vector traveled, and v_{Mode} which stores whether the packet is charting its path and adding to v or backtracking and taking data from v.

At each cycle, each cell generates $PacketFreq$ packets and sends them to adjacent cells in random directions. When a cell receives a packet, it increments its top vector's distance. For each packet, received in a given cycle, the cell will either (1) send the packet along the same direction as its top vector's direction, (2) send the packet in a new direction, or (3) hold the packet. In order for a cell to hold a packet, this packet must have at least $MinVectorsToHold$, and the top vector must have a distance of at least $MinTopLen$. If the packet is not held, there is a $BendProb$ probability that the packet will be sent in a new direction. This new direction should be different from the opposite direction of the top vector's direction.

When a packet reaches a cell, the cell verifies the number of bends until that moment. If this value is equal to $MinBends$, then the packet will backtrack, regenerating dead cells during this process.

The model runs as a discrete-time simulation for a defined number of cycles, $RunCycles$. And at each cycle, the cells perform only two processes: sensing and acting. In the first they receive packets from their neighbors and decide if the packets will be held or sent (see Algorithm 1). Moreover, if a packet is backtracking, and the neighbor is dead, then that neighbor is regenerated during sensing process. The acting process is just the cells sending packets to their neighbors (see Algorithm 2).

In the proposed model, each cell creates packets to send to its neighbors, the only interaction between agents. This local interaction creates an emergent behavior of structure maintenance where cells along the travel path are restored. As long as some packets will eventually hit each dead cell, the system is guaranteed to keep the structure intact.

Regarding the stochastic procedure, there are two cases where they occur. First when a cell needs to decide the direction of a packet (a new packet or a received packet that

Algorithm 1 Pseudo code of the sensing process performed by the cells.

Sense(i)

  for all $packet \beta \in i.ReceivedPackets$ **do**
    $top \leftarrow \beta.TopVector$
    if $top.Mode == Charting$ **then**
      $top.Distance \leftarrow top.Distance + 1$
      if $\beta.Bends \geq MinVectorsToHold$ **and** $top.Distance \geq MinTopLen$ **and** $isAlive(i.Neighbors[top.Direction])$ **then**
        $i.HeldPackets.add(\beta)$
      else
        if $random() \leq BendProb$ **then**
          $\beta.addVec(getNewDirection(top.Direction))$
        else
          $\beta.addVec(top.Direction)$
        end if
        $i.SendingPackets.add(\beta)$
      end if
    else
      $top.Distance \leftarrow top.Distance - 1$
      if $top.Distance \leq 0$ **then**
        $i.ReceivedPackets.pop()$
      end if
      if $top \neq nil$ **then**
        if $!isAlive(i.Neighbors[reverse(top.Direction)])$ **then**
          $regenerateCell(i, reverse(top.Direction))$
        end if
        $i.SendingPackets.add(\beta)$
      end if
    end if
  end for
  if $i.HeldPackages.size() \geq MinBends$ **then**
    for all $packet \beta \in i.HeldPackets$ **do**
      $\beta.Mode \leftarrow Backtracking$
      $i.SendingPackets.add(\beta)$
    end for
  end if

Algorithm 2 Pseudo code of the acting process performed by the cells.

Act(i)

  for all $packet \beta \in i.SendingPackets$ **do**
    $top \leftarrow \beta.TopVector$
    if $top.Mode == Backtracking$ **then**
      $top.Direction \leftarrow reverse(top.Direction)$
    end if
    if $isAlive(i.Neighbors[top.Direction])$ **then**
      $sendPacket(i, top.Direction, packet)$
    end if
  end for
  $i.SendingPackets.clear()$

---

needs to change direction). The second stochastic procedure is the random death of cells which will be explained in the next section.

Simulation Experiments

The goal of the experimental evaluation was to see whether the proposed cell-to-cell communication mechanism would be sufficient to maintain the structure of an organism over time in light of random cell death. The model was implemented in our Java-based agent-based *SimWorld* simulation environment (Scheutz and Harris, 2011).[1] For all simulations runs, we consider a prototypical 3D Planarian-like structure with a fixed shape of 8 layers containing 339 cells each, resulting in 2712 cells total (the top-most layer of cells of the employed shape is depicted Figure 2).
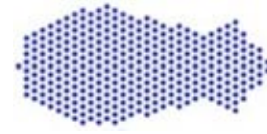
Figure 2: Shape of the topmost layer of the worm containing 339 cells.

To simulate the process of structural deterioration (e.g., due to a toxic or radioactive environment, or the natural aging and death of cells), we fixed a particular cycle in the simulation when this process would start to occur (*Death-Time*=80). At the moment that a cell dies, all held packets are lost and consequently it cannot transmit other packets that reach it later. To verify whether enough of the structure of the organism's body was still intact, we fixed the $Threshold$ as 90% of alive cells for the entire simulation, i.e., for the organism to be considered "intact" at least 90% of its cells must be alive at any given cycle.

The function $isAlive$ verifies if a specific cell is alive and can transmit packages. If a cell tries to send a packet which has its top vector in the $Backtracking$ mode, and the neighbor cell supposed to receive this packet is not alive, then the alive cell calls the function $regenerateCell$ which "revives" the dead neighbor. The function $getNewDirection$ randomly chooses a new direction distinct from the direction passed as a parameter and also distinct from the reverse of this direction, to assure that the packet would not return from the cell it comes. Thus, $reverse$ is a function that given the direction to one side of the dodecahedron, returns the direction to the reverse side of this polyhedron. Finally, the func-

---

[1]SimWorld is a versatile environment with support for graphical and batch runs of models. It is easy to program and easy to extend, and it provides an interactive graphical interface user interface for inspecting agent behavior (and novel mechanisms for playing a simulation forward and backward, which supports the modeller in detecting interesting emergent behaviors). SimWorld has been under development in our lab for over a decade.

tion $sendPackets$ adds the packet to the $ReceivedPackets$ list of the cell that exists on the direction of the top vector.

Simulation runs can terminate in two different cases: for every cycle during the simulation, the organism must have at least $Threshold$ percent alive cells otherwise the simulation stops. The second condition is when the simulation reaches the pre-determined limit of 500 cycles (if the organism can maintain its structure through 420 cycles, then we assume it can do so indefinitely, at least in approximation).

To explore the parameter space of the model, we first varied the probability of a cell dying on a given cycle (*DeathProb*) in order to simulate the death of cells as time passes. For our experiments, *DeathProb* $\in$ $\{0.0, 0.01, 0.02, 0.03, 0.04\}$. For example, with a 2% death rate per cell per cycle, every cell will on average die every 50 cycles or 10 times in the course of the 500 cycle simulation. Since there are 2712 cells in the body, over 54 cells will die on average at any given cycle which is significant structural damage that accrues over time if not repaired quickly.

We also varied the number of new "packets" a cell produces on each cycle (*PacketFreq*). In our experiments, we varied *PacketFreq* $\in \{1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 31\}$.

In order to control the variety of navigation patterns of packets, we varied the minimum size of the vector of bends before a packet can backtrack (*MinBends*); the minimum length the top vector of a packet should be to be able to bend, by adding a vector in new direction (*MinTopLen*); the probability that a packet will bend, given the top vector length is at least *MinTopLen* (*BendProb*). For our experiments, *Min-Bends* $\in \{1, 3, 5, 7\}$, *MinTopLen* $\in \{1, 3, 5, 7\}$ and *Bend-Prob* $\in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$.

As our model has stochastic processes, we need to explore this parameter space using different random number generator seeds. Thus, for each point in the parameter space, we ran 8 different simulations resulting in a total of 47520 simulations. The dependent variable was the number of cycles the simulation ran with more than *Threshold* alive cells.

Results and Analyses

From 50688 different data points that have *DeathProb* > 0.0, 28961 points maintained a rate of 90% of cells alive during the whole simulation, i.e., 500 cycles. More specifically, 11801 points were with $rd = 0.01$, 9685 with $rd = 0.02$, 5802 with $rd = 0.03$ and 1673 with $rd = 0.04$ as shown in Figure 3. These results show that there exists a parameter space in which our model can repair death cells and maintain the individual's structure indefinitely. The mean number of cycles with number of alive cells above 90% for all simulations was 388.521.

In order to compare the main effects of each independent variable on the cycles above threshold, we performed an ANOVA with *PacketFreq*, *MinBends*, *MinTopLen*, *Bend-Prob* and *DeathProb* as independent variables and the *CyclesAboveMin* as dependent variable. The ANOVA shows
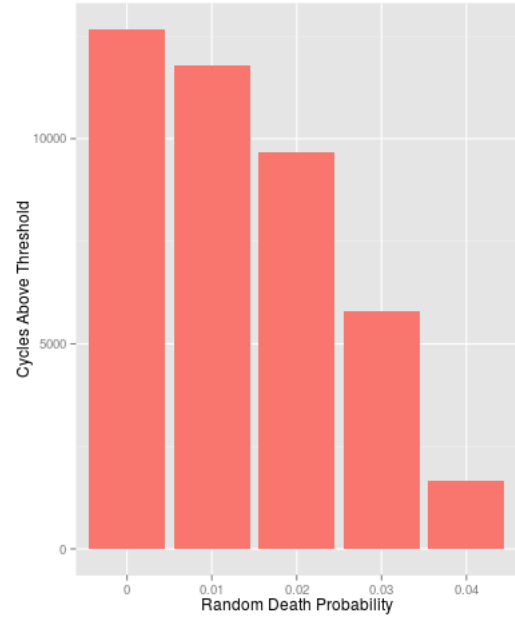
Figure 3: Histogram of points which maintain the structure of the worm after 500 cycles.
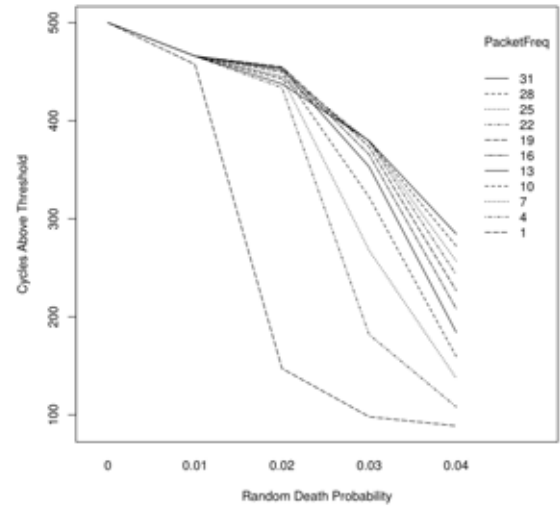
Figure 4: Interaction between RandomDeath and CyclesAboveMin for each number of produced packets per cycle.

significant main effects for all independent variable other than $BendProb$. Significant two-way, three-way and four-way interactions among variables other than $BendProb$ were also found. These results confirm our hypothesis that variance in the packet vector is not relevant for the process

of structure maintenance.

As expected, there was a significant negative effect between *DeathProb* and *CyclesAboveMin* as shown in Figure 4. We also found a positive correlation with *PacketFreq* and *CyclesAboveMin* as shown in Figure 5. Increasing the value of *PacketFreq* means more variations of possible packets are explored, and from a certain point on increasing this value will generate more redundant packets than novel ones, maintaining an asymptote. The point by which the performance does not change depends on the probability of cell death, because as we increase the probability of cells deteriorating, more packets are not redundant; more specifically, more packets are necessary to maintain the structure of the organism.
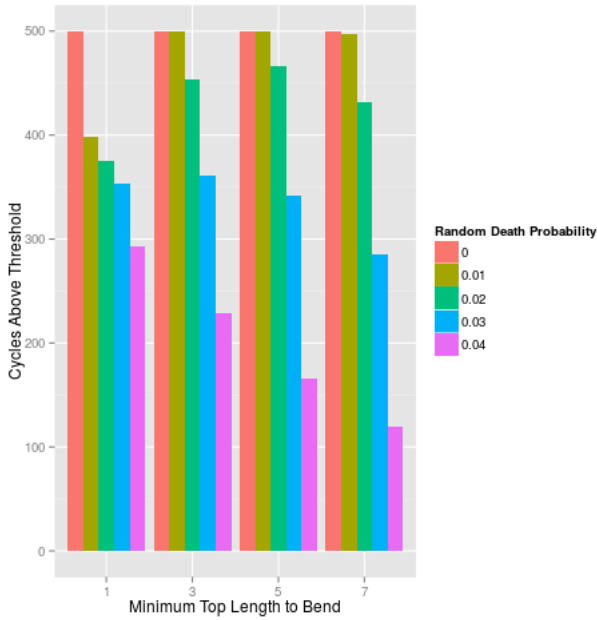
Figure 5: Interaction between PacketFreq and CyclesAboveMin for each probability of a random cell death.

Regarding the *MinBends* (see Figure 6), a moderate level (*MinBends*= 3) of minimum bends (before a packet can backtrack performed best. This value shows the optimization between the tradeoff of a longer packet covering a large area of the individual but also being more at risk of losses happening due to random cell deaths.

Figure 7 shows the interaction between *MinTopLen* and *CyclesAboveMin*. It is important to note that as *MinTopLen* increases, the length of the packet increases, therefore the packet must spend more time traversing before backtracking. Consequently, it increases the chance of a packet being lost to random death before it can repair another dead cell. Thus, for lower values of *RandomDeath*, it is good to have high values of *MinTopLen*. However, increasing *RandomDeath* confirms the tradeoff between better coverage
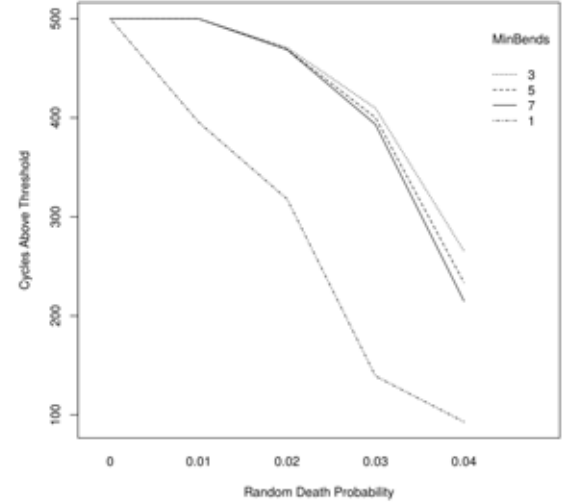
Figure 6: Interaction between MinBends and CyclesAboveMin for each probability of a random cell death.

and the risk of longer packets shifts, and for *RandomDeath* = 0.04, *MinTopLen* = 1 performs best.

The interaction between *MinTopLen* and *MinBends* shows an optimal combination with *MinTopLen* = 1 and *MinBends* = 3 (see Figure 8). Inverting the values of the two variables reduces the performance, even though these two combinations yield the same total packet length. The explanation for this is that the same length of packet can cover a wider space if it has more bends. This tradeoff is most pronounced when changing from a single bend to two bends.

Discussion

Our results show that organisms were able to maintain their structure using the proposed cell-to-cell communication mechanism for the right set of parameters: a high *PacketFrequency* > 22, a moderate *MinBends* = 3, a low *MinToplen* = 1, and the value of *BendProb* not being relevant. Without modifying the algorithm, we hypothesize that it is possible to regenerate the worm from various more systematic cuts as well where a large part of the body is removed. For such a lesion to be healed, the packets residing in alive cells in the remaining body would have to be such that their collective paths would cover all excised cells which would then be regenerated during backtracking.

Space limitations allowed us to only discuss one particular structure but the proposed mechanisms are general enough to work for a very large set of structures. Whether a structure will be maintainable will effectively depend on both how cells die (e.g., randomly or because of lesions cutting of whole segments of the body) and how many bends packets can have which they will need to recover complex struc-
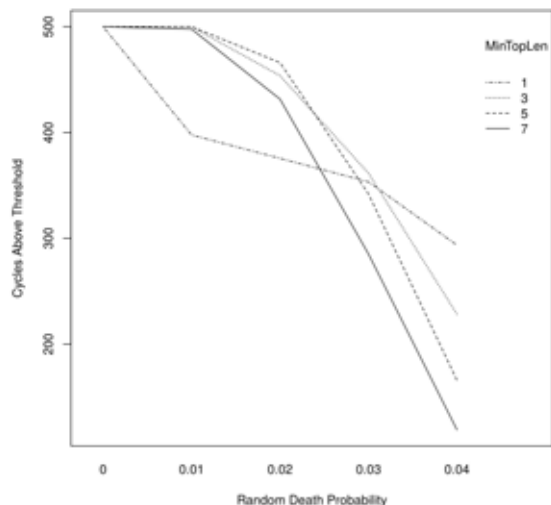
Figure 7: Interaction between MinTopLen and CyclesAboveMin for each probability of a random cell death.

Figure 8: Interaction between MinTopLen and CyclesAboveMin for each minimum number of bends before backtracking.

tures that require many path segments to hit all component cells (e.g., to regenerate a cut-off arm packets need to travel through the upper arm, the lower arm, the wrist, the palm, and the various finger segments, thus requiring a larger number of segments in the packet).

Bi-directional cell communication in vivo takes place via several kinds of physical media (Edelstein et al., 2016), including chemical signals (diffusible molecules), physical forces (pressures and tensions), and bioelectric signaling (voltage gradients) (Levin, 2012). The latter is especially interesting because it enables many of the functions described in our model (Funk, 2013). Indeed, brains evolved by specializing such communication functions that were present from the dawn of multicellularity, and optimizing it for communication and message-passing functions in the central nervous system (Keijzer et al., 2013). The more ancient form, developmental bioelectricity (Bates, 2015)m is a modality by which collections of cells communicate, store memory, and make group decisions about growth and form during embryogenesis and regeneration (Pezzulo and Levin, 2015). Using proteins such as ion channels and pumps, cells regulate their bioelectric dynamics (Levin, 2014; Mustard and Levin, 2014). However, using electrical synapses (gap junctions), cells can detect the presence and physiological state of neighbors (Palacios-Prado and Bukauskas, 2009). Communication via gap junctions has recently been shown to exert significant instructive control over growth and form during regeneration in planaria and other model systems (Emmons-Bell et al., 2015).

Conclusion

In this paper we introduced the first agent-based model of structure discovery and repair which allows 3D cell structure to *discover their organization and repair it from damages occurring due to cell death*. We demonstrated the efficacy of the mechanism in large set of simulations of random cell death occurring at different rates in simulated body shaped as a Planarian. For even high cell death rates, we found parameters for the proposed cell-to-cell communication mechanisms that could maintain the structure indefinitely.

As a next step, we would like to verify how the model behaves with non-equally distributed cell death, i.e., where a cluster of adjacent cells dies at the same time due to, for example, the action of some toxin or an impact on a specific area of the organism. If for all dead cells there is a remaining packet held by an alive cell, then all cells can be regenerated.

In addition, we intend to investigate the regeneration from cuts that *in vivo* worms present. It is well known that the Planaria is capable of regeneration from cuts $\frac{1}{279^{th}}$ of the intact animal volume (Morgan, 1898). Our hypothesis is that there exists a parameter assignment by which our model is capable of regenerating structure from simultaneous death of a large area of cells and also from any number of cuts for the right set of communication parameters.

References

Andersen, T., Newman, R., and Otter, T. (2009). Shape homeostasis in virtual embryos. *Artificial life*, 15(2):161–183.

Basanta, D., Miodownik, M., and Baum, B. (2008). The evolution of robust development and homeostasis in artificial organisms. *PLoS Comput Biol*, 4(3):e1000030.

Bates, E. (2015). Ion channels in development and cancer. *Annual review of cell and developmental biology*, 31:231–247.

Birnbaum, K. D. and Alvarado, A. S. (2008). Slicing across kingdoms: regeneration in plants and animals. *Cell*, 132(4):697–710.

Bubenik, G. A. (1990). The role of the nervous system in the growth of antlers. In *Horns, pronghorns, and antlers*, pages 339–358. Springer.

Doursat, R., Sayama, H., and Michel, O. (2013). A review of morphogenetic engineering. *Natural Computing*, 12(4):517–535.

Edelstein, L., Fuxe, K., Levin, M., Popescu, B. O., and Smythies, J. (2016). Telocytes in their context with other intercellular communication agents. In *Seminars in Cell & Developmental Biology*. Elsevier.

Emmons-Bell, M., Durant, F., Hammelman, J., Bessonov, N., Volpert, V., Morokuma, J., Pinet, K., Adams, D. S., Pietak, A., Lobo, D., and Levin, M. (2015). Gap junctional blockade stochastically induces different species-specific head anatomies in genetically wild-type girardia dorotocephala flatworms. *International journal of molecular sciences*, 16(11):27865–27896.

Friston, K., Levin, M., Sengupta, B., and Pezzulo, G. (2015). Knowing one's place: a free-energy approach to pattern regulation. *Journal of The Royal Society Interface*, 12(105):20141383.

Funk, R. H. (2013). Ion gradients in tissue and organ biology. *Biol Syst Open Access*, 2(105):2.

Gerlee, P., Basanta, D., and Anderson, A. R. (2011). Evolving homeostatic tissue using genetic algorithms. *Progress in biophysics and molecular biology*, 106(2):414–425.

Grimm, V., Berger, U., DeAngelis, D. L., Polhill, J. G., Giske, J., and Railsback, S. F. (2010). The odd protocol: a review and first update. *Ecological modelling*, 221(23):2760–2768.

Ingber, D. E. and Levin, M. (2007). What lies at the interface of regenerative medicine and developmental biology? *Development*, 134(14):2541–2547.

Keijzer, F., Van Duijn, M., and Lyon, P. (2013). What nervous systems do: early evolution, input-output, and the skin brain thesis. *Adaptive Behavior*, page 1059712312465330.

Levin, M. (2012). Morphogenetic fields in embryogenesis, regeneration, and cancer: non-local control of complex patterning. *Biosystems*, 109(3):243–261.

Levin, M. (2014). Endogenous bioelectrical networks store non-genetic patterning information during development and regeneration. *The Journal of physiology*, 592(11):2295–2305.

Lobo, D. and Levin, M. (2015). Inferring regulatory networks from experimental morphological phenotypes: A computational method reverse-engineers planarian regeneration. *PLoS Comput Biol*, 11(6):e1004295.

Lobo, D., Solano, M., Bubenik, G. A., and Levin, M. (2014). A linear-encoding model explains the variability of the target morphology in regeneration. *Journal of The Royal Society Interface*, 11(92):20130918.

Morgan, T. H. (1898). Experimental studies of the regeneration of planaria maculata. *Development Genes and Evolution*, 7(2):364–397.

Mustard, J. and Levin, M. (2014). Bioelectrical mechanisms for programming growth and form: taming physiological networks for soft body robotics. *Soft Robotics*, 1(3):169–191.

Oviedo, N. J., Newmark, P. A., and Sánchez Alvarado, A. (2003). Allometric scaling and proportion regulation in the freshwater planarian schmidtea mediterranea. *Developmental dynamics*, 226(2):326–333.

Palacios-Prado, N. and Bukauskas, F. F. (2009). Heterotypic gap junction channels as voltage-sensitive valves for intercellular signaling. *Proceedings of the National Academy of Sciences*, 106(35):14855–14860.

Pezzulo, G. and Levin, M. (2015). Re-membering the body: Applications of computational neuroscience to the top-down control of regeneration of limbs and other complex organs. *Integrative Biology*, 7(12):1487–1517.

Scheutz, M. and Harris, J. (2011). An overview of the simworld agent-based grid experimentation system. In Werner, D., Kurowski, K., and Schott, B., editors, *Large-Scale Computing Techniques for Complex System Simulations*. Wiley.

Tosenberger, A., Bessonov, N., Levin, M., Reinberg, N., Volpert, V., and Morozova, N. (2015). A conceptual model of morphogenesis and regeneration. *Acta biotheoretica*, 63(3):283–294.

Evolved Developmental Strategies of Artificial Multicellular Organisms

Jean Disset[1], Sylvain Cussat-Blanc[1] and Yves Duthen[1]

[1]University Toulouse 1 Capitole - IRIT
{disset; cussat; duthen}@irit.fr

Abstract

We present the use of a new computationaly efficient 3D physics model for the simulation of cells in a virtual aquatic world. In this model, cells can freely assemble and disconnect along the simulation without any separation between the development and evaluation stages, as is the case in most evo-devo models which only consider one cell cluster. While allowing for the discovery of interesting behaviors through the addition of new degrees of freedom, this 3D center-based physics engine and its associated virtual world also come with their drawbacks when applied to evolutionnary experiments: larger search space and numerous local optima. In this paper, we have designed an experiment in which cells must learn to survive by keeping their genome alive as long as possible in a demanding world. No morphology or strategy is explicitly enforced; the only objective the cells have to optimize is the survival time of the organism they build. We show that a novelty metric, adapted to our evo-devo matter, dramatically improves the outcome of the evolutionary runs. This paper also details some of the developmental strategies the evolved multicellular organisms have found in order to survive.

Introduction

Over the past two decades, the artificial life community has seen the development of several models for the simulation of environments in which cells can freely evolve. Many 2-dimensional models have been used, mainly for their simplicity and their computational efficiency, (Doursat, 2009; Joachimczak et al., 2013), but also because they are often sufficient to let interesting cellular behaviors emerge. With the addition of the third dimension come both large possibilities in the exploration of artificial life and the exciting opportunity to more precisely compare and understand real world observations. While there are several 3D physics engines and simulators developed specifically for artificial life (Joachimczak and Wróbel, 2011; Fontana and Wróbel, 2013; Doursat and Sánchez, 2014; Cheney et al., 2014), combining low scale features of cells with efficient simulation at the scale of a whole organism can prove challenging. It requires either ignoring interesting aspects of cells such as their polarisation system, complex adhesive properties or variable stiffnesses, or abandoning computational efficiency.

Of course, many models of cellular simulations are not directly linked to the aLife community (although some have been used for artificial life experiments) and are more tightly related to bio-simulation, strongly focusing on the realism of the simulations they produce. Over the years, many models have been developed using various approaches, among which 2D lattice based cellular automata (Ouchi et al., 2003), various off-lattice 3D center-based models and even precise hybrid multi-scale systems which combine cell-level deformations as well as tissue-scale constraints (Lowengrub et al., 2009), to cite just a few. In the context of artificial life, and specifically when growing multicellular artificial organisms, the complexity of the simulated world directly impacts the developmental strategies and possible morphologies of the creatures. As this can make for some behaviors and strategies that are more desirable and might also help in the understanding of real-life behaviors by bringing more realism, it also comes with at least two obvious trade-offs. First, adding realism and complexity to the artificial world will often increase the required computational power, which is a resource of prime importance when using genetic algorithms that require the simulation of thousands of instances of these worlds. Secondly, and still in the context of artificial evolution, adding complexity to the world can dramatically broaden the search space, requiring even more simulations for evolutionary algorithms to come up with a convincing organism, and further complexifying the fitness landscape. It can thus be argued that the simulation of cells for the growth of artificial multicellular organisms is, while sharing obvious common roots, a different problem than the simulation of real world cells. In this context, while we take our inspiration from biology when designing a cell simulation engine, it is of prime importance to keep these trade-offs in mind and to try and see where the truly desirable features lie, those from which an evolved multicellular organisms might benefit, and those that can be simplified.

In this work, we propose to set up artificial life experiments in a 3-dimensional world using a fast cellular physics engine tailored to artificial life, MecaCell, that offers dynamic cell-cell interactions such as collision, adhesion and

volume conservation approximation while keeping the computational cost in reasonable limits. We have designed an experiment in which the virtual multicellular organism will have to face many local optima created by both the added degrees of freedom and the rules of the world in which it evolves. We show how novelty search with a morphology metrics can, when used in conjunction with a fitness function, help overcome many of these local optima. The experiment we present in this paper challenges one cell to preserve its genetic material in a sea-like environment as long as possible. In order to do so, the cell (which can choose to eventually become an organism after division) will have to face harsh conditions where energy is a difficult resource to harvest. Organisms, or rather same-DNA cell colonies, will thus have to balance their in-water morphology to collect light energy while maintaining solid roots in the ground in order to collect a second essential type of energy. While division of labor might play a determining role in the survival of the colony (harvesting nutrients and light, sharing energy, maintaining the structure of the organism), the rules of the simulated world should make for the appearance of different viable strategies. In the lineage of our previous work (Disset et al., 2014), and to reduce the clues provided by a heavily engineered fitness function as much as possible, the cell controllers, based on gene regulation, are only evolved for survival (duration of the simulation). In addition, we study the impact of a novelty search criterion.

Simulated world

This section presents the different aspects of the simulated world we propose to investigate[1]. The main goal is to try various characteristics of the physics engine and to explore ways to mitigate the adverse effect of added degrees of freedom (comparatively to a 2D simulator or a 3D cell simulator which doesn't account for precise dynamic adhesions, for example). We want our virtual organisms to be able to evolve efficient and varied solutions to the problem of survival in a constrained environment.

Cell physics - MecaCell

MecaCell[2] aims to be an artificial life friendly and generic platform for the 3D simulation of cells. Its goal is to provide a continuous physics environment that is computationally efficient and versatile enough to tackle various ALife experiments and configurations (with exotic or simplified physics rules, for example).

---

[1]All the source code as well as images and videos are available at https://github.com/jdisset/seacells

[2]MecaCell is written in C++ and available (under LGPL license) at https://github.com/jdisset/MecaCell. It includes a custom OpenGL display engine with a plugin system for the extensibility of its interface.

Cell and volume conservation   In MecaCell, each cell is an agent represented by a center, a membrane and an orientation. A cell can freely evolve in a 3D continuous environment, where it will collide and adhere with other cells. Here we consider cells to be spherical objects filled with a mostly incompressible fluid and wrapped in an elastic membrane. Every cell has a rest radius R_r and a dynamic radius R_d. The dynamic radius was introduced to enable an approximation of volume conservation: at each time step t, if a cell is cut (overlapping either another cell or a 3D object), we recompute both its membrane surface area A_t and its current volume V_t. The net difference in volume (relatively to its rest value V_r) is then translated into a pressure stress p_t:

$$p_t = \frac{I \times (V_t - V_r)}{A_t}$$

where I is the compressibility coefficient of the cell. Cell pressure acts as a force governing R_d's growth. When pressure increases under stress, the cell will compensate by expanding its radius in order to recover its original volume. This variation naturally implies a modification of its current membrane surface area A_t, which will also act as a shrinking force on the dynamic radius. The cell membrane is thus, in a computationally efficient manner, brought into equilibrium between volume conservation and surface area conservation, using the following explicit integration scheme:

$$R_{dt} = R_{dt-1} + \Delta t^2 \times (\Delta V - \Delta A - \frac{dR_d}{d_t} \times C)$$

where ΔV is the volume variation $V_t - V_r$, ΔA is the surface area variation $A_t - A_r$ and C is a damping coefficient.

Collisions   In this model, collisions are easily handled by detecting two overlapping cells and by computing the normal and the area of the resulting contact plane. Each cell will then push on the other perpendicularly to this plane and according to their internal pressure (resulting from their deformation). The intensity of the force applied between a cell C_a of internal pressure p_a and a cell C_b (with internal pressure p_b) through a contact plane of area A_c is given by:

$$||\vec{F}|| = A_c \times (P_a + P_b)$$

$$\text{with } P_i = \begin{cases} 0, & \text{if } p_i < 0 \\ p_i, & \text{otherwise} \end{cases}$$

A tunable damping term C_{col} is also added.

Adhesions   When wanting to simulate artificial multicellular organism in a 3D environment, the capability to maintain oriented connections is of prime importance. In MecaCell, cell-cell adhesions use the same kind of contact planes than for the collisions. A cell can choose its adhesive properties distribution across its membrane through the definition of an

adhesion function f_{adh} which associates an adhesive receptor density d_{adh} to a unit vector expressed in the local coordinate system of the cell (and represents the adhesive potential at a given membrane location). We simulate an adhesion between two cells by the creation of a dynamic mass-spring-damper system of length 0, attached to the centers of the contact surfaces on both cells membranes. This spring acts on both membranes but all of the generated forces and momentum is applied at the respective cells centers. When the two adhesive cells get closer from each other, the centers of the adhesion planes are updated, as well as all the mechanical properties of the adhesion mass-spring-damper system. The stiffness K and damping coefficient C are proportional to the contact plane surface area as well as the average receptor density on said surface (and to the intrinsic characteristics of these receptors, which can be different for every cells, or favor certain cell-cell affinities between cellular types). When two adhesive cells are pulled (or rotated) apart, the adhesive dynamic mass-spring-damper system can elongate up to a certain length defined by the maximum length reachable by an adhesion receptor. Thus, if the cells are pulled apart too strongly (relatively to the strength of their connection), they can actually come out of contact again. Similarly, if they experience a torque of too much intensity or a shear stress above a certain threshold, they will be able to slide on each other's membrane (the centers of their adhesion plane will have moved too far apart due to rotation).

Environment - Ground and sea

In addition to the cellular physics model presented previously, the world of this particular experiment is divided in two parts: the ground and the sea.

Ground   The ground is a dense medium in which cells cannot easily move. In order to achieve this effect, we used a special integrator which does not take into account any inertia term, using only the force exerted on each cell to compute its next position. This ground acts as a solid when the forces exerted by the cells are below a certain threshold, only allowing cells to move if they push hard enough. This is, although in a simplified manner, a depiction of the mechanical characteristics of dense mud.

The ground contains nutrients, which are not available in the water. They are present in the mud at various depth, in small areas and finite amounts. At the beginning of the simulation, we initialize $N = 200$ nutrients sources. For a given nutrient source i placed at a random position (x_i, y_i, z_i) in the mud, the initial amount of nutrient n_i is given by:

$$n_i = Q_n \times (1 + C_n \times |y_i|^{P_n})$$

where Q_n is a constant and P_n and C_n are two parameters that determine how the amount of nutrient varies for each nutrient source according to its depth y_i. This is meant to

mimic how the nutrient distribution can be different according to the type of soil. It also allows for the tuning of some aspects of the fitness landscape: with $C_n < 0$, the selective pressure would force the cells to expand laterally while a positive value of C_n should favor a vertical growth to find more reliable sources of nutrients. In this particular experiment, we use $Q_n = 0.03$, $P_n = 1.5$ and $C_n = 0.035$. These values have been chosen empirically in order to create an environment in which organisms can easily survive for a short amount of time but must develop complex strategies to survive longer.

Sea   The second layer of the world is placed on top of the ground. We call it water, because its mechanical characteristics, namely density and viscosity, are supposed to mimic those of a still body of water. Here, a classic semi-implicit Euler integration scheme is used to update the cell positions and orientations. For computational efficiency purposes, no flows are simulated in this water. However, the cells are all slightly buoyant which means that they need to keep adhesions to cells that are still inside the mud in order to avoid being taken away.

Light is abundantly available in the water but stopped by the ground. It only comes in straight rays, perpendicular to the ground, and if one light ray shines upon a given cell, it won't be able to reach any other cell below that first one. In other words, cells block light and their shadows prevent other cells to be lit. We implemented this feature using a classical depth-buffer and depth-culling algorithm.

Cells

Cell life cycle   In order to survive in this world, a cell has to fulfill one requirement: all its energy levels must stay above zero. In this particular experiment, a cell needs to handle two forms of energy: light and nutrient. At the initialisation stage, we place one unique "seed" cell in the mud, just below the water (precisely one cell diameter deep). When the simulation starts, the seed cell has maximum levels of light and nutrient, mimicking the seed endosperm (which provide the initial energy to the seed). At each time step, every cell consumes a fixed amount of light and nutrient energy. When any of the two levels of energy reach 0, the cell dies.

We implemented a simplified cell cycle in which every cell can choose between three actions: growth, quiescence, apoptosis. This lifecycle is controlled by an aGRN that will be detailed at the end of this section. When in quiescent mode, the cell consumes normal amount of nutrients and light. When choosing apoptosis, the cell will disappear and all the nutrients and light it contained will be lost. When a cell enters its growth phase, it will grow (while consuming 20% more energy) until its volume has doubled; at which point division will happen along a particular axis, determined by the cell's aGRN. When division occurs, the mother cell is replaced by two identical daughter cells whose energy

levels are exactly half those of the mother cell at the time of division. Only one variable, the age of the cell, differs between the two daughters cells: one is kept, the other is restarted at zero. This variable is incremented at each time step and is an input to the cells' aGRN.

Energy Nutrients and light are not available at the same place, which means the cells of our organism need to be able to absorb nutrients and light and share that energy with each other. More generally, a cell with large quantities of energy should be able to transfer part of it to any cell in need. In this experiment, we approximate this process through a passive diffusion based on Darcy's law, which describes the flow of an incompressible fluid throughout a porous isotropous medium in the laminar case (which is arguably the case here given the low Reynolds numbers involved). The energy (nutrient or light) flow F_n between two connected cells a and b is thus described by the following equation:

$$F_n = \frac{-k \times A \times \Delta p}{\mu L}$$

where Δp is the energy's pressure drop (here approximated by the difference in levels $n_b - n_a$ or $l_b - l_a$ where n_x and l_x are respectively the nutrient and the light level of cell x) between cell b and cell a. This flow is also determined by the intrinsic permeability of the medium k, the viscosity of the nutritive fluid μ as well as the connection area A and the distance L between the two cells centers. The value of this flow is computed at each time step for each active connection (i.e. real adhesions) between two cells using an explicit integration scheme. Using the free surface area of a cell's membrane, we also use this diffusion system to simulate the absorption of both light and nutrients from the environment. Any lit cell will perceive a light intensity proportional to its elevation (above the ground) until a certain altitude where this intensity is capped to one. Inside the ground and from any cell positioned at $\vec{P}_c$, the available nutrients concentration A_s coming from a nutrient source s at position $\vec{P}_s$, with current absolute content in nutrient C_t, initial diffusion radius of R_{t0} and an initial content of C_{t0} is given by

$$A_s = C_t \times (1 - (|\vec{P}_s - \vec{P}_c|/R_{t0} * (C_t/C_{t0})) * C_t/C_{t0})$$

Morphogens Bio-inspired communication through the diffusion of molecules in the environment has successfully been used in numerous artificial life experiment and has proven to be an efficient way to enable information transmission between agents. While some authors use detailed and realistic diffusion of signalling molecules, here we use a simple instantaneous diffusion system. Every cell can emit one or several of N_m morphogens through the m_i output protein concentration of its aGRN, and can sense the concentration of each morphogens through its c_i input proteins. The perceived intensity of a morphogen follows an inverse squared law. Thus, for any receiver positioned at $\vec{P}_r$, the perceived intensity I_m of a morphogen m emitted by N sources placed at positions $\vec{P}_{si}$ with intensity E_{mi} is given by:

$$I_m(\vec{P}_r) = \sum_i^N \frac{E_{mi}}{A_m \times ||\vec{P}_{si} - \vec{P}_r||^2 + 1}$$

where A_m is the attenuation coefficient of morphogen m. For each cell, we compute the gradient of a morphogen m as the averaged variation of its intensity along the x, y, and z axis, from one extremity of the cell to the other.

Cell adhesion In the early stages of this experiment, every cell would automatically establish a strong connection with every other cell upon contact. This led to the invariable collapsing of the morphology diversity, especially in the water part of the world, where inertia is not negligible. Indeed, as cells divide, they experience various forces that propagate along the entirety of the organism. As a result, opposite ends of an organism often come in contact, bouncing against each other; but the automatic creation of a strong connection would prevent cells to go back apart and will eventually make for the construction of an unordered blob of connected cells. In various multicellular artificial life models, this problem is avoided because the actual simulation stage, in which the organism is evaluated, is separated from the development phase, where the cells are positioned and linked without perturbations. While this simplifies things and allows for the creation of complex morphologies without the risk of discovering a spherical amalgamation of cells at the end of the evaluation, it also means that we lose some of the properties of real world organisms which can be of prime interest, especially for this experiment which aims to get closer to real world organism development: mainly self-repair and real time morphology adaptation to a changing environment. To tackle this problem, we once again take inspiration from biology by introducing a mechanism which lets the cell decide if it wants to create new connections or only keep the ones already existing and bounce off of a potential companion. This capacity, named "solidify", is managed by the cells' gene regulatory network. In MecaCell, the normal algorithm for adhesion creations between two cells is to "ask" them what are their reciprocal affinities at each time step. In order to let the cells decide when they are open to new adhesions, we add an "active connections" list to each cell that keeps track of all their "real" adhesions. At each time step, and for every cell, we compare this active connections list with a candidate list of cells that are currently colliding. A new bond is then created only if both candidates decide not to solidify. In combination with the other proteins provided as inputs to the aGRN (such as the cell age t and its mechanical pressure p), this, in theory, allows for the emergence of complex adhesions strategies.

Cell controller - aGRN Within our multicellular organism, each cell has its own gene regulatory network that controls the cell lifecycle. Even though the aGRNs are physically different in the cells of the same organism, as in nature, they share the same genetic code and thus, the same topology. When a cell division occurs, an exact clone of the mother cell's aGRN is copied into the daughter cell. In this work, the gene regulatory network used to control the cells is inspired by Banzhaf's model. This model has been designed for computational efficiency and is not meant to simulate a real biological gene regulatory network in all its complexity.

This model is composed of a set of abstract proteins. A protein a is composed of three tags: (1) the *protein tag* id_a that identifies the protein, (2) the *enhancer tag* enh_a that defines the enhancing matching factor between two proteins, and (3) the *inhibitor tag* inh_a that defines the inhibiting matching factor between two proteins. These tags are coded with an integer in $[0, p]$ where the upper bound p can be tuned to control the precision of the network. In addition to these tags, a protein is also defined by its concentration that will vary over time with particular dynamics described later. A protein can be of three different types: *input*, a protein whose concentration is provided by the environment, which regulates other proteins but is not regulated; *output*, a protein with a concentration used as output of the network, which is regulated but does not regulate other proteins; and *regulatory*, an internal protein that regulates and is regulated by others proteins.

With this structure, the dynamics of the aGRN are computed by using the protein tags. They determine the productivity rate of pairwise interaction between two proteins. For this, the affinity of a protein a for another protein b is given by the enhancing factor u_{ab}^+ (resp. the inhibiting factor u_{ab}^-) calculated with the euclidean distance between protein b id tag and protein a enhancer (resp. inhibitor) tag. The proteins are then compared pairwise according to their enhancing and inhibiting factors. For a protein a, the total enhancement g_a and inhibition h_a are given by the sum of the exponential influences between the proteins. Two parameter β and δ are used to control the dynamics of the system: β affects the importance of the matching factors and δ is used to modify the production level of the proteins in the differential equation. In summary, the lower both values are, the smoother the regulation is; the higher the values are, the more sudden the regulation is. The concentrations are updated with a simple differential equation taking into account the newly produced proteins and the destroyed one. More details on the aGRN dynamics can be found in (Cussat-Blanc et al., 2015).

Table 1 describes the configuration of our aGRN input and output proteins when applied to this artificial embryogenesis experiment. A few clarifications on the role of some of these inputs and outputs is necessary. First, the sensed nutrients (c_n) input represents the actual concentration in nutri-

Name	Type	Description or use
$c_{i, \forall i \in [0,2]}$	input	concentration of morphogen i
c_n	input	sensed nutrients
n	input	current nutrients level.
c_l	input	sensed light intensity
l	input	current light level.
t	input	age of the cell
p	input	mechanical pressure
$o_{i, \forall i \in [0,2]}$	output	morphogen i production
o_N	output	normalisation of o_i
$d_{i, \forall i \in [0,2]}$	output	divide along morphogen i gradient
d_n	output	divide along nutrient gradient
a	output	apoptosis
q	output	quiesence
s	output	solidify: no new adhesion
s_T	output	threshold for s activation
pd	output	perpendicular division

Table 1: List of our artificial grn inputs and outputs proteins.

ents sensed by the cell in its surrounding environment. The current nutrients level (n) input is the actual current level of nutrients in the cell. The same goes for the light intensity sensed by the cell (c_l) and the current amount of light energy accumulated in it (l).

The cells express their choices between division, quiescence or apoptosis through the concentrations of the output proteins d_i, a and q respectively. The protein with the biggest concentration represents the cell's choice. In addition to starting a division, the d_i outputs proteins of the aGRN also controls the cells' division plane: each d_i output protein corresponds to a morphogen, and the d_i or d_n protein with maximum concentration is used to determine the gradient (morphogen or nutrient) along which the cell must divide. If no gradient of said morphogen or nutrient is present, the axis of division is randomly chosen. The pd protein allows the cell to choose between a division along the morphogen gradient or perpendicular to it when the concentration of protein pd is greater than the concentration of the selected division protein.

The solidify output protein s controls the solidify capacity of a cell: if the concentration of protein s rises above the threshold protein s_T, the cell solidifies and will not accept any more adhesion from not yet connected cells until the concentration of protein s decreases again.

To obtain a usable GRN, both the protein tags and the dynamics coefficients need to be optimized. The next part presents the specificities of the genetic algorithm used in this work.

Evolution

One of the goals of this experiment is to explore how artificial multicellular organisms could survive in a harsh envi-

ronment without explicitly being led toward a given strategy or morphology. We want to explore the organisms that the rules of this world could create without constraining the creativity of evolution through some restrictive objective function. Therefore, the only objective for the cells is to survive as long as they can, or more precisely, to keep at least one copy of their DNA in our virtual world for as long as possible. This gives full freedom to the cells on the developmental strategies they can use and opens a wide range of possibilities of morphologies the organisms can develop. The drawback is that it also dramatically increase the search space and fills it with many local optima that pave the way to increased longevity. The evolutionary algorithm we use in this work to evolve the aGRN is based on the Gene Regulatory Network Evolution through Augmenting Topology algorithm (GRNEAT) (Cussat-Blanc et al., 2015).

GRNEAT

In this algorithm inspired by the NEAT algorithm (Stanley and Miikkulainen, 2002) and adapted to evolve gene regulatory network, the first population of aGRNs is initialized with small topologies, containing only input and output proteins. The population is evaluated standardly with a fitness promoting survival time. After a 3-player tournament selection, offsprings are crossed over using a protein alignment operator. This operator uses a genetic distance metric to compute topological distances between two aGRN proteins. Each type of proteins is processed separately. Both the input and the output proteins are treated with the same method. One of each input (or output) protein linked to a sensor (or an actuator) is randomly selected from one of the parents. The regulatory proteins are then aligned before being crossed: for each regulatory protein p_i^1 from the first parent, the closest regulatory protein p_j^2 not yet aligned is selected from the second parent. The distance between two proteins is computed as follows:

$$D(A, B) = \frac{1}{p} \big(a|id_A - id_B| + b|enh_A - enh_B| + \\ c|inh_A - inh_B| \big)$$

where id_x is the tag, enh_x is the enhancer tag and inh_x is the inhibiter tag of protein x and p is the precision of the aGRN. $a = 0.75$, $b = 0.125$ and $c = 0.125$ are constants that weight each part of the protein properties. If the distance $D(p_i^1, p_j^2)$ is lower than a given alignment threshold σ_a, both proteins are aligned. Once alignment of all proteins has been attempted, one protein of each aligned pair is randomly selected and added to the offspring. The regulatory proteins that failed to align in both parents are also added to the offspring. This ensures that no crucial genetic material is deleted during the crossover. Finally, the dynamics coefficients are also crossed. One of the β and the δ coefficients is randomly selected from the parent genomes and used in the offspring genome.

Crossed-over aGRNs represent 30% of the offsprings. The rest of the offsprings are built using tournament selected genomes from the previous generation. All offsprings except the elite (the best genome) are then subject to mutation with a 75% rate. When mutated, a genome can be modified in three different ways: (1) delete a protein, with a 15% probability, randomly select a regulatory protein, if any, that is removed from the aGRN; (2) add a protein, with a 15% probability, adds a randomly generated regulatory protein; (3) modify a protein, with 70% probability, randomly modify exactly one parameter of the aGRN, either one protein tag or one of the dynamics coefficients.

Novelty metrics

In order to try to mitigate the adverse effects of increased degrees of freedom and numerous local optima in the morphological parameter space, we added a novelty metric as defined in (Lehman and Stanley, 2008). We combined this novelty score with our main survival objective by modifying the selection phase of our genetic algorithm: each potential parent is selected through a tournament based on either novelty or survival time, with a 50% chance. While not as complex as some other integrations of novelty in a multi-objective genetic algorithm (Mouret, 2011), this proved sufficient to harness some of the exploratory power of novelty. In this experiment, we tried three different novelty metrics, which are based on the capture (and comparison) of various aspects of a developing phenotype:

- Nm_0 is composed of three numbers: the maximum number of cells during the simulation, the maximum depth reached by a cell and the total survival time (which is also the main objective).

- Nm_1 is composed of 5 snapshots of the simulation (at times $t = 10$, $t = 20$, $t = 50$, $t = 75$ and $t = 100$). Each snapshot contains 2 numbers: the number of cells and the maximum depth of a cell at the time of the capture.

- Nm_2 is a set of 5 captures (taken at the same time steps as for Nm_1) represented as a 20×20 integer matrix. It is actually a set of pictures in which each pixel's value represents the number of cells stacked. The plane of the shot is determined through a Principal Component Analysis on the cells position (it is the most discriminant plane). This metric is meant to capture the morphologies of the organisms in all their subtleties

Results

Influence of novelty

In Figure 1, we can see the median (with first and third quartile) survival times of the best genomes evolved during 300 generations in 10 independent runs. On a 2014 high-end laptop, the average evaluation time for an individual was of 0.2s during the first 5 generations and ended at an average of
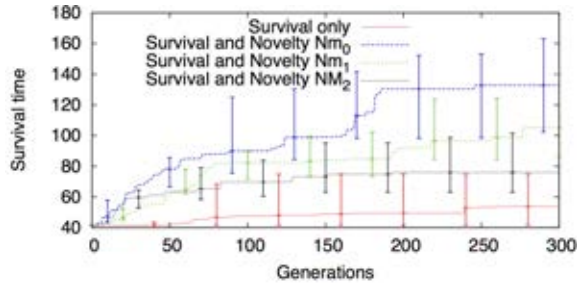
Figure 1: Error bar plots of the best individuals obtained on 10 independent runs. Error bars represents the median, the first and third quartiles. All novelty objectives are obviously helping to escape local optimum. However, the novelty measure Nm_0 is giving better results. The initial value of 41 obtained at generation 0 represents the survival time for a seed cell that stays quiescent during the whole simulation.

1.3s. The best organisms obtained with these runs are presented in Figure 2(a, c-h). This graph reveals both the deceptiveness of the fitness landscape when the survival time is used as only fitness objective as well the beneficial impact of novelty. This is undeniable (Student t-test p-values are provided in table 2): where a classical objective based evaluation struggles to find solutions that pass the first local optima (for example: not dividing and surviving on the initial resources of the seed cell, or just doing a few divisions in order for some cells to reach the surface and bring in a little bit of light), the novelty based approaches successfully find solutions to overcome these optima and efficiently pave the way to more robust organisms.

The three novelty measures tested in this experiment show that too much information loses the evolution in the vast search space: the novelty measure Nm_0 globally does better than both other measures. This measure is the one that includes the fewer parameters. In our opinion, when too much parameters are used to describe a phenotype, the exploration space becomes too large and individual with minor differences are considered too novel. Therefore, it is of high importance to wisely choose parameters that describe the phenotypes. As depicted in table 2, the relatively high p-values between novelty based runs reveal the necessity to make a broader study on the influence of the novelty parameters in order to find the best possible measures for evo-devo models and validate our preliminary results.

Developmental strategies and world setup influence

Along all the evolutionary runs, we observed an important diversity of developmental strategies and morphologies, especially when any form of novelty was involved. Figure 2(a,b) shows examples of cells arrangements obtained with different worlds parameters. The distribution of nutrients in the world was also found to be of huge influence over the preferred strategies: as expected, large values of C_n and P_n

Wins\Loses	Survival	Nm_0	Nm_1	Nm_2
Survival	-	0.002	0.011	0.089
Nm_0	0.002	-	0.360	0.050
Nm_1	0.011	0.360	-	0.250
Nm_2	0.089	0.050	0.250	-

Table 2: p-values of the paired Student t-test run comparison between runs with survival fitness and the different novelty measures calculated on 10 runs at generation 300.

favored a very vertical growth of the cell colony, with the formation of a relatively thick trunk in the ground enabling fast nutrient and light transfer between the deep roots cells and the emerged ones. One of the most interesting results might be the emergence of a form of reproduction through parthenogenesis when the nutrients concentration was uniform. Cells indeed understandably found the benefits of a vertical growth to be incomparable with the efficiency of a vertical growth. They also adopted, as shown in Figure 2(b), a spread method where they would laterally develop just below the surface. When a root cell encountered a nutrient source, it would also divide upward (to the surface) and the cells between the two formed cluster would undergo apoptosis, thus creating a simple form of parthenogenesis reminiscent of the biological reproduction of some plants.

Conclusion

We have presented a new developmental model based on MecaCell, a physics engine tailored for artificial life experiments. This model shows how novelty search can help when stepping artificial embryogenesis up to the third dimension. Indeed, this allows for more degrees of freedom for the multicellular organisms but also adds complexity for the cell controller to handle. As a result, this makes the search space much harder to explore with standard fitness function. In addition to the use of a 3D developmental model, we also wanted to remove all engineering from the main fitness objective: it only targets to the survival duration of the organisms. By only using this objective, we showed that the evolution is stuck in one or few local optima, but by adding different novelty metrics based on the organisms morphologies and capabilities to explore its environment, we showed that evolution can escape from these local optima and develop more complex morphologies and behaviors able to survive longer in the exact same environment.

This new developmental model opens many research perspectives. Firstly, we need to study more precisely the influence of the environment parameters on the multicellular organisms. During the development of the presented experiment, one of the major difficulty was to produce a viable environment, easy enough to allow the organisms to grow but difficult enough to require complex behaviors. Balancing this is difficult task and needs to be studied in detail.
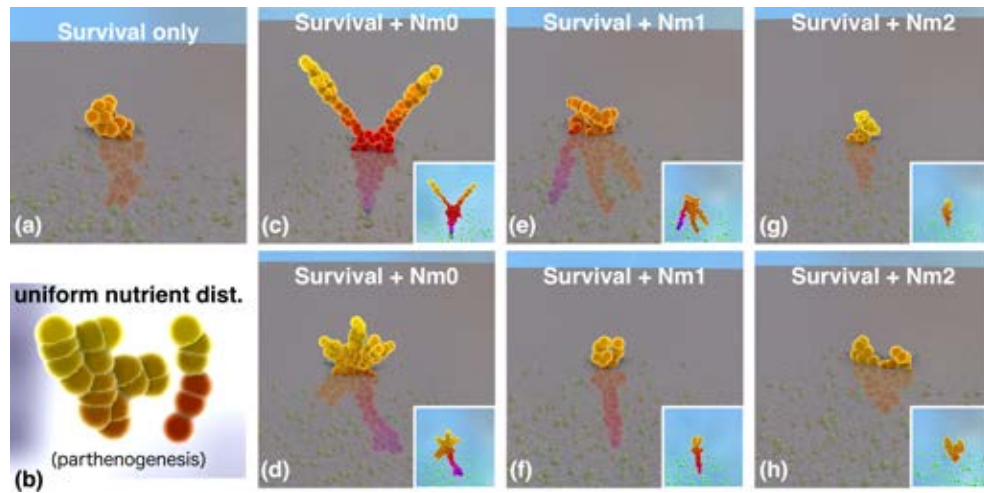
Figure 2: Examples of organisms obtained with the different fitnesses. (a, c-h) survival only and novelty metrics Nm_0, Nm_1 and Nm_2 in the novelty impact study. (a, b) same fitness with different environmental conditions.

Once done, we want to produce an artificial world in which different organisms would coexist, cooperating or competing for survival and reproduction. This will require specialization capacities of the cells in order to balance the capacities of the organisms, with for example a light extracting cell type and a reproductive one. We hope to produce more complex organisms, further mimicking some aspects of the early stages of the appearance of life on earth.

References

Cheney, N., Clune, J., and Lipson, H. (2014). Evolved electrophysiological soft robots. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 222–229.

Cussat-Blanc, S., Harrington, K., and Pollack, J. (2015). Gene regulatory network evolution through augmenting topologies. *Evolutionary Computation, IEEE Transactions on*, 19(6):823–837.

Disset, J., Cussat-Blanc, S., and Duthen, Y. (2014). Self-organization of symbiotic multicellular structures. In *the Fourteenth International Conference on the Synthesis and Simulation of Living Systems-ALIFE 2014*, pages pp–541.

Doursat, R. (2009). Organically grown architectures: Creating decentralized, autonomous systems by embryomorphic engineering. In *Organic computing*, pages 167–199. Springer.

Doursat, R. and Sánchez, C. (2014). Growing fine-grained multicellular robots. *Soft Robotics*, 1(2):110–121.

Fontana, A. and Wróbel, B. (2013). An artificial lizard regrows its tail (and more): regeneration of 3-dimensional structures with hundreds of thousands of artificial cells. In *Advances in Artificial Life, ECAL*, volume 12, pages 144–150.

Joachimczak, M., Kowaliw, T., Doursat, R., and Wrobel, B. (2013). Controlling development and chemotaxis of soft-bodied multicellular animats with the same gene regulatory network. In *Advances in Artificial Life, ECAL*, volume 12, pages 454–461.

Joachimczak, M. and Wróbel, B. (2011). Evolution of the morphology and patterning of artificial embryos: scaling the tricolour problem to the third dimension. *Advances in artificial life. Darwin meets von Neumann*, pages 35–43.

Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pages 329–336.

Lowengrub, J. S., Frieboes, H. B., Jin, F., Chuang, Y., Li, X., Macklin, P., Wise, S., and Cristini, V. (2009). Nonlinear modelling of cancer: bridging the gap between cells and tumours. *Nonlinearity*, 23(1):R1.

Mouret, J.-B. (2011). Novelty-based multiobjectivization. In *New horizons in evolutionary robotics*, pages 139–154. Springer.

Ouchi, N. B., Glazier, J. A., Rieu, J.-P., Upadhyaya, A., and Sawada, Y. (2003). Improving the realism of the cellular potts model in simulations of biological cells. *Physica A: Statistical Mechanics and its Applications*, 329(3):451–458.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127.

Robotic cell surface mechanics

Martin Hinsch[1], Athanasius F. M. Marée[1] and Verônica Grieneisen[1]

[1]CSB, JIC, Norwich
hinsch.martin@gmail.com

Abstract

In swarm robotics simple identical robots have to be made to coordinate in such a way that they can perform a task. Multi-cellular organisms similarly during development have to be able to create spatial patterns using many identical components (cells) without being able to draw on an absolute frame of reference. Finding and understanding existing solutions to the latter problem might therefore be a promising route to solving the former. Cell-surface mechanics, i.e. cell movement based on surface tension or adhesion is a mechanism that is known to be involved in many basic processes of morphogenesis. We implemented a simplified model of cell surface mechanics on the kilobot, a small robot with limited computational power and without any spatial orientation capabilities. Using only distance measurements to their neighbours kilobots were able to perform various morphogenetic tasks.

Introduction

In the last decades miniaturization and efficiency increases in processor and battery technology have led to the rise of swarm robotics (Brambilla et al., 2013). It is built on the promise that for many tasks a single complicated and expensive custom robot can be replaced by a potentially more robust swarm of simple and cheap off-the-shelf units (Barca and Sekercioglu, 2012).

Kilobots have been developed as a new platform with the explicit aim to make swarm robotics affordable for individuals or institutions with a limited budget (Rubenstein et al., 2012). Accordingly their design prioritizes simplicity and low hardware costs resulting in very limited capabilities.

While great advances have been made on the hardware side, programming robot swarms is still a challenge (Rubenstein et al., 2014). Swarm members have to differentiate from essentially identical initial states to a structured entity where different units or group of units perform different activities, solely based on local communication.

A similar challenge is faced by most multicellular organisms during ontogeny. Starting with identical units of limited complexity that only have access to local information and communication they have to create spatial and temporal patterns in order to develop differentiated tissues and organs.

Mixed masses of animal cells of different tissue types will over time rearrange leading to a number of different spatial configurations (Glazier and Graner, 1993). This sorting process is based on properties of the cells and their membranes. Although its biophysics are not completely understood, phenomenologically it can be modelled with a high degree of accuracy (Brodland, 2004; Marée et al., 2007).

In the past, various attempts have been made to translate biological mechanisms of pattern formation into robots (e.g. Zahadat et al., 2013). Here we will present an implementation of a cell surface mechanics-like system on the kilobot platform enabling basic morphogenetic processes.

Methods

Cell surface mechanics is based on the idea that interfaces between cells differ in their adhesion and/or surface tension depending on the types of the cells on both sides. In analogy to the formalisms applied e.g. in modelling foams, the free energy of a given membrane configuration can then be calculated. Local thermodynamic fluctuations in membrane state will produce stochastic changes in membrane configuration with a corresponding change in free energy. Assuming that changes to a lower energy state are more likely, the system as a whole will tend towards a decrease in free energy. Depending on the combination of cell types three main equilibrium patterns have been found: Cells will arrange in a *mixed* or checkerboard pattern, they can *separate* by type into two clusters or one cell type can form a layer *engulfing* the other one (Glazier and Graner, 1993).

Various different types of formalisms with different degrees of abstraction have been used to model cell surface mechanics (see review by Brodland, 2004). Some models track microscopic changes in membrane configuration and thus enable large scale changes in cell shape, others operate on the level of entire cells and provide less flexibility.

For our system we chose single cells as the basic unit of abstraction. We let each bot represent the centre of a cell with a circular "virtual cell" of a fixed radius extending around it. If the virtual cells of two bots overlap we assume they form an interfacial membrane on the line between the
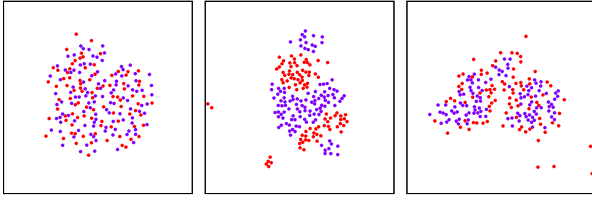
Figure 1: Robot positions after 50k seconds for *mixing* (left, $J_{1,1} = 8$, $J_{2,2} = 8$, $J_{1,2} = 2$), *separation* (middle, 2, 2, 8) and *engulfment* (right, 2, 12, 8). $J_M = 10$ in all cases.
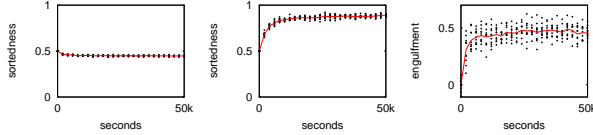
Figure 2: Sortedness or degree of engulfment, respectively, over time in 10 random replicates for *mixing* (left), *separation* (middle) and *engulfment* (right).

two intersection points of their circular cell surfaces. We use the standard formulation of the Hamiltonian to calculate energy for a given set of bot positions and resulting membrane configuration. Given segment lengths l_i, surface energy constant J and cell types at segment i, $\tau(i)$ and $\tau'(i)$, respectively, we obtain: $H = \sum_i J(\tau(i), \tau'(i)) \cdot l_i$

Kilobots can measure the distance to their neighbours, but are not able to determine their own or their neighbours' absolute position. We therefore approximate the degree to which the overlap between neighbouring interface segments reduces overall cell surface by calculating segment length as $l_i = l_{i,\text{orig}}/(1 + \sum l_{\text{orig}}/c)$.

Minimization of the energy state of the system happens by means of a Monte Carlo process: Each robot measures distances to its neighbours and from that calculates its current surface configuration. Then it performs a random movement step (within the limits of the kilobots' locomotion), after which it measures distances again. Based on this information the change in surface lengths and consequently the change in energy is calculated. If the result is favourable (i.e. $H' < H$) the robot maintains its position. Otherwise it moves back to its starting point.

In simulations of real cells different equilibrium states of a cell cluster will be reached depending on specific combinations of values of the surface energy constants $J(a, b)$ (Glazier and Graner, 1993). Using a freely available, accurate kilobot simulator (Jansson et al., 2015) we tested values of J corresponding to each of the end configurations *mixing*, *separation* and *engulfment*. We simulated 200 robots (100 per cell type) moving for 50,000 simulated seconds starting from random initial positions.

Results

For all three configurations (*mixing*, *separation* and *engulfment*) the spatial configuration of the robots clearly corresponds to the expected pattern (see fig. 1). Emergence of the patterns is robust against starting conditions (fig. 2)

Conclusion

We have successfully implemented a key morphogenetic process in a low capability swarm robot system. Despite the high stochasticity and the lack of information on absolute or relative position membrane-based changes in local potential energy lead to large-scale patterning in the robot swarm analogous to those observed in real tissues.

References

Barca, J. C. and Sekercioglu, Y. A. (2012). Swarm robotics reviewed. *Robotica*, 31(July 2012):1–15.

Brambilla, M., Ferrante, E., Birattari, M., and Dorigo, M. (2013). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41.

Brodland, G. W. (2004). Computational modeling of cell sorting, tissue engulfment, and related phenomena: A review. *Applied Mechanics Reviews*, 57(1):47.

Glazier, J. a. and Graner, F. (1993). Simulation of the differential adhesion driven rearrangement of biological cells. *Physical Review E*, 47(3):2128–2154.

Jansson, F., Hartley, M., Hinsch, M., Slavkov, I., Carranza, N., Olsson, T. S. G., Dries, R. M., Grönqvist, J. H., Marée, A. F. M., Sharpe, J., Kaandorp, J. A., and Grieneisen, V. A. (2015). Kilombo: a Kilobot simulator to enable effective research in swarm robotics. *CoRR*, abs/1511.0.

Marée, A. F. M., Grieneisen, V. a., and Hogeweg, P. (2007). The Cellular Potts Model and biophysical properties of cells, tissues and morphogenesis. In Anderson, A. R. A., Chaplain, M. A. J., and Rejniak, K. A., editors, *Single-cell-based models in Biology and Medicine*, pages 107–136. Basel.

Rubenstein, M., Ahler, C., and Nagpal, R. (2012). Kilobot: A low cost scalable robot system for collective behaviors. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3293–3298.

Rubenstein, M., Cornejo, a., and Nagpal, R. (2014). Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799.

Zahadat, P., Schmickl, T., and Crailsheim, K. (2013). Evolution of Spatial Pattern Formation by Autonomous Bio-Inspired Cellular Controllers. *Advances in Artificial Life, ECAL 2013*, pages 721–728.

The Relationship between Microscopic and Collective Properties in Gene Regulatory Network-based Morphogenetic Systems

Hyobin Kim[1,2] and Hiroki Sayama[1,2]

[1] Department of Systems Science and Industrial Engineering
[2] Center for Collective Dynamics of Complex Systems
Binghamton University, State University of New York, Binghamton, NY, USA
hkim240@binghamton.edu

Abstract

Gene regulatory network (GRN)-based morphogenetic systems have recently attracted an increasing attention in artificial life and morphogenetic engineering research. However, the relationship between microscopic properties of intracellular GRNs and collective properties of morphogenetic systems has not been fully explored yet. Thus, we propose a new GRN-based framework to elucidate how critical dynamics of GRNs in individual cells affect cell fates such as proliferation, apoptosis, and differentiation in resulting morphogenetic systems. Our model represents an aggregation of cells, where each cell has a GRN in it. We used Kauffman's NK Boolean networks for GRNs. Specifically, we randomly assigned three cell fates to the attractors. Varying the properties of GRNs from ordered, through critical, to chaotic regimes, we observed the process that cells are aggregated. We found that the criticality of a GRN made an optimal partition of basins of attraction, which led to a maximum balance between cell fates. Based on the result, we can conclude that the criticality of a GRN is an important controller to determine the frequencies of cell fates in morphogenetic systems.

Gene regulatory network (GRN)-based morphogenetic systems have been actively developed and their properties have been studied in artificial life and morphogenetic engineering (Doursat, 2008; Schramm, et al. 2012). However, the relationship between microscopic properties of intracellular GRNs and collective properties of morphogenetic systems has not been fully explored yet.

Here, we study the relationship between the critical dynamics of GRNs in cells and cellular functions performed in morphogenetic systems at a collective level. We used Kauffman's NK Boolean network as a model of GRNs (Kauffman, 1969, 1993, 1996). In NK Boolean networks, a dynamic attractor can be considered as a cellular function or a cell type. Thus, staying in different attractors can be interpreted as a dynamical representation of the cellular function. There exists much experimental evidence to support this view of cellular dynamics (Huang et al. 2005; Chang et al. 2008). Based on this view, Huang explained stochastic and reversible switching between cell fates using NK Boolean networks (Huang, 1999; Huang and Ingber, 2000).

Extending Huang's conceptual framework, we implemented NK Boolean network-based morphogenetic systems. In our model, we assumed that a cell has three fundamental cellular functions: proliferation, apoptosis, and differentiation. Our model represents an aggregation of cells, where each cell has an identical random NK Boolean network which consists of 20 nodes. By adjusting in-degree (K) of nodes of a GRN in the model, we can obtain various properties of GRNs from ordered, through critical, to chaotic regimes; K=1 is ordered, K=2 is critical, and K>2 is chaotic [3,4,5]. We generated random GRNs from K=1 to K=4. For each GRN, we randomly chose one attractor and assigned it the cellular function of proliferation. If there is any other attractor available, we chose another attractor randomly and assigned it the cellular function of apoptosis. If there is still any attractor available, we chose an attractor randomly and assigned it the cellular function of differentiation. This means that, if a GRN has only one attractor, it conducts only proliferation. If it has two attractors, it performs proliferation and apoptosis. With three or more attractors, all the cell fates assumed in the model can take place. Fig. 1(a) is a schematic diagram that shows three cell fates randomly assigned in a GRN which has more than three attractors.

Jumping from one cell fate to another may occur in every time step by perturbations in internal gene expression caused by cell-cell interactions within a morphogenetic system. Specifically, cells interact with one another through the transport of signal molecules between the environment and cells. The transport occurs through diffusion by the concentration difference of signal molecules. If the concentration of a signal molecule is beyond a certain threshold, it can control the expression of assigned genes.

Our morphogenetic model starts from one seed cell. The change of concentrations of signal molecules by diffusion leads to the change of gene expression. The altered gene expression finally converges to one attractor. If the converged attractor is proliferation, the cell is divided into two at the next time step. One is mother cell and the other is daughter one. Two cell share the half concentrations of signal molecules of the mother cell before division and they have the same GRN. With the process of proliferation, aggregated cells are composed of all the same GRNs. If the converged attractor is apoptosis, the cell dies. Once the cell becomes dead, it remains as it is for every time step. This is to examine how apoptosis has an influence on morphology based on the biological fact that apoptotic cell death contributes to cell morphology (e.g. the separation of fingers and toes in development). Last, if the converged attractor is differentiation, the cell is regarded as differentiated.
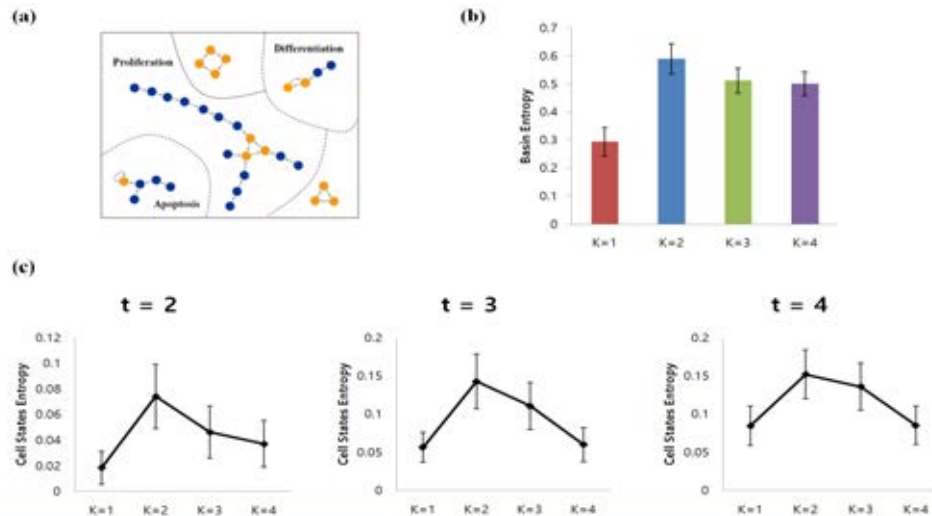
Figure 1: (a) Schematic diagram of randomly assigned three cell fates in a GRN which has more than three attractors. Each node represents a cell's dynamical state. Orange nodes are attractors. (b) Average basin entropy for K = 1–4. (c) Average states entropy of cell fates performed in simulations at each time step for K=1–4.

In our model, we assumed that cells staying in proliferation and differentiation states continue to switch between cell fates.

To investigate the structure of basins for cell fates according to the properties of GRNs, we applied revised basin entropy, using log base two. Basin entropy is a measure of the complexity of information that a system is capable of storing (Krawitz and Shmulevich, 2007). In the context of GRNs, the basin entropy represents the effective functional versatility of the cell. Originally, Krawitz's basin entropy is computed considering all the attractors and their basins. Meanwhile, focusing on the basins into which three cell fates are assigned, we calculated the values of basin entropy based on relative sizes of the basins. Fig. 1(b) shows the average of basin entropy for cell fates from K=1 to K=4. The average basin entropy is highest at K=2, i.e., the GRNs' basins of three cell fates are most evenly distributed at K=2.

For each group, we conducted 100 independent computational simulations of morphogenetic cell growth processes on a 2D spatial grid for t=0–4. By counting the numbers of cells expressing proliferation, apoptosis, or differentiation state at each time step in those simulations, we obtained the average values of cell states entropy based on relative frequencies (see Fig. 1(c)). Because there are too small number of cells at t=1 to capture distinct differences with K, we excluded a graph for the average values at t=1. As seen in Fig. 1(c), the cell states entropy is the highest at K=2 for t=2-4, which means when GRNs are critical, cells are aggregated remaining the most balanced three cell fates at each time step. The result confirmed that GRNs at K=2 have a maximum balance between cell fates. In the evolutionary view, the maximum balance has a significant implication; in any environmental changes, key cellular functions, such as proliferation, apoptosis, and differentiation, must be maintained in balance. Because those cellular functions are expressed by the attractors of a GRN in a cell, if the basins of cellular functions are distributed more evenly, the cell fates can better persist against environmental changes, which may work as a selective advantage in the process of evolution.

Our finding suggests that the criticality of a GRN may play an important role in modulating the frequencies of cell fates in morphogenetic systems. To obtain more theoretical/ empirical support for this suggestion, we plan to conduct large-scale evolutionary simulations of the ecologies of morphogenetic systems where the evolutionary success of multicellular organisms is determined by implicit fitness functions.

References

Chang, H. H., Hemberg, M., Barahona, M., Ingber, D. E., & Huang, S. (2008). Transcriptome-wide noise controls lineage choice in mammalian progenitor cells. *Nature*, 453(7194):544-547.

Doursat, R. (2008). Programmable Architectures That Are Complex and Self-Organized-From Morphogenesis to Engineering. In S. Bullock, J. Noble, R. Watson, and M. A. Bedau editors, A*rtificial Life XI*, pages 181-188. MIT Press, Cambridge, MA

Huang, S., Eichler, G., Bar-Yam, Y., & Ingber, D. E. (2005). Cell fates as high-dimensional attractor states of a complex gene regulatory network. *Physical review letters*, 94(12):128701.

Huang, S. (1999). Gene expression profiling, genetic networks, and cellular states: an integrating concept for tumorigenesis and drug discovery. *Journal of Molecular Medicine*, 77(6):469-480.

Huang, S, & Ingber, D. E. (2000). Shape-dependent control of cell growth, differentiation, and apoptosis: switching between attractors in cell regulatory networks. *Experimental cell research,* 261(1): 91-103.

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology*, 22(3):437-467.

Kauffman. S. A. (1993). *The origins of order: Self organization and selection in evolution*. Oxford University Press, USA.

Kauffman, S. A. (1996). *At home in the universe: The search for the laws of self-organization and complexity*. Oxford University Press.

Krawitz, P., & Shmulevich, I. (2007). Basin entropy in Boolean network ensembles. *Physical review letters*, 98(15):158701.

Schramm, L., Jin, Y., and Sendhoff, B. (2012). Evolution and analysis of genetic networks for stable cellular growth and regeneration. *Artificial life*, 18(4):425-444.

The Evolutionary Origins of Phenotypic Plasticity

Alexander Lalejini and Charles Ofria

Department of Computer Science and Engineering
Program in Ecology, Evolutionary Biology and Behavior
BEACON Center for the Study of Evolution in Action
Michigan State University, East Lansing, MI 48824
lalejini@msu.edu

Abstract

Many effective and innovative survival mechanisms used by natural organisms rely on the capacity for phenotypic plasticity; that is, the ability of a genotype to alter how it is expressed based on the current environmental conditions. Understanding the evolution of phenotypic plasticity is an important step towards understanding the origins of many types of biological complexity, as well as to meeting challenges in evolutionary computation where dynamic solutions are required. Here, we leverage the Avida Digital Evolution Platform to experimentally explore the selective pressures and evolutionary pathways that lead to phenotypic plasticity. We present evolved lineages wherein unconditional traits tend to evolve first; next, imprecise forms of phenotypic plasticity often appear before optimal forms finally evolve. We visualize the phenotypic states traversed by evolved lineages across environments with differing rates of mutations and environmental change. We see that under all conditions, populations can fail to evolve phenotypic plasticity, instead relying on mutation-based solutions.

Introduction

Phenotypic plasticity is the capacity for a genotype to express different phenotypes in response to different environmental conditions (Ghalambor et al., 2010) and is ubiquitous throughout nature. The capacity for phenotypic plasticity is central to many complex traits and developmental patterns found in nature and often serves as a key strategy employed by organisms to respond to spatially and temporally variable environments (Bradshaw, 1965; Murren et al., 2015). For example, *Daphnia pulex* use plasticity to differentially invest in morphological defenses during development, depending on the presence of predators in their local environment (Black and Dodson, 1990). Genetically homogeneous cells in a developing multicellular organism leverage their capacity for phenotypic plasticity to coordinate their expression patterns through environmental signals (Schlichting, 2003). Thus, understanding the evolution of plasticity is an important step toward a deeper understanding of biological complexity.

Phenotypic plasticity also has practical applications in the field of evolutionary computation where evolution by natural selection is harnessed to solve challenging computational and engineering problems. In many realistic problem domains, conditions are noisy or cyclically change. Plasticity could enable solutions to dynamically respond to changing problem conditions and be robust to noise. Both the biological and evolutionary computation domains motivate the following questions: (1) Under what conditions does phenotypic plasticity evolve? And (2), what are the evolutionary stepping stones for phenotypic plasticity?

Ghalambor *et al.* identify four conditions that are necessary for phenotypic plasticity to evolve: (1) populations are exposed to temporally or spatially varying environments, (2) the environments are differentiable by reliable signals, (3) different environments favor different phenotypes, and (4) no single phenotype can exhibit high fitness across all environments (Ghalambor et al., 2010). Theoretical and empirical findings support that phenotypic plasticity can evolve under these conditions in both natural and artificial systems (Clune et al., 2007; Goldsby et al., 2010, 2014; Hallsson and Björklund, 2012; Nolfi et al., 1994).

In addition to exploring the conditions that facilitate the evolutionary origin of phenotypic plasticity, it is also important to explore the step-by-step process in which plasticity actually evolves. What are the reoccurring themes as evolution progresses toward more plastic strategies? Are there genotypic or phenotypic patterns present in lineages leading to phenotypically plastic organisms? These types of questions are especially difficult to address in laboratory systems due to the slow pace of natural evolution, imperfections in lineage tracking, and the difficulty of acquiring high-resolution data on genotypes and phenotypes. As such, artificial life systems are the most effective way to observe and analyze the process by which phenotypic plasticity evolves.

Here, we use the Avida Digital Evolution Platform (Ofria et al., 2009) to explore the process by which phenotypic plasticity evolves in a fluctuating environment. We experimentally address two questions related to the evolution of phenotypic plasticity. First, do digital organisms evolve to express traits unconditionally before evolving to conditionally express them as a function of their environment, and do sub-optimal forms of plasticity evolve before more opti-

mal forms of plasticity? Second, how do mutation rate and environmental fluctuation rate affect the evolution of phenotypic plasticity? We also examine alternative evolutionary strategies to phenotypic plasticity in fluctuating environments and see evidence for bet-hedging strategies that use mutationally induced phenotype switching as a substitute for sensory-dependent plasticity.

Methods

The Avida Digital Evolution Platform

The Avida software provides a computational instance of evolution and enables researchers to experimentally test hypotheses about evolution that would otherwise be difficult or impossible to test in natural systems (Ofria et al., 2009). Avida has been demonstrated to have a robust genetic encoding; all possible genetic sequences are well-defined in any context (Ofria et al., 2009). Avida has also been shown to be capable of evolving to use a wide range of capabilities (Bryson and Ofria, 2013), making it an ideal choice for studying phenotypic plasticity. Here, we provide a brief overview of Avida as it is relevant to this work.

Digital Organisms Populations in Avida are made up of self-replicating computer programs that compete for space in a finite, toroidal grid. Each of these digital organisms is defined by a sequence of instructions (*i.e.* its genotype), virtual hardware to execute the instructions, and a position on the grid. The instruction set of Avida is Turing-Complete and enables organisms to perform basic computations, control their own execution flow, and replicate. An organism's virtual hardware (Figure 1) includes components such as a central processing unit (CPU), registers used for computation, input and output buffers, and memory stacks. Organisms replicate asexually by copying themselves line-by-line and dividing; however, an organism's copy instruction is imperfect, which can result in mutated offspring.

Organisms can gain additional CPU cycles by performing tasks – such as mathematical computations – to improve their metabolic rate. An organism's metabolic rate determines how rapidly it can execute its genome; a higher metabolic rate allows an organism to replicate faster. Initially, an organism's metabolic rate is roughly proportional to its genome length; however, the organism's metabolic rate can be adjusted when the organism completes a task. In this way, performance of tasks can be differentially rewarded or punished. When an organism successfully replicates, its offspring is placed in a random location in the world, replacing the organism formerly occupying that location. In this way, becoming a more efficient replicator in Avida is advantageous in the competition for space. The combination of competition for replication efficiency and heritable variation due to imperfect copying during the replication process results in evolution by natural selection.

Sensing in Avida In a typical Avida run, organisms must execute an instruction called IO to output the result of a computation. That output is analyzed to determine if any tasks have been performed, and if so, the organism is appropriately rewarded or punished. However, in this default scenario, organisms cannot sense the result, even after the task has been performed. To provide organisms with a mechanism to sense their environment, we added an IO-Sense instruction to the set of available instructions[1].

The IO-Sense instruction simulates IO and provides the organism with feedback on what would have happened if the organism had executed an IO instruction instead. This separation of IO performance and sensing allows organisms to determine whether or not a particular task is being punished without the risk of punishment, lowering the potential cost of sensing. If an IO operation would have resulted in a punishment, a -1 is added to the top of the organism's stack memory; if it would have resulted in a reward, a 1 is placed there. If an IO operation would have resulted in neither a reward nor a punishment, a 0 is placed on the organism's stack memory. In this way, organisms are able to sense whether or not a particular computational task is being rewarded or punished in their current environment and are able to react accordingly.

Identifying Phenotypic Plasticity in Avida We define a phenotypically plastic organism in Avida as an organism that leverages sensory information to alter the phenotype that they express based on the environment they are in. We restrict the definition of an organism's phenotype to the set of unique tasks it performs in the target environment. We don't consider *how many* times an organism performs a task in a given environment, but only whether the organism does the task at all. Thus, to be phenotypically plastic, an organism must express a different task profile – perform different tasks – in different environments.

Experimental Design

To explore the evolutionary history of phenotypically plastic organisms, we used an experimental design based on (Clune et al., 2007).

Environments We constructed two experimental environments named ENV-NAND and ENV-NOT. In ENV-NAND, organisms were rewarded for performing the NAND logical task but were punished for performing the NOT logical task. Conversely, in ENV-NOT, organisms were rewarded for performing the NOT logical task but were punished for

---

[1]IO-Sense is based on the IO-Feedback instruction implemented in (Clune et al., 2007), which worked exactly as the default IO instruction, but provided the organism with feedback on the result. Thus, an organism must first do a particular task once – and potentially get punished – to sense whether or not the task is beneficial.
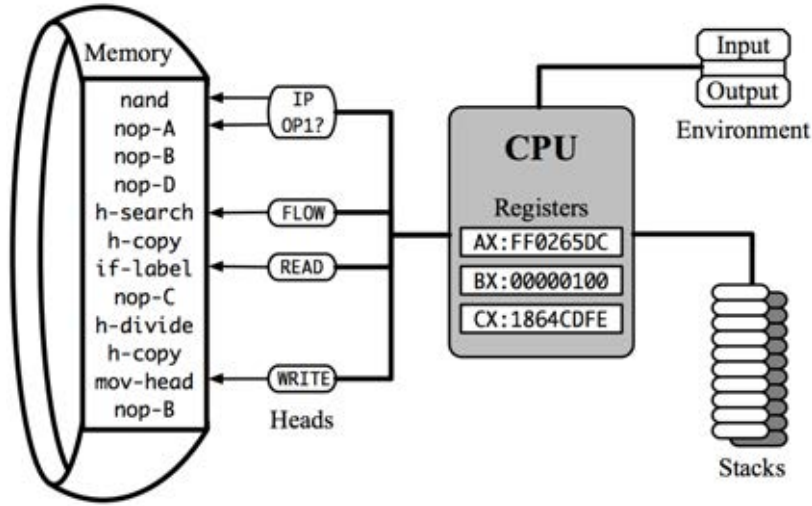
Figure 1: A visual representation of the default virtual hardware used by organisms in Avida. Original figure from: (Ofria et al., 2009)

performing the NAND logical task. In each of our experimental treatments, we cycled between these two environmental conditions. In this way, genotypes with the capacity to sense the current environment and express the appropriate task had a competitive advantage over phenotypically non-plastic organisms.

Phenotypes Given our simple definition of a phenotype, there are only four possible phenotypes in each of the two previously described environments: (1) perform only NAND, (2) perform only NOT, (3) perform both NAND and NOT, and (4) perform neither NAND nor NOT. When considering an organism's phenotype across both ENV-NAND and ENV-NOT, there are sixteen possible combinations. We enumerate these phenotypes in Figure 2. Of these sixteen possible phenotypes, only four express the identical task profile in both environments; the other 12 all exhibit some form of plasticity. The optimal form of plasticity is to perform only the NAND task in ENV-NAND and to perform only the NOT task in ENV-NOT; any other form of plasticity is sub-optimal. There are five possible phenotypes that leverage plasticity to perform punished tasks instead of rewarded tasks in a given environment; we did not expect these forms of phenotypic plasticity to be successful.

Treatments Our experimental design consisted of five treatments and a control: (1) a baseline treatment with a moderate point-mutation rate and environmental-cycle length, (2) a low-mutation-rate treatment, (3) a high-mutation-rate treatment, (4) a short-environment-cycle-length treatment, (5) a long-environment-cycle-length treatment, and (6) a control where both NAND and NOT were rewarded and the environment did not fluctuate. See Table 1 for treatment details.

We created the baseline treatment to produce phenotyp-

Treatment	Point-mutation Rate	Environment Cycle Length
Baseline	0.0075	100 updates
Low Mutation Rate	0.0025	100 updates
High Mutation Rate	0.0125	100 updates
Short Environment Cycle Length	0.0075	50 updates
Long Environment Cycle Length	0.0075	200 updates

Table 1: Differences among the five experimental treatments. Point-mutation rate is given as mutations per instruction copied. Environment cycle length describes the length of time (in updates) an environment is active before toggling to the alternative environment.

ically plastic organisms for lineage analysis. We limited the population size to 3600 organisms and seeded the world with an ancestral genotype capable only of self-replication. We then evolved populations for 100,000 updates[2] in Avida. We imposed a 0.0075 probability of point-mutation per instruction copied, as well as a 0.05 probability for each of single-instruction insertion and deletion per genome copied. We fluctuated the current environment between ENV-NAND and ENV-NOT every 100 updates in the baseline treatment. We ran 50 replicates of each treatment, including the control.

---

[2]An update in Avida is an experimental length of time. One update is defined as the amount of time it takes for the average organism to execute 30 instructions (see (Ofria et al., 2009) for more details).

#	Task Profile				Color Code	Type of Plasticity
	ENV-NAND		ENV-NOT			
	NAND	NOT	NAND	NOT		
1	–	–	–	–	(dark gray)	Non-plastic
2	X	–	X	–	(blue)	
3	–	X	–	X	(cyan)	
4	X	X	X	X	(white)	
5	–	–	–	X	(red)	Actively Beneficial
6	X	–	–	–	(dark red)	
7	X	X	–	X	(pink)	
8	X	–	X	X	(purple)	
9	X	–	–	X	(gold)	Optimal
10	X	X	–	–	(gray)	Neutral
11	–	–	X	X	(gray)	
12	–	X	–	–	(green)	Actively Harmful
13	–	–	X	–	(green)	
14	–	X	X	X	(green)	
15	X	X	X	–	(green)	
16	–	X	X	–	(green)	

Figure 2: Enumeration of all possible complete phenotypes. Each row represents a distinct phenotype. A green 'X' indicates that the associated task is performed in the specified environment, while a red '–' indicates that the task is not performed. For each environment, the column of the rewarded task is highlighted in green, and the column of the punished task is highlighted in red. A green 'X' in a green column or a red '–' in a red column is optimal. Each phenotype has a color code, which is used in our visualization tool. Note that the first four rows are non-plastic phenotypes, rows 5–8 exhibit partially beneficial plasticity, and row 9 is optimally beneficial. Rows 10–11 are mostly neutral, while rows 12–16 are detrimental forms of plasticity.

Lineage Visualization To explore evolutionary strategies evolved in fluctuating environments, we visualized the lineages of evolved genotypes as vertical bars where time (in updates) proceeds from top to bottom beginning with the lineage's original ancestor genotype. Any given genotype on the lineage must express one of the sixteen possible phenotypes enumerated in Figure 2. At each point in time, the color of the visualized lineage corresponds to the color representing the phenotype expressed by the lineage at that point in time. For example, because the ancestral organism is capable only of self-replication, all visualized lineages should show that the original ancestor's phenotype performed neither the NAND task nor the NOT task. In addition to the visualized lineages, we indicate the actual environmental conditions experienced by the evolving populations at each point in time by the color of the vertical axis. This type of visualization allows us to display the phenotypic states traversed by any given lineage, which allowed us to explore evolutionary strategies leveraged by all evolved lineages.

Results and Discussion

What conditions promote the evolution of phenotypic plasticity?

Ghalambor *et al.* identified four environmentally-dependent requirements for the evolution of phenotypic plasticity (Ghalambor et al., 2010), and our experimental design conforms to these conditions, enabling us to test their validity and relative importance. The oscillation between ENV-NAND and ENV-NOT provides temporal variation. The IO-Sense instruction reliably indicates the current environment. The two environments favor opposing phenotypic traits, and the only way for an individual organism to achieve a high fitness in both is to alter its phenotypic expression. Given the existing theoretical and empirical support for these conditions, we expected to see the evolution of phenotypic plasticity in each of our experimental treatments. However, we were unsure of the impact of altering environmental factors such as mutation rate and environment fluctuation rate.

At the end of the experiment, we extracted the dominant (most abundant) genotype from the population of each replicate. We tested these genotypes in both ENV-NAND and ENV-NOT and recorded each genotype's expressed phenotype across both environments. In Table 2, we report the number of replicates in which the dominant genotype at the end of the experiment was plastic and the number of replicates in which the dominant genotype was optimally plastic. Note that for these results we only evaluated the most abundant genotype at the end of the experiment. An ancestor of the evaluated genotype may have been plastic, but if that plasticity was not maintained in the lineage, we did not count it in Table 2.

As expected, the capacity for phenotypic plasticity evolved in each experimental treatment; in 31 of the 50 baseline treatment replicates, phenotypic plasticity was present in the final dominant organism. None of the final dominant genotypes from the control replicates were phenotypically plastic. In all control replicates, the dominant genotype performed both the NAND and NOT tasks unconditionally. Our results are consistent with existing theoretical and empirical work supporting the validity of the conditions likely to facilitate the evolution of phenotypic plasticity (Clune et al., 2007; Ghalambor et al., 2010; Hallsson and Björklund, 2012; Nolfi et al., 1994).

How do environmental factors impact the evolution of phenotypic plasticity?

While our results show phenotypic plasticity can evolve under the conditions identified in (Ghalambor et al., 2010), how do mutation rate and fluctuation rate affect the evolution of phenotypic plasticity under these conditions? We found compelling results for both mutation rate and environmental cycle length.

Treatment	Plastic Replicates		Unconditional Precedes Conditional		Sub-optimal Precedes Optimal
	Total	Optimal[*]	NAND Task	NOT Task	
Baseline	31 (62%)	17 (34%)	31 (100%)	28 (90.3%)	16 (94.1%)
Low Mutation Rate	38 (76%)	30 (60%)	34 (89.5%)	35 (92.1%)	30 (100%)
High Mutation Rate	25 (50%)	11 (22%)	25 (100%)	24 (96%)	10 (90.9%)
Short Environment Cycle Length	36 (72%)	18 (36%)	33 (91.7%)	28 (77.8%)	18 (100%)
Long Environment Cycle Length	16 (32%)	10 (20%)	14 (87.5%)	16 (100%)	9 (90%)
Control	0 (0%)	0 (0%)	–	–	–

[*]Optimal is defined as the complete phenotype that only performs the rewarded task in each environment.

Table 2: A summary of evolutionary outcomes across all five experimental treatments and control. Plastic Replicates indicates the number of replicates (out of 50 per treatments) in which the final dominant genotype was plastic at all (Total) and perfectly plastic (Optimal). Unconditional Precedes Conditional indicates the number of times the NAND task and NOT task were expressed unconditionally before eventually evolving to be express conditionally (out of total plastic). Finally, Sub-optimal Precedes Optimal indicates how many runs had an imperfect form of plasticity before eventually evolving to be optimally plastic (out of total optimally plastic).

Mutation Rate While only of borderline statistical significance ($p = 0.058$ using Fisher's Exact Test with Bonferroni corrections for multiple comparisons; all statistics were done in R version 3.2.2 (R Core Team, 2015)), our results trend such that populations at lower mutation rates appear more likely to evolve phenotypic plasticity than do populations at higher mutation rates. The most abundant genotypes exhibited some plasticity in 38/50 runs at a low mutation rate, 31/50 at the baseline mutation rate, and 25/50 and the high mutation rate. While higher mutation rates increase genetic variation from one generation to the next, most mutations that have phenotypic effects are deleterious (Sniegowski et al., 2000). Thus, at higher mutation rates, the elevated influx of deleterious mutations could increase the difficulty of maintaining the necessary genetic machinery for phenotypic plasticity. Qualitative evidence for this effect can be seen in the time-sliced visualized lineages of final dominant, non-plastic genotypes from the high-mutation-rate treatment (Figure 3) where lineages traverse states of plasticity for some time before reverting back to states of non-plasticity [3]. Furthermore, more phenotypic shifts in general increase the probability of quickly finding an appropriate non-plastic phenotype after each environmental change.

Environment Fluctuation Rate We found a highly significant difference ($p = 0.00028$ using Fisher's Exact Test with Bonferroni corrections for multiple comparisons) as we varied the cycle length for environmental switching. Specifically, in the long-environment-cycle-length, only 16/50 runs ended with a final dominant genotype that was phenotypi-

cally plastic, while the basline and short-environment-cycle-length produced 31 and 36 plastic outcomes, respectively.

We expect that the short-environment-cycle-length treatment is biased toward the evolution of phenotypic plasticity because of the rapid environment fluctuations relative to other experimental treatments. Rapid fluctuations cause lineages to be less able to rely on mutational input for adaptation. In the long-environment-cycle-length treatment, environmental fluctuations may not be occurring rapidly enough to produce a sufficient selective pressure for phenotypic plasticity, allowing alternative adaptive strategies to evolve instead.

What are the evolutionary stepping stones for phenotypic plasticity?

In an attempt to identify patterns frequently encountered during the evolution of phenotypically plastic organisms, we extracted and analyzed the full lineages from our experiments. We tested each ancestor genotype in both ENV-NAND and ENV-NOT and classified their phenotype across both environments. In addition to a quantitative analysis, we also visualized the lineages of the dominant, plastic genotypes; see Figure 4 for the visualization of the baseline treatment. Using our visualizations and ancestor phenotype classifications, we addressed the following two questions: (1) Do the lineages of phenotypically plastic organisms first evolve to perform tasks unconditionally before evolving to perform them conditionally as a function of their current environment? And (2), do imperfect forms of phenotypic plasticity tend to precede optimal forms?

Unconditional Task Performance To explore whether or not unconditional task performance was an evolutionary stepping stone for conditional task performance (*i.e.* phenotypic plasticity), we determined whether a task was per-

---

[3]For fully interactive visualizations of evolved lineages from all treatments, see `http://cse.msu.edu/~lalejini/evo-origins-of-phenotypic-plasticity-web/lineage_visualization.html`
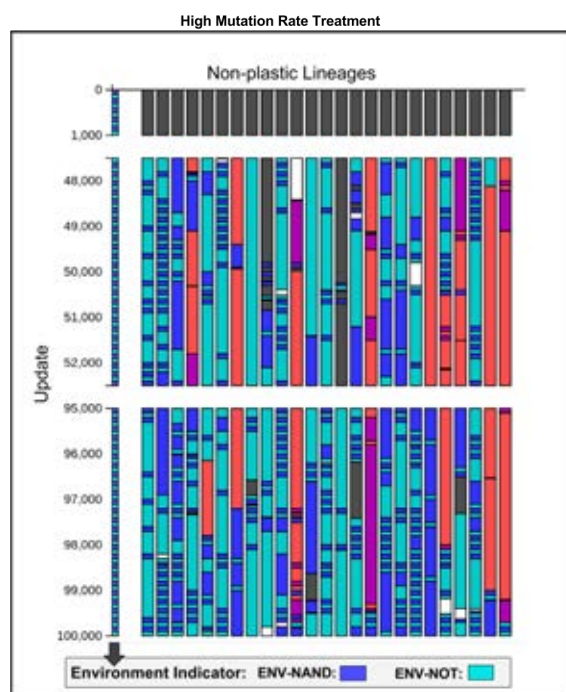
Figure 3: Time-sliced visualization of lineages for non-plastic, dominant genotypes from the high-mutation-rate treatment. Quick color reference: cyan represents unconditional NOT task performance, dark blue represents unconditional NAND task performance, and red/purple are sub-optimal forms of plasticity. Refer to Figure 2 for a full legend of phenotype colors.

Figure 4: Time-sliced lineage visualization of dominant, plastic genotypes from the baseline treatment. Quick color reference: cyan represents unconditional NOT task performance, dark blue represents unconditional NAND task performance, different shades of red/purple are sub-optimal forms of plasticity, and yellow represents optimal plasticity. Refer to Figure 2 for a full legend of phenotype colors.

formed unconditionally prior to being performed conditionally by the ancestors of plastic genotypes. We analyzed both tasks – NAND and NOT – separately. These results are reported in Table 2.

Across all experimental treatments, non-plastic ancestors generally preceded plastic ancestors. In other words, unconditional task performance of the NAND and NOT tasks generally preceded the conditional performance of either task. Examples of this can be seen in time-sliced plastic lineages from the baseline treatment (Figure 4) where many lineages maintain states of unconditional task expression prior to entering states of conditional task expression. These results suggest that, in fluctuating environments similar to those in our experiment, the evolutionary path to phenotypic plasticity usually traverses states of unconditional trait expression prior to entering states of conditional trait expression. This result should be unsurprising. In order to evolve a regulated function, the capacity for both the regulation and the function must exist. In our experiment, the function can be selected for without regulation; however, regulation of the function is unlikely to be selected for without the prior capacity for the function.

Sub-optimal Phenotypic Plasticity To investigate sub-optimal phenotypic plasticity as an evolutionary stepping

stone for optimal phenotypic plasticity in our experiment, we analyzed lineages of optimally plastic genotypes. Again, we consider only complete phenotypes that exclusively perform the rewarded task in each environment to be optimal. For each optimally plastic genotype's lineage, we determined whether or not the evolution of optimal plasticity was preceded by the evolution of sub-optimal phenotypic plasticity. The results of this analysis are reported in Table 2.

Across all experimental treatments, the evolution of sub-optimal plasticity did, indeed, generally precede the evolution of optimal phenotypic plasticity. Examples of sub-optimal plasticity preceding more optimal forms of plasticity can be seen in some of the time-sliced lineages from the baseline treatment visualized in Figure 4. These results suggest that, in fluctuating environments similar to those in our experiment, sub-optimal forms of phenotypic plasticity tend to arise before the evolution of optimal forms of phenotypic plasticity.

Unconditional trait expression tends to evolve first; then, sub-optimal forms of plasticity appear before optimal forms finally evolve. While challenging to verify, we expect our results to be applicable to biological systems. The evolution of complex functions (*e.g.* optimal phenotypic plasticity) build on simpler, previously evolved functions (*e.g.* unreg-

ulated or sub-optimally regulated functions) (Lenski et al., 2003). These results, however, are particularly useful for applied evolutionary computation. If an evolved problem solution must respond dynamically to environmental variables, it is likely that the solution will need to be able to traverse through states of rigidity and sub-optimal plasticity prior to reaching a state of optimal plasticity. Thus, first evolving rigid solutions in fixed environments and then gradually starting to fluctuate more aspects of the environment over time could provide a scaffolding for the evolution of optimally plastic solutions.

Are stochastic strategies evolving as an alternative to phenotypic plasticity?

Stochastic phenotype switching – a form of bet hedging (Seger, 1987) – is a common strategy leveraged by bacteria in fluctuating environments (Rainey et al., 2011). Unlike phenotypic plasticity where environmental conditions alter gene expression, stochastic phenotype switching relies on mutational input to induce phenotypic changes. This strategy is thought to be a viable alternative to phenotypic plasticity in the absence of reliable environmental signals or when the processing of sensory information is costly (Rainey et al., 2011). Strategic stochastic phenotype switching often relies on contingency loci – hypermutable regions of the genome that can induce phenotype switching via mutational input (Moxon et al., 2006).

We hypothesized that stochastic phenotype switching was an alternative evolutionary strategy to phenotypic plasticity because of its commonality in bacteria. We most expected to see stochastic phenotype switching in our experimental treatments where the fewest number of replicates produced phenotypically plastic final, dominant genotypes.

Lineage Visualization It can be difficult to intuitively understand evolutionary strategies leveraged by a lineage without a visual aid. To explore evolutionary strategies alternative to phenotypic plasticity in fluctuating environments, we visualized the lineages of dominant, non-plastic genotypes from our experimental treatments.

If a lineage relied on stochastic phenotype switching, we would expect it to switch between phenotypic states of unconditional NAND task performance and unconditional NOT task performance in approximate synchronization with the changing environment. Specifically, we should see ancestors along a lineage perform NAND unconditionally during periods of ENV-NAND and see ancestors performing NOT unconditionally during periods of ENV-NOT. We show a time-sliced lineage visualization of dominant, non-plastic genotypes at the end of our experiment for the long-environment-cycle-length treatment (Figure 5).

From Figure 5, we see what appear to be cases of stochastic phenotype switching – lineages switching between phenotypic states of unconditional NAND task performance and
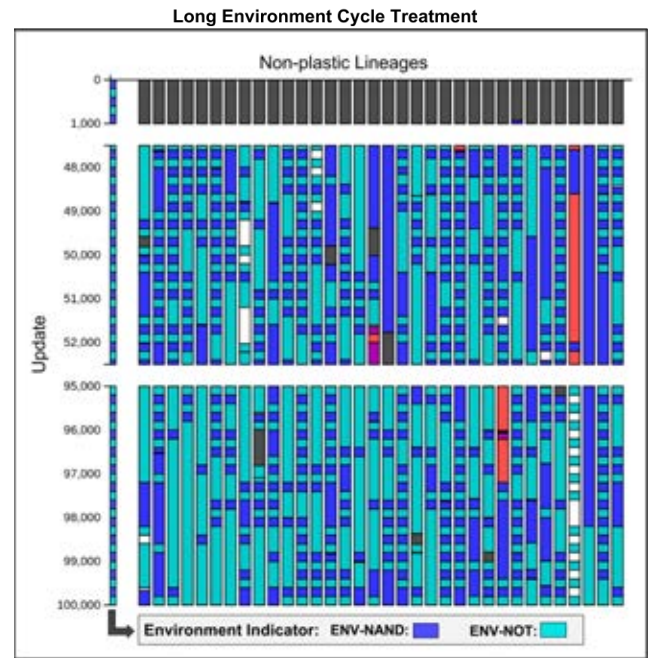
Figure 5: Time-sliced lineage visualization of non-plastic, dominant genotypes from the long environment cycle treatment. Quick color reference: cyan represents unconditional NOT task performance, dark blue represents unconditional NAND task performance, and red/purple are sub-optimal forms of plasticity. Refer to Figure 2 for a full legend of phenotype colors.

unconditional NOT task performance in approximate synchronization with the environment. Many of the lineages in the long-environment-cycle treatment seem to be undergoing stochastic phenotype switching. A few examples of what appear to be stochastic phenotype switching can even be seen in Figure 4 (the plastic lineages from our baseline treatment) between updates 47,500 and 52,500 (the middle time-slice), prompting the following open question: in addition to being an alternative strategy to plasticity in fluctuating environments, could stochastic phenotype switching also act as a precursor or building block toward plasticity?

Our visualizations only provide an exploratory method for understanding evolutionary strategies employed by a lineage. Further analysis would be required to confirm or reject our hypothesis that stochastic phenotype switching is evolving as an alternative strategy to phenotypic plasticity in our system. This hypothesis is particularly worthwhile to explore because our mutation rate was fixed across the genome, preventing the evolution of contingency loci. Furthermore, because sensing mechanisms were perfectly accurate, phenotypic plasticity was a reliable strategy. We hypothesize that genotypes are moving to a region of the mutational landscape that straddles the boundary between expressing unconditional NAND task performance and unconditional NOT task performance such that minimal mutational input is required to switch phenotypes. This type

of evolutionary trajectory has been demonstrated by Crombach and Hogeweg in evolutionary simulations of simple, genome-encoded gene regulatory network models (Crombach and Hogeweg, 2008). In their simulations, Crombach and Hogeweg found that networks evolved in an oscillating environment possessed genotype to phenotype mappings that were mutationally more efficient at generating adaptive phenotypes in alternative environments.

Conclusion

In this work, we evolved populations of phenotypically plastic organisms at varied rates of environmental fluctuation and mutation using the Avida Digital Evolution Platform. We analyzed the lineages of evolved genotypes for clues about the evolutionary stepping stones toward phenotypic plasticity. We found that the capacity for phenotypic plasticity evolved under conditions identified by previous research (Clune et al., 2007; Ghalambor et al., 2010). We found evidence that traits are generally expressed unconditionally prior to the evolution of conditional trait expression and that sub-optimal forms of phenotypic plasticity generally evolve before optimal forms of phenotypic plasticity. Both of these results are examples of evolution's use of simpler functions as building blocks for more complex functions as in Lenski *et al.* (Lenski et al., 2003).

Visual inspection of the evolutionary histories leading to phenotypically plastic organisms suggests that under certain conditions stochastic phenotype switching evolves as an alternative strategy to phenotypic plasticity, just as it does in many bacteria (Moxon et al., 2006; Rainey et al., 2011). Of course, in these bacterial cases, hypermutable sites tend to appear in the genomes (called "contingency loci") that facilitate such task switching.

Given these promising results, we plan to explore whether stochastic phenotype switching can be a viable evolutionary strategy in the absence of the ability to evolve hypermutable regions of the genome. Given the potential difficulty in maintaining the necessary genetic machinery associated with phenotypic plasticity, are there cases in which stochastic phenotype switching is more robust than phenotypic plasticity? And, does this contribute to the evolution of stochastic phenotype switching as an evolutionary strategy? Metrics are clearly needed for quantifying stochastic phenotype switching in digital systems and for evaluating the mutational landscapes of genotypes along a lineage.

Acknowledgments

References

Black, A. R. and Dodson, S. I. (1990). Demographic costs of chaoborus-induced phenotypic plasticity in daphnia pulex. *Oecologia*, 83(1):117–122.

Bradshaw, A. D. (1965). Evolutionary significance of phenotypic plasticity in plants. *Advances in Genetics*, 13(1):115–155.

Bryson, D. M. and Ofria, C. (2013). Understanding evolutionary potential in virtual cpu instruction set architectures. *PLoS ONE*, 8:e83242.

Clune, J., Ofria, C., and Pennock, R. T. (2007). Investigating the emergence of phenotypic plasticity in evolving digital organisms. In *Advances in Artificial Life*, pages 74–83. Springer.

Crombach, A. and Hogeweg, P. (2008). Evolution of evolvability in gene regulatory networks. *PLoS Comput Biol*, 4(7):e1000112.

Ghalambor, C. K., Angeloni, L. M., and Carroll, S. P. (2010). Behavior as phenotypic plasticity. *Evolutionary Behavioral Ecology*, pages 90–107.

Goldsby, H. J., Knoester, D. B., and Ofria, C. (2010). Evolution of division of labor in genetically homogenous groups. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pages 135–142. ACM.

Goldsby, H. J., Knoester, D. B., Ofria, C., and Kerr, B. (2014). The evolutionary origin of somatic cells under the dirty work hypothesis. *PLoS Biology*, 12:e1001858.

Hallsson, L. R. and Björklund, M. (2012). Selection in a fluctuating environment leads to decreased genetic variation and facilitates the evolution of phenotypic plasticity. *Journal of Evolutionary Biology*, 25(7):1275–1290.

Lenski, R. E., Ofria, C., Pennock, R. T., and Adami, C. (2003). The evolutionary origin of complex features. *Nature*, 423(6936):139–144.

Moxon, R., Bayliss, C., and Hood, D. (2006). Bacterial contingency loci: the role of simple sequence dna repeats in bacterial adaptation. *Annu. Rev. Genet.*, 40:307–333.

Murren, C. J., Auld, J. R., Callahan, H., Ghalambor, C. K., Handelsman, C. A., Heskel, M. A., Kingsolver, J. G., Maclean, H. J., Masel, J., Maughan, H., et al. (2015). Constraints on the evolution of phenotypic plasticity: limits and costs of phenotype and plasticity. *Heredity*, 115(4):293–301.

Nolfi, S., Miglino, O., and Parisi, D. (1994). Phenotypic plasticity in evolving neural networks. In *From Perception to Action Conference, 1994., Proceedings*, pages 146–157. IEEE.

Ofria, C., Bryson, D. M., and Wilke, C. O. (2009). Avida: A software platform for research in computational evolutionary biology. In *Artificial Life Models in Software*, pages 3–35. Springer.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rainey, P. B., Beaumont, H. J., Ferguson, G. C., Gallie, J., Kost, C., Libby, E., and Zhang, X.-X. (2011). The evolutionary emergence of stochastic phenotype switching in bacteria. *Microbial Cell Factories*, 10(1):1.

Schlichting, C. D. (2003). Origins of differentiation via phenotypic plasticity. *Evolution & Development*, 5(1):98–105.

Seger, J. (1987). What is bet-hedging? *Oxford Surveys in Evolutionary Biology*, 4:182–211.

Sniegowski, P. D., Gerrish, P. J., Johnson, T., Shaver, A., et al. (2000). The evolution of mutation rates: separating causes from consequences. *Bioessays*, 22(12):1057–1066.

Learning and Memory

Fully Autonomous Real-Time Autoencoder-Augmented Hebbian Learning through the Collection of Novel Experiences

Joshua A. Bowren[1], Justin K. Pugh[1], and Kenneth O. Stanley[1]

[1]Department of Computer Science, University of Central Florida, Orlando, FL 32816 USA
jbowren@cs.ucf.edu, jpugh@cs.ucf.edu, kstanley@cs.ucf.edu

Abstract

Hebbian plasticity in artificial neural networks is compelling for both its simplicity and biological plausibility. Changing the weight of a connection based only on the activations of the neurons it connects is straightforward and effective in combination with neuromodulation for reinforcing good behaviors. However, a major obstacle to any ambitious application of Hebbian plasticity is that the performance of a layer of Hebbian neurons is highly sensitive to the choice of inputs. If the inputs do not represent precisely the features of the environment that Hebbian connections must learn to correlate to actions, the network will struggle to learn at all. A recently-proposed solution to this problem is the *Real-time Autoencoder-Augmented Hebbian Network* (RAAHN), which inserts an autoencoder between the inputs and the Hebbian layer. This autoencoder then learns in real time to encode the raw inputs into higher-level features while the Hebbian connections in turn learn to correlate these higher-level features to correct actions. Until now, RAAHN has only been demonstrated to work when it is driven by an autopilot during training (in a robot navigation task), which means its experiences are carefully controlled. Progressing significantly beyond this early demonstration, the present investigation now shows how RAAHN can learn to navigate from scratch entirely on its own, without an autopilot. By removing the need for an autopilot, RAAHN becomes a powerful new Hebbian-centered approach to learning from sparse reinforcement with broad potential applications.

Introduction

As a key mechanism behind adaptation in natural organisms, neural plasticity has attracted significant interest in artificial life (alife) (Floreano and Urzelai, 2000; Niv et al., 2002; Soltoggio et al., 2008, 2007; Soltoggio and Jones, 2009; Soltoggio and Stanley, 2012; Risi et al., 2011; Risi and Stanley, 2012; Stanley et al., 2003; Coleman and Blair, 2012). A popular option for studying neural plasticity in artificial neural networks (ANNs) is Hebbian learning, which follows the simple mechanism of increasing connection weights proportionally to the activation strengths of the neurons they connect (Hebb, 1949). For example, researchers often incorporate Hebbian learning into evolutionary algorithms that evolve ANNs to control agents in dynamic or uncertain environments (Floreano and Urzelai, 2000; Soltoggio et al., 2008;

Risi et al., 2011). Sometimes such Hebbian networks are accompanied by *neuromodulation* (Soltoggio et al., 2008, 2007; Soltoggio and Jones, 2009; Soltoggio and Stanley, 2012; Risi and Stanley, 2012; Coleman and Blair, 2012), which allows a reward or penalty signal to turn on or off the plasticity of Hebbian connections appropriately. However, a major obstacle to the success of Hebbian ANNs is that the inputs to Hebbian layers must be carefully selected to encompass the right incoming environmental features or the proper correlations will otherwise become too difficult to learn. In domains in which the right features may not be known a priori, or where it may even be necessary to learn them from raw inputs through experience, Hebbian learning thereby becomes brittle or even prohibitive.

Responding to this challenge, Pugh et al. (2014) proposed recently that it might be possible to bridge the gap between the raw inputs to an ANN and a Hebbian layer through an *autoencoder*, which is itself an ANN with at least one hidden layer that is trained to reconstruct its inputs (Bengio, 2009). The value of the hidden layer in the autoencoder is that it typically comes to represent higher-level features of the environment (because they then aid in the reconstruction of the inputs). These higher-level features distilled from raw inputs could be just the features needed by a Hebbian layer to learn correlations between environmental features and appropriate agent actions. The idea behind Pugh et al. (2014) is that in principle both an autoencoder layer and a neuromodulated Hebbian layer can be learned simultaneously, in real time, which would allow an agent to construct a higher-level representation of its environment at the same time as it learns to navigate based on that developing representation. Pugh et al. (2014) call this hybrid combination of autoencoder and Hebbian layers a *Real-time Autoencoder-Augmented Hebbian Network* (RAAHN).

To validate RAAHN, Pugh et al. (2014) showed that it can learn key features of a two-dimensional maze domain at the same time as learning to navigate the domain in real time. Furthermore, a pure Hebbian learner could not effectively learn the same policy, thereby confirming the advantage of Hebbian learning from the autoencoder layer. However, a

major limitation of this demonstration is that the agent was guided during learning by an *autopilot* that ensured that the agent experienced a prescripted succession of inputs identified by the experimenters as appropriate to the task. In this way, the autopilot phase resembles supervised learning more than the kind of autonomous exploratory learning one might hope to see in alife. Ideally, the agent would explore on its own, accumulating higher-level features as it goes, and improving its ability to navigate based on those features at the same time.

The aim of this paper is to take that next step, demonstrating that RAAHN is indeed sufficiently capable of doing all the learning on its own, without an autopilot to guide it. Such a result would open up a broad range of possible experiments and applications, where agents can be released into a world to explore and learn without explicit guidance, more in the spirit of reinforcement learning (RL) (Watkins and Dayan, 1992; Rummery and Niranjan, 1994) than supervised learning. To enable this capability, RAAHN is slightly elaborated through a new kind of novelty-based history buffer (which decides from which experiences it is trained) and neural noise to encourage autonomous exploration. The result, demonstrated in a two-dimensional maze domain, is ultimately that RAAHN can learn effectively on its own, and furthermore that RAAHN can learn even when the provided sensors are insufficient for neuromodulated Hebbian learning.

With RAAHN's ability to learn control policies and higher-level features in real time established, RAAHN can begin to be employed in more sophisticated unguided scenarios. RAAHN also goes beyond traditional RL algorithms (Watkins and Dayan, 1992; Rummery and Niranjan, 1994) because it has the potential to learn increasingly high-level features through stacking autoencoders (Le et al., 2012) in the future. Such autoencoder stacks and entire RAAHN architectures potentially can even be evolved in the future through neuroevolution (Stanley and Miikkulainen, 2002; Floreano et al., 2008; Yao, 1999). As a first step towards such ends, this study accordingly establishes best practices for successfully running RAAHN without the need for an autopilot.

Background

Before previewing the original work on RAAHN, this section begins with a review of Hebbian learning and autoencoders, which are the two core components of RAAHN.

Hebbian Learning

Basic Hebbian learning is implemented in ANNs with the simple learning rule

$$\Delta w_i = \eta x_i y, \tag{1}$$

where w_i is the weight connecting two neurons with activations x_i and y, and η is the learning rate. This learning rule

has the advantage of being completely local, making its application flexible. It is also biologically motivated, reflecting basic principles of neural plasticity.

Researchers interested in evolving ANNs in particular took interest in the Hebbian rule as a means to allowing evolved ANNs to exhibit plasticity during their lifetime (Floreano and Urzelai, 2000; Niv et al., 2002; Risi and Stanley, 2010; Risi et al., 2011; Stanley et al., 2003). Furthermore, by adding *neuromodulation* to the basic Hebbian learning rule, Hebbian ANNs are able to be trained with rewards and penalties similar to reinforcement learning algorithms (Watkins and Dayan, 1992). Neuromodulation allows an experience to strengthen, weaken, or have no effect on learned Hebbian correlations by associating a *modulatory signal* with the experience. The modulatory signal can be calibrated to guide learning based on an agent's behavior within its environment. Interestingly, neuromodulation has been shown to elicit agent behavior reminiscent of operant conditioning in animals (Soltoggio and Stanley, 2012; Soltoggio et al., 2013). Researchers in neuroevolution and artificial life have also shown that evolutionary algorithms benefit from Hebbian learning combined with neuromodulation by allowing their discovered ANNs to learn from reward signals (Soltoggio et al., 2008, 2007; Soltoggio and Jones, 2009; Soltoggio and Stanley, 2012; Risi and Stanley, 2012; Coleman and Blair, 2012).

A major obstacle to building a general learning system around the Hebbian rule is that its success depends greatly upon receiving inputs that correspond to precisely the domain features necessary to learn the right correlations for the task (Field, 1994). RAAHN introduced the idea of placing an autoencoder, reviewed next, before the Hebbian layer so that such essential features can be learned from raw inputs without the need for human engineering.

Autoencoders

An autoencoder is an ANN that is trained to learn a feature representation of its inputs that is conceptually at a higher level. For example, edge detectors are a higher-level feature of images than raw pixels (Hinton and Salakhutdinov, 2006). The autoencoder achieves such representation by *encoding* its inputs in a hidden layer (the learned feature representation) that is then *decoded* by an output layer (of the same dimensionality as the input layer) representing the autoencoder's *reconstruction* of its inputs. The autoencoder is trained to minimize the error between its reconstruction and its inputs (Bengio et al., 2013). Rather than a different set of weights representing the encoder and the decoder, the same set of weights can compute both the encoding and reconstruction, which is called *tied weights* (Vincent, 2011).

Deep learning researchers attracted fresh interest in autoencoders by showing that they can be stacked into layers that learn increasingly high-level features (Le et al., 2012). That way, for example, raw inputs can be encoded into edge detectors, which can be encoded into increasingly high-level

concepts until face detectors arise. There are many ways to train autoencoders, and many heuristics to help them learn meaningful feature representations (Ranzato et al., 2006; Le et al., 2012), but for RAAHN the precise autoencoder implementation is not the key concern because in theory any autoencoder can be plugged into RAAHN, so as autoencoders improve, RAAHN also improves.

RAAHN

RAAHN was recently introduced by Pugh et al. (2014), who were motivated by the limitations of Hebbian learning to combine an autoencoder with the Hebbian layer so that Hebbian correlations could be learned from the features extracted by the autoencoder. The idea is that in theory both the features and the neuromodulated Hebbian correlations can be learned in real time, as an agent navigates its environment, thereby offering an appealing new approach to reinforcement-like learning. However, this original work relied on an initial *autopilot* phase for RAAHN to learn a feature set representative of the domain. The autopilot in effect ensures that the feature set learned by the autoencoder is reliable and consistent because the learner is forced to encounter a prescripted chronology of experiences. While this setup helps to demonstrate that RAAHN can learn in principle, it is in effect a form of supervised learning, which leaves open the question of whether RAAHN can really learn *on its own* in real time.

Approach

The key hypothesis driving this paper is that RAAHN can still perform well in real time even without an initial autopilot phase if the autoencoder component learns a good feature set. Most of the original RAAHN setup from Pugh et al. (2014) does not need to change, but there are several proposed implementation differences to support fully autonomous learning. The experiments in this paper will apply RAAHN to a similar two-dimensional agent navigation task to that in Pugh et al. (2014), where the agent learns on every simulation tick.

Autoencoder Component

The autoencoder implementation follows conventional practice (Hinton and Salakhutdinov, 2006; Bengio, 2009). In particular, it computes neural activations with tied weights trained with stochastic gradient descent and error backpropagation. Recall that the aim of an autoencoder is to reproduce its own inputs. That way, its hidden layer becomes an *encoding* of the input space that captures its essential underlying features. The forward activation A_j for a hidden neuron j with input neurons I is calculated with

$$A_j = \sigma \left(\sum_{i \in I} (A_i \cdot w_{i,j}) \right), \qquad (2)$$

where σ is the activation function (in this paper the logistic function), A_i is the activation of a given input neuron $i \in I$,

and $w_{i,j}$ is the weight between input neuron i and hidden neuron j. After calculating the forward activations, the backward activation B_i (whereby the reconstruction is computed) for every input neuron is calculated with

$$B_i = \sigma \left(\sum_{j \in H} (A_j \cdot w_{i,j}) \right), \qquad (3)$$

where σ is still the logistic function, and A_j is the previously computed activation for a hidden neuron $j \in H$.

The backward activations thus serve as reconstructions of the original inputs. With these reconstructions, the reconstruction error E_i can be calculated for each input i:

$$E_i = A_i - B_i. \qquad (4)$$

From these error values the delta δ_i is calculated for each input neuron i, which will help to compute backpropagated error:

$$\delta_i = E_i \cdot \sigma'(B_i), \qquad (5)$$

where σ'_i is the derivative of the logistic function at the reconstruction B_i for the input neuron i. Now the delta for a given hidden neuron j can be calculated as

$$\delta_j = \left(\sum_{i \in I} (\delta_i \cdot w_{i,j}) \right) \sigma'(A_j), \qquad (6)$$

where δ_i is from equation 5, $w_{i,j}$ is from equation 2, and $\sigma'(A_j)$ is the derivative of the logistic function at the forward pass A_j for hidden neuron j.

With the original error deltas δ_i and backpropagated error deltas δ_j, the tied weights of the autoencoder component can be updated. The change in weight $\Delta w_{i,j}$ for the connection from input neuron i to hidden neuron j is then

$$\Delta w_{i,j} = \alpha \left(\delta_i A_j + \delta_j A_i \right), \qquad (7)$$

where α is the learning rate (held constant in later experiments at 0.1), and the remaining variables as described above.

In RAAHN this autoencoder is trained in real time as the agent explores its world, raising the question of on what data it should be trained at any given tick. To address this question a *history buffer* saves n experiences (sets of rangefinder values) that the agent encounters in the domain. However, the method through which these n experiences are chosen turns out instrumental in facilitating the novel real-time exploratory autonomy of RAAHN investigated in this paper, as explained next.

History Buffer Management

A simple way to manage the history buffer is to save every experience as it is discovered and drop the oldest ones when the buffer is at capacity. This method is called **Queue-RAAHN** because the history buffer is in effect managed as

a queue. It gives RAAHN the ability to learn a feature set from past experiences in the domain, but it also has the significant downside that the history buffer may accumulate too much of a single kind of experience (e.g. crash experiences), eventually causing it to "forget" other important aspects of the domain. While this approach worked in research by Pugh et al. (2014) under the controlled environment of the autopilot, when RAAHN is allowed to run autonomously the chance of a succession of negative experiences like crashes is much higher.

To address this problem the history buffer can instead be managed by saving only the most *novel* experiences encountered during the agent's lifetime. That way, commonly repeated experiences (such as crashing into a wall), will not come to occupy the entire buffer. This new method is called **Novelty-RAAHN**. To determine the novelty of an experience the Euclidean distance is calculated between it and all the other experiences in the buffer. Inspired by the calculation of novelty in the novelty search algorithm (Lehman and Stanley, 2011), the experience is then assigned a *novelty score*, which is the sum of the 20 smallest such distances. The current experience is then added to the buffer only if its novelty score is greater than that of the least novel experience in the buffer, which it replaces. That way, the buffer fills with diverse rather than redundant experiences that give the autoencoder a representative sample of the entire domain.

Hebbian Component

The Hebbian component of RAAHN, which learns a controller based on the features from the autoencoder, trains with the *reconfigure and saturate* method described by Soltoggio and Stanley (2012). This method adds a modulatory signal (which can either be positive or negative) to the basic Hebbian learning rule, thus giving reward and penalty feedback to the otherwise-naive Hebbian component. The modulatory signal m influences the weights through the update equation

$$\Delta w_i = m\eta x_i y, \tag{8}$$

where η is the learning rate, and x_i and y are the activations of the two neurons connected by the weight w_i. As prescribed by reconfigure and saturate, noise is added to each output and weight delta to facilitate exploration (which is essential when there is no autopilot). Thus the noisy activation A_j for each output neuron is computed as

$$A_j = \sigma \left(\sum_{i \in I} (x_i w_{i,j}) \right) + \xi_j, \tag{9}$$

where ξ_j is the neural noise associated with A_j. Noise is added to each weight delta with

$$\Delta w_i = m\eta x_i y + \xi_i, \tag{10}$$

where ξ_i is noise sampled from the same distribution as ξ_j in equation 9 (in this paper: a uniform distribution in the range $[-0.1, 0.1]$).

Hebbian and RAAHN Architectures

The experiments in this paper compare pure Hebbian learning and RAAHN in a two-dimensional agent navigation domain. The raw inputs for both pure Hebbian and RAAHN come from 11 simulated wall-sensing rangefinders. Both network architectures output a single value denoting the fraction of the maximal turn angle to steer the agent. The pure Hebbian architecture is simple in that it merely connects the raw inputs directly to the output with a single layer of Hebbian-trained connections (figure 1a). The RAAHN architecture includes two layers of weights and an intervening five-neuron hidden layer (figure 1b), where the first layer of weights is trained by the autoencoder learning rule and the second layer is Hebbian-trained. Therefore, RAAHN's Hebbian component essentially learns driving behavior from a set of five higher-level features extracted from the raw inputs by the autoencoder component. The number five, which preliminary experiments suggest is not sensitive to minor variation, is chosen to avoid learning only the identity function.

Experiments and Results

Recall that the hope in this paper is to advance beyond the previous finding that RAAHN can navigate a two-dimensional maze domain with an initial autopilot-driven training phase (Pugh et al., 2014). Although that study established that RAAHN can learn features as it learns to control, the more exciting potential of RAAHN is the ability to learn by itself, without an autopilot, in the spirit of real organisms.

By promising to learn *new* features at the same time as it learns a control policy even without any preliminary training, RAAHN adds a novel capability over and above what modulated Hebbian plasticity can offer. However, that new capability also raises the possibility that RAAHN might be overall more difficult to train without the help of the autopilot. For that reason, the experiments that follow establish first that RAAHN remains competitive with Hebbian on problems that Hebbian can solve. Once that is established, showing in effect that the new capabilities of RAAHN cost very little, the next logical step is an experiment that shows that on some problems (where feature learning is essential), RAAHN becomes critical to making effective learning possible.

Queue-RAAHN Experiment

To explore the potential of RAAHN to learn through its own exploration, a two-dimensional maze experiment similar to the one carried out by Pugh et al. (2014) is conducted without the initial autopilot. Instead, the agent is allowed to learn from its own decisions as it explores the world, as described in the Approach section.

Queue-RAAHN manages its history buffer as a queue. Every experience is saved in real time and the oldest experiences are deleted when the buffer is at capacity. Recall that the experiences in the buffer will periodically train the autoencoder in real time. This simple queue-based approach to storing

(a) Hebbian Architecture

(b) RAAHN Architecture
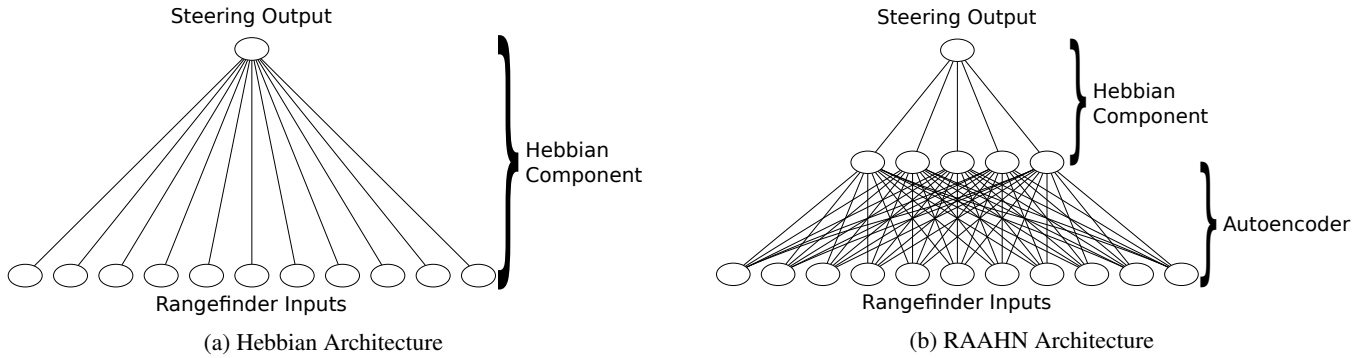
Figure 1: **Hebbian and RAAHN Architectures.** The Hebbian architecture (a) simply connects the 11 rangefinder inputs to the one output corresponding to the agent's steering. The RAAHN architecture (b) introduces an autoencoder layer between the rangefinder inputs and the steering output. The first layer of connections is trained as the autoencoder with tied weights, to learn (in the experiment) five high-level features from the rangefinder inputs. These features are then connected in RAAHN to the steering output with Hebbian-trained connections.

experiences gives the agent memory of previous experiences so it does not immediately forget them.

To test this approach (and others to be introduced shortly), the agent is given 11 rangefinders (depicted in figure 2) to detect how close it is to nearby walls. The rangefinders are separated from each other by an angle of 18 degrees. The two rangefinders at the edges of the range are 90 degrees from the middle rangefinder. Each rangefinder in this first experiment is 350 units long; the whole track is 3,140 units from left to right and 2,160 units from top to bottom. Rangefinders produce a minimum activation of 0.0 when they do not intersect walls, and a maximum activation of 1.0 when the entire rangefinder intersects a wall. Intermediate intersections produce an activation between 0.0 and 1.0.

Queue-RAAHN is compared to a single-layer neural network with Hebbian connections. The aim is to show that the autoencoder layers in RAAHN, which are lacking in the Hebbian network, do not diminish the ability of RAAHN to learn on its own in real time compared to the Hebbian layer alone. The agents controlled by Queue-RAAHN and the Hebbian neural network are referred to as **Queue-RAAHN** and **Hebbian**, respectively. Both neural network topologies take the 11 inputs and produce one output denoting the turning angle, with a range of [-2.0, 2.0]. A turn angle output of 2.0 changes the agent's direction by 2.0 degrees for the given tick. The neural network topology of Queue-RAAHN includes an autoencoder with a hidden layer of five neurons to learn features from the 11 rangefinder inputs, as depicted in figure 1b. The history buffer size for Queue-RAAHN is 500. For both methods Hebbian training occurs once every tick based on only the most recent experience. In Queue-RAAHN the autoencoder component also trains on 20 randomly-selected experiences from the history buffer (which for Queue-RAAHN is of course managed as a queue). The learning rates for autoencoder and Hebbian training (for the Hebbian layer of RAAHN and the pure Hebbian network) are held constant

at 0.1 and 1.0 respectively. Both the single-layer Hebbian network and the Hebbian component of Queue-RAAHN receive positive modulation for turning away from walls, and negative modulation for turning towards walls, in the range of [-1.0, 1.0], where the magnitude of modulation is proportional to the magnitude of the turn. The wall chosen for this calculation is the closest wall colliding with an imaginary line projecting from the center of the agent 400 units in the direction the agent is facing. If the imaginary line does not intersect any wall then modulation is zero.

The simulation is run 200 times for both methods, each time for 10,000 ticks (i.e. simulation state updates). Agents that fail to complete at least one lap are *complete failures*. Agents that complete more than one lap but fewer than four
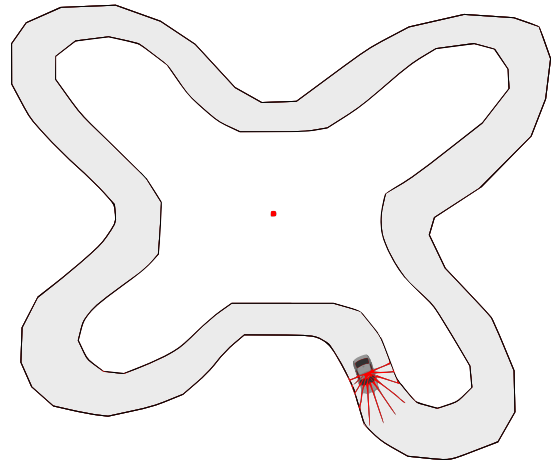
Figure 2: **X-Shaped Domain.** The track is not uniform to ensure the agent does not simply just repeat one behavior several times. The red dot in the center determines agent performance; every time the agent completes a circle around the dot, it completes one lap.

laps are *partial failures* because their performance is less than half that of correctly-driving agents.

Hebbian on average completes 9.2 laps and exhibits no complete failures nor partial failures. However, Queue-RAAHN on average completes 7.8 laps (which is significantly less at $p < 0.01$; Student's t-test) and yields 14% complete failures and a further 1% partial failures. The failure of Queue-RAAHN to approximate the performance of the plain Hebbian network suggests that the queue-based learning approach does not enable RAAHN to approach optimal performance.

Observing runs visually in real time reveals that after colliding with a wall, Queue-RAAHN agents either escape after a few hundred ticks or else become stuck for the duration of the run. Interestingly, an analysis of Queue-RAAHN neural networks suggests that this behavior results from poor driving compounded with poor representation. That is, sometimes agents drive poorly initially, which leads them to fill their history buffer with only crash experiences. This misadventure then leads to learning a poor feature set also representative only of crash experiences. The poor feature set then makes it difficult for such agents to escape their perpetual crashing and learn the types of better driving behaviors employed by agents who complete more laps.

Thus one hypothesis is that the main problem with Queue-RAAHN is that the autoencoder is unable to learn a feature set that is representative of the entire domain. If this hypothesis is true then if the feature set is constrained to be *novel* and therefore representative of the entire domain, RAAHN should perform about as well as Hebbian, which is tested next.

Novelty-RAAHN Experiment

By constraining the history buffer of RAAHN to contain only the most novel experiences, the data set on which RAAHN trains can become representative of the entire domain. For example, when the agent crashes into a wall, because crash experiences tend to be similar, they are not able to flood the buffer the way they do in Queue-RAAHN. By thereby avoiding a buffer with only one or few kinds of experience, the history buffer accumulates experiences of the entire domain as the agent explores it. This experiment uses the same parameters as the previous experiment aside from the difference in history buffer management. The novelty constraint on the history buffer still allows the simulation to run in real-time, as can be observed in the source available at:

http://eplex.cs.ucf.edu/uncategorised/software

The simulation is run 200 times for 10,000 ticks each. On average Novelty-RAAHN completes 8.8 laps, which is significantly above the 7.8 laps achieved by Queue-RAAHN ($p < 0.05$; Student's t-test). While the 8.8 laps of Novelty-RAAHN is still significantly below the 9.2 of Hebbian alone ($p < 0.01$; Student's t-test), this difference is small (less than one lap), and moreover *some* small disparity is essential in

practice because Novelty-RAAHN must consume some extra time at the beginning of each run acquiring a novel set of experiences. Novelty-RAAHN also suffers only 0.5% complete failures and no partial failures. Thus it is likely close to performing as well as possible for a method that learns both features and policy at the same time. In conclusion, in this task in which the Hebbian network is provided a good input representation, RAAHN can learn in real time to approximate the same performance all while learning its feature representation in real time as well.

Increased Rangefinder Length Experiment

While it is important to establish that RAAHN can learn a representation in real time competitive with Hebbian alone, RAAHN's real promise is to learn *better* feature representations that overcome the limitations of the raw sensors. One way to investigate this idea is to *degrade* the quality of the rangefinders by increasing their length. Such an increase makes distinguishing different situations more difficult by forcing more sensory input into the agent as the sensors intersect walls more frequently and with greater activation. In theory RAAHN can overcome this challenge to some extent because it learns a new representation from the sensory input. However, Hebbian is forced to learn from degraded inputs.

To investigate whether RAAHN can indeed gain an advantage by learning a new feature representation, this experiment tests Hebbian and Novelty-RAAHN over ten variations of rangefinder lengths. These variants range from 10% longer to 100% longer with an interval of 10% between each variant.

The simulation is run for each variant 100 times, each for 10,000 ticks. As the rangefinder length increases both Hebbian and Novelty-RAAHN experience significantly more failures. However, Novelty-RAAHN indeed exhibits far fewer failures (figure 3). Hebbian begins to experience dozens of failures as early as 30%, 40%, and 50% longer while Novelty-RAAHN experiences fewer than four complete failures and fewer than seven partial failures at the same lengths. In addition, the number of laps completed by Novelty-RAAHN is significantly greater from 30% onward ($p < 0.05$). Novelty-RAAHN is only affected eventually by dramatically increasing the rangefinder lengths to values that provide little discernible information. Thus Novelty-RAAHN is significantly less sensitive to the precise sensory setup than Hebbian.

Discussion and Future Work

The experimental results establish for the first time that RAAHN is able to learn effective maze navigation behavior without the need for an autopilot. This achievement opens up a wide range of application domains because it means RAAHN does not need knowledge of the problem domain a priori. Interestingly, the implication is also that RAAHN can now be applied to conventional RL problems where training data is not labeled because now RAAHN only needs a modulation scheme to learn Hebbian correlations. Of course,
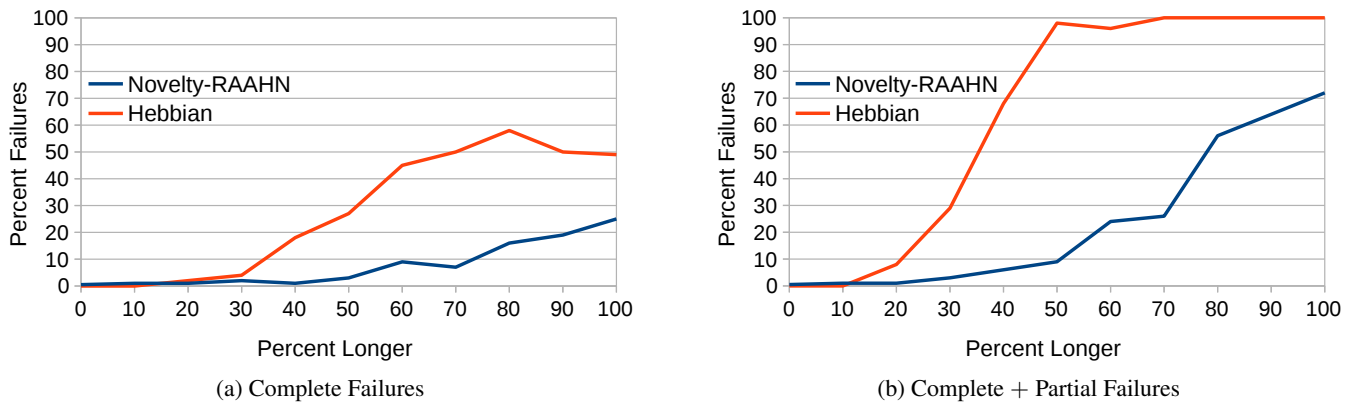
Figure 3: **Complete and Partial Failures (lower is better).** The number of complete failures (a) and complete plus partial failures (b) is shown for Hebbian and Novelty-RAAHN for rangefinder length increments between 0% and 100%. In both cases, the fact that Novelty-RAAHN completes significantly more laps in all cases from 30% longer onward ($p < 0.01$ for all 30% and above) is reflected in the higher number of runs that fail for Hebbian.

because RAAHN is significantly different from conventional RL algorithms such as Q-learning (Watkins and Dayan, 1992) or SARSA (Rummery and Niranjan, 1994) its strengths and weaknesses are likely different as well, but the opportunity for an entirely new path of research in this direction offers the potential for new discoveries and insights that would not emerge from conventional RL.

For example, already in this paper we begin to glimpse some principles behind learning a useful feature set in real time. In particular, it is likely that Queue-RAAHN performs significantly worse than Novelty-RAAHN because it has a tendency to "forget" salient aspects of the domain if they are not constantly revisited, as is the case when an agent crashes into a wall for an extended period of time: eventually its buffer becomes filled entirely with crash experiences that offer little utility beyond the scenario of a crash. Novelty-RAAHN avoids this trouble by only forgetting experiences that are redundant, instead attempting to maintain a set of experiences representative of the entire domain. The success of the novelty-driven buffer in Novelty-RAAHN thus hints at the importance of gathering and retaining experience in a principled manner.

Advances in autoencoders, which provide RAAHN the ability to represent higher-level concepts related to its domain, also lead to possible enhancements to RAAHN as well. In this way, RAAHN's potential reaches beyond simply calibrating sensitivity to a range of input parameters. Rather, as the domains in which RAAHN is applied become more complex, so does the possibility for more interesting feature sets. RAAHN can potentially stack several autoencoders (Hinton and Salakhutdinov, 2006) to learn high-level features from the complex data of vision, audition, or any other sensory modalities. The correlations learned by the Hebbian component can then serve to respond to those features in real time. It may also be possible to adapt RAAHN archi-

tectures through neuroevolution (Stanley and Miikkulainen, 2002; Floreano et al., 2008; Yao, 1999). If the topology of a RAAHN architecture can change over evolution, the process of finding an effective architecture (including autoencoder stacks) could be automated.

Furthermore, by pairing Hebbian learning with an autoencoder, much more becomes possible through Hebbian modulation than would be possible in a simple Hebbian network alone (without an autoencoder), thereby breathing new life into research focused on Hebbian learning. Supporting this view, when rangefinder lengths are extended beyond what is optimal, RAAHN exhibits significantly fewer failures than Hebbian alone. The performance of Hebbian alone degrades quickly without the autoencoder because Hebbian learns correlations best with sparse input activations (Olshausen and Field, 2004). With very long rangefinders, most of the inputs are highly active at any given time, so Hebbian cannot learn meaningful correlations. RAAHN's performance does not degrade as quickly because its autoencoder transforms the highly active inputs into more distributed features better-suited for the Hebbian component to learn effective driving behavior. *Sparse autoencoders* (Le et al., 2012) might help to limit such degradation even further.

As an example of how recent work in Hebbian learning can enhance RAAHN, new ideas on augmenting Hebbian connections to react to distal rewards (Soltoggio, 2015) (i.e. rewards from far away in time) can potentially shift RAAHN from its current limited temporal context to learning long-term causal dependencies. Recurrent connections might further allow learning to react to experiences from the past. These possibilities in effect draw on advances in Hebbian learning in general, and provide fuel for further research into improving Hebbian learning.

Finally, as a novel approach to RL, RAAHN aligns naturally with research in alife because agent behavior is shaped

in RAAHN through low-level neuromodulation as opposed to high-level value-function approximation (as in Q-learning and SARSA), making its analogy to low-level biological processes (in particular Hebbian plasticity) more accessible and open to study. Future work will focus on more complex domains that require for example behaving in location-dependent contexts by learning in real time the identifying features of different locations.

Conclusion

Moving beyond the original demonstration of RAAHN (Pugh et al., 2014) that depended on an initial autopilot phase, this paper showed how RAAHN can explore and learn on its own without an autopilot phase. By maintaining a buffer of experiences representative of the domain, such real-time learning becomes realistic. This new capability was demonstrated in a robot control domain where RAAHN learned to steer a robot through hallways on its own from scratch. The benefit of the architecture was further demonstrated by showing how much less its performance degrades compared to a simple Hebbian network when the sensory inputs become less optimal. The long-term implication is that RAAHN is a new sandbox and a new model for experimenting with modulation and reinforcement learning, which in the future can benefit from advances in both Hebbian learning and autoencoders alike.

References

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pages 1798–1928.

Coleman, O. J. and Blair, A. D. (2012). Evolving plastic neural networks for online learning: review and future directions. In *AI 2012: Advances in Artificial Intelligence*, pages 326–337. Springer.

Field, D. J. (1994). What is the goal of sensory coding? *Neural Computation*, 6(4):559–601.

Floreano, D., Dürr, P., and Mattiussi, C. (2008). Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1:47–62.

Floreano, D. and Urzelai, J. (2000). Evolutionary robots with on-line self-organization and behavioral fitness. *Neural Networks*, 13:431–4434.

Hebb, D. O. (1949). *The Organization of Behavior: A Neuropsychological Theory*.

Hinton, G. and Salakhutdinov, R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

Le, Q., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G., Dean, J., and Ng, A. (2012). Building high-level features using large scale unsupervised learning. In *International Conference on Machine Learning (ICML-2012)*.

Lehman, J. and Stanley, K. O. (2011). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223.

Niv, Y., Joel, D., Meilijson, I., and Ruppin, E. (2002). Evolution of reinforcement learning in uncertain environments: A sim-ple explanation for complex foraging behaviors. *Adaptive Behavior*, 10(1):5–24.

Olshausen, B. A. and Field, D. J. (2004). Sparse coding of sensory inputs. *Current opinion in neurobiology*, 14(4):481–487.

Pugh, J. K., Soltoggio, A., and Stanley, K. O. (2014). Real-time hebbian learning from autoencoder features for control tasks. In *Proceedings of the Fourteenth International Conference on Artificial Life (Alife XIV)*, Cambridge, MA. MIT Press.

Ranzato, M., Poultney, C., Chopra, S., and LeCun, Y. (2006). Efficient learning of sparse representations with an energy-based model. In et al., J. P., editor, *Advances in Neural Information Processing Systems (NIPS 2006)*, volume 19. MIT Press.

Risi, S., Hughes, C., and Stanley, K. (2011). Evolving plastic neural networks with novelty search. *Adaptive Behavior*, 18(6):470–491.

Risi, S. and Stanley, K. O. (2010). Indirectly encoding neural plasticity as a pattern of local rules. In *Proc. of the 11th Intl. Conf. on Simulation of Adaptive Behavior*, Berlin. Springer.

Risi, S. and Stanley, K. O. (2012). A unified approach to evolving plasticity and neural geometry. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN-2012)*, Piscataway, NJ. IEEE.

Rummery, G. A. and Niranjan, M. (1994). On-line Q-learning using connectionist systems. Technical Report TR 166, Cambridge University Engineering Department, Cambridge, UK.

Soltoggio, A. (2015). Short-term plasticity as cause–effect hypothesis testing in distal reward learning. *Biological Cybernetics*, 109(1):75–94.

Soltoggio, A., Bullinaria, A. J., Mattiussi, C., Drr, P., and Floreano, D. (2008). Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios. In Bullock, S., Noble, J., Watson, R., and Bedau, M., editors, *Proc. of the 11th Intl. Conf. on Artificial Life*, Cambridge, MA. MIT Press.

Soltoggio, A., Dürr, P., Mattiussi, C., and Floreano, D. (2007). Evolving neuromodulatory topologies for reinforcement learning-like problems. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 2471–2478.

Soltoggio, A. and Jones, B. (2009). Novelty of behaviour as a basis for the neuro-evolution of operant reward learning. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, GECCO '09, pages 169–176, New York, NY, USA. ACM.

Soltoggio, A., Lemme, A., Reinhart, F., and Steil, J. J. (2013). Rare neural correlations implement robotic conditioning with delayed rewards and disturbances. *Frontiers in neurorobotics*, 7(6).

Soltoggio, A. and Stanley, K. O. (2012). From modulated hebbian plasticity to simple behavior learning through noise and weight saturation. *Neural Networks*, 34:28–41.

Stanley, K. O., Bryant, B. D., and Miikkulainen, R. (2003). Evolving adaptive neural networks with and without adaptive synapses. In *Proceedings of the 2003 Congress on Evolutionary Computation*, Piscataway, NJ. IEEE.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10:99–127.

Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674.

Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*, 8(3-4):279–292.

Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447.

Body Representations for Robot Ego-Noise Modelling and Prediction. Towards the Development of a Sense of Agency in Artificial Agents

Guido Schillaci[1], Claas N. Ritter[1], Verena V. Hafner[1], Bruno Lara[2]

[1]Adaptive Systems Group, Department of Computer Science, Humboldt-Universität zu Berlin, Berlin, Germany
[2]Cognitive Robotics Group, Center for Science Research, Universidad Autonoma del Estado de Morelos, Cuernavaca, Mexico
Corresponding author: guido.schillaci@informatik.hu-berlin.de

Abstract

We present an implementation of a biologically inspired model for learning multimodal body representations in artificial agents in the context of learning and predicting robot ego-noise. We demonstrate the predictive capabilities of the proposed model in two experiments: a simple ego-noise classification task, where we also show the capabilities of the model to produce predictions in absence of input modalities; an ego-noise suppression experiment, where we show the effects in the ego-noise suppression performance of coherent and incoherent proprioceptive and motor information passed as inputs to the predictive process implemented by a forward model. In line with what has been proposed by several behavioural and neuroscience studies, our experiments show that ego-noise attenuation is more pronounced when the robot is the owner of the action. When this is not the case, sensory attenuation is worse, as the incongruence of the proprioceptive and motor information with the perceived ego-noise generates bigger prediction errors, which may constitute an element of surprise for the agent and allow it to distinguish between self-generated actions and those generated by other individuals. We argue that these phenomena can represent cues for a sense of agency in artificial agents.

Introduction

Empirical evidence from cognitive science and neuroscience suggests that we, as humans, maintain an internal representation of our body, or a model of our motor system, and that such an internal model would be involved in processes of simulation of sensorimotor activity. These processes would affect the way we experience the interaction with the environment and would be fundamental for the implementation of basic cognitive skills. For example, simulation processes are thought to have a role in the way we differently perceive self-generated actions or actions performed by other subjects. One of the proposals that explains this phenomenon (Blakemore et al., 2000a,b) says that when we perform a motor action, an efferent copy of the motor commands that our brain sends to our muscles would be used in a predictive process that anticipates the sensory outcomes of the movement. Such predictions would then be compared to the actual sensory consequences and, if the two correspond, the perceived sensory consequences are attenuated. This would enable a differentiation between self-generated sensory events and those externally generated that are not mapped to any internally generated efferent copy of the motor commands (Blakemore et al., 2000a). The existence of such a self-monitoring mechanism would explain, for example, why tickling sensations cannot be self-produced (Blakemore et al., 2000b), why people are better at recognising themselves than others when watching movies of only point-light walkers (Loula et al., 2005), why people are more accurate in predicting the landing point of a thrown dart from a video screen when they observe their own throwing action than when observing another person's throwing action (Knoblich and Flach, 2001), or why people perceive the loudness of sounds as less intensive when they are self-generated, than when they are generated by other persons or by a software (Weiss et al., 2011). In this latter study on selective attenuation of self-generated sounds, the authors proposed that the experience of perceiving actions as self-generated would be caused by the anticipation and, thus, the attenuation of the sensory consequences of such motor commands, which would be related to "the privileged access to internally generated efferent information during one's own action" (Weiss et al., 2011). The sense of agency, that is the pre-reflective experience that *we* are the owner of an action we are executing, is thus proposed to be dependent on the degree of congruence vs. incongruence between predicted and actual sensory consequences of our bodily actions.

In the investigation on sensorimotor simulation processes in the human brain, internal forward and inverse models have been proposed (Wolpert et al., 2001). A forward model (illustrated in Figure 1) - or predictor, as firstly proposed in the control literature as a means to overcome problems such as the delay of feedback in control strategies - incorporates knowledge about sensory outcomes of self-generated actions. Inverse models - or controllers, as they were initially proposed for implementing inverse kinematics processes for controlling robotic manipulators - perform the opposite transformation providing a system with the necessary motor command to go from an initial sensory situation to a desired
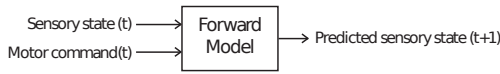
Figure 1: An illustration of the forward model (predictor).

one. Such models encode the dynamics of the motor system and can provide artificial agents with multimodal representations, as they fuse together sensory and motor information (Wilson and Knoblich, 2005), and with the capability to predict sensorimotor activities based on previous experience. Studies such as the ones reported above shed light on the importance that predicting sensory consequences of self-generated actions have for basic motor tasks and cognitive skills. Equipping artificial agents with similar computational processes has been shown to be a promising approach in the development of different skills, such as navigation (Möller and Schenck, 2008; Escobar et al., 2012), perception of the functional role of objects (Kaiser, 2014), action selection and tool-use (Schillaci et al., 2012) and sense of agency (Pitti et al., 2009).

The work presented here adopts a biologically inspired framework for internal body representations (Schillaci et al., 2014) that can enable a robot with the capability to perform simulations of sensorimotor activities based on previous experience. Inspired by human development, the learning of this body representation is intertwined with the interaction experience of the robot with the external environment. In particular, we frame this work into the context of one of the biggest - and most unexplored - challenges of robot audition, the artificial capability of listening, that is the presence of *ego-noise*, or the noise that the robot generates while moving around. Being able to estimate self-induced changes in the auditory signal is not only crucial for attenuating the noise, and thus for enhancing the auditory signal for further processing such as speech recognition, but also for distinguishing ego-noise from other sounds in natural acoustic environments, which is a prerequisite for efficient and intuitive interaction with other people and with the surroundings.

We demonstrate the predictive capabilities of the model in the auditory domain in two tasks. Firstly, we introduce the framework in a simple classification task, where the robot has to recognise a behaviour that it executes based on the comparison of the produced ego-noise to internal simulations of ego-noise produced by *intended* actions. We also show how our model can deal with the situation when input information are missing, for example by simulating a damage in the system, resulting in the model still being able to classify, although with poorer performance. Secondly, we show how the proposed framework, and the predictive capabilities that it provides, could serve as a basis for the development of a sense of agency in artificial agents. In particular, we report an experiment on ego-noise attenuation based on sensorimotor predictions, where the quality of the attenua-

tion is dependent on the degree of congruence vs. incongruence between predicted and actual sensory consequences of self-generated actions. In line with the behavioural studies reported above, we show that prediction errors generated by internal sensorimotor simulations are smaller when the proprioceptive information is coherent with the events that are perceived from the external environment. Simply put, we show that sensory attenuation is more pronounced when the robot is the owner of the action, and we argue that this could serve as a cue for self-agency in artificial agents.

In the rest of the paper we firstly introduce the framework presented in (Schillaci et al., 2014) and extend it. Therefore, we illustrate and discuss the experiments mentioned above. Finally, we draw the conclusions and the outlines of future work.

An Internal Body Representation for a Humanoid Robot

Evidences from behavioural sciences and neuroscience suggest that motor and brain development are strongly intertwined with the experiential process of exploration, where internal body representations would be formed and maintained over time (Cang and Feldheim, 2013). Kaas (1997) reported the existence of topographic maps in the visual, auditory, olfactory and somatosensory systems, as well as in parts of the motor brain areas. Researchers proposed that such maps would self-organise throughout the brain development and along the sensorimotor experience of the individual with the external environment. They would function as projections of sensory receptors and of effector systems, and are arranged in a way that adjacent regions process spatially close sensory parts of the body. Many studies support the existence of an integrated representation of visual, somatosensory, and auditory peripersonal space in human and non-human primates (see for example Holmes and Spence (2004)), suggesting that the brain maintains integrated multimodal representations, which are essential for sensorimotor control (Maravita and Iriki, 2004).

During the last couple of decades, interest in the possibility to develop models inspired by the mechanisms of human body representations has been growing also in the robotics community. In robot audition, for example, Ince and colleagues investigated methods for learning, predicting and suppressing robot ego-noise (Ince et al., 2009). The authors built up an internal body representation of a humanoid robot consisting in motor sequences mapped to the recorded motor noises and their spectra. This resulted in a large noise template database that was then used for ego-noise prediction and subtraction.

Here, we report an implementation of a biologically inspired model for body representations that can encode experience gathered through sensorimotor learning and that can generate predictions of auditory and motor states. In particular, we propose an internal models framework consist-

ing of connected neural networks that simulate distinct sensorimotor brain areas. The internal model encodes sensory and motor modalities as topographic maps that self-organise throughout the interaction of the robotic agent with the external environment. Moreover, a parallel intermodal mapping is performed: sensory and motor maps are connected through Hebbian links that are strengthened when an occurrence of multi-modal activity is observed.

The model architecture is inspired by the Epigenetic Robotics Architecture (Morse et al., 2010), where a structured association of multiple Self-Organising Maps (SOMs) (Kohonen, 1982) is adopted for mapping different sensorimotor modalities in a humanoid robot, and it is based on similar works we previously published (Kajić et al., 2014; Schillaci et al., 2014). Self-organising maps have the advantage of producing low-dimensional and discretised representations of the input space of the training samples.

In the proposed model, multiple SOMs, each representing a sensory or motor modality, are associated through unidirectional Hebbian links: each node of the input map is connected to each node of the output map, where the connection is characterised by a weight. The weight is updated according to a positive Hebbian rule that simulates synaptic plasticity of the brain: the connection between a pre-synaptic neuron (a node in the input map) and a post-synaptic neuron (a node in the output map) increases if the two neurons are simultaneously activated. Learning of the internal model consists in updating the SOMs and the Hebbian connections with sensory and motor data gathered through an exploration behaviour executed by the robot. During the execution of the robot movements, sensory and motor data are provided as training inputs to the corresponding maps in an online fashion. A SOM is constructed as a grid of neurons, where each neuron is represented as an n-dimensional weight vector $\mathbf{w_i}$ (Kajić et al., 2014; Kohonen, 1982). The number of dimensions of a weight vector corresponds to the dimensionality of the input data. Weights in the network are initially set to random values and then adjusted iteratively by presenting the input vector x_p. In each iteration, the winning neuron i is selected as a neuron whose weights are closest to the input vector in terms of the Euclidean distance. After selecting a winning neuron, the weights of all neurons are adjusted:

$$\Delta w_j = \eta(t)h(i,j,t)(w_j - x_p) \tag{1}$$

The parameter $\eta(t)$ is a learning rate which defines the speed of change. The function $h(i,j,t)$ is a Gaussian neighborhood function defined over the grid of neurons as:

$$h(i,j,t) = e^{\left(\frac{w_i^2 - w_j^2}{2\pi\sigma(t)^2}\right)} \tag{2}$$

The learning rate $\eta(t)$ and the spread of the Gaussian function $\sigma(t)$ are held constant for a certain time interval,

and are annealed exponentially afterwards.[1] The function is centered around the winning neuron i and its values are computed for all neurons j in the grid. The spread of the function determines the extent to which neighbouring weights of a winning neuron are going to be affected in the current iteration. The topology of the network is preserved by pulling together neurons towards the winning node.

After every update of the SOMs, the Hebbian links connecting each pair of maps are updated as well. The Hebbian update corresponds to the following steps. For mapping an input map (e.g. the motor map) to an output map (e.g. the auditory map):

- select the pre-synaptic neuron (winner node) as the closest node i in the input map to the current input pattern $\mathbf{x}$ (e.g. the joint rotation);

- select the post-synaptic neuron (winner node) as the closest node j in the output map to the current output pattern $\mathbf{y}$ (e.g. the robot ego-noise);

- strengthen the connection w_{ij} between the pre and post-synaptic neurons according to the modified positive Hebbian rule:

$$\Delta w_{ij} = \lambda A_i(\mathbf{x})A_j(\mathbf{y}) \tag{3}$$

where $A_i(\mathbf{x})$ is the activation function of the neuron i over the Euclidean distance between the neural weights and the data pattern $\mathbf{x}$, λ is a learning rate used for slowing down the growth of the weights (in the experiments presented here, it is initialised to 0.1). The activation function of a neuron, $A(\mathbf{d})$, is computed as:

$$A(d) = \frac{1}{1 + 2\tanh(d)} \tag{4}$$

where d is the normalised Euclidean distance between the position of the node and the input pattern.

After the update, a normalisation is performed on all the links from the input map converging to a node in the output map, for each node in the output map, as described by Miikkulainen (1990). Such a normalisation implements a forgetting process, since it strengthens the updated link and it weakens all the other connections. The same process is performed on the unidirectional links connecting each pair of maps in the model in both directions.

The trained model can be used for performing sensory and motor predictions. Predictive processes can be activated by querying the model with partial or full sensorimotor information. For example, we can infer the ego-noise produced by the execution of a specific motor command (forward prediction) from the model depicted in Figure 2 by querying the model with an input to the proprioceptive map, consisting of the joints configuration of the robot, and an input to the motor map, consisting of the joints rotations, which are

---

[1]In the experiments presented in the the following section, we set $\eta = 0.9$ and $\sigma = 0.7$.

392

therefore propagated to the auditory map. In fact, a predictive system based on propagation of signals between maps has been implemented. The propagation of signals works as follows. Given a sensory or motor input:

- Find the winner node w and its k neighbours (k set to 5, in the experiments presented here) in the corresponding map, as the closest node to the input, and calculate its activation using the activation function described in (4);

- Propagate the activation of the nodes in the winners list of the input map to all the nodes in the output map connected to it. The propagated value to each node in the output map is equal to the activation of the selected node in the input map multiplied by the weight of the Hebbian link connecting the selected node in the input map to the corresponding node in the output map; multiple propagations to the same node in the output map are summed up;

- Compute the prediction in the output modality as the weighted average of the positions of the nodes in the output map, each weighted by the incoming propagation.

If an observation of the output modality is available, a *prediction error* can be computed as the distance between the predicted outcome and the observation. Moreover, multiple propagations can be executed from different input modalities to the same output modality, as illustrated in Figure 2. From each input modality, signals can be spread out to the desired output modality as described above. Thus, incoming propagations onto the output map can be summed up and a prediction can be computed as the weighted average of the nodes' positions multiplied with their activations.
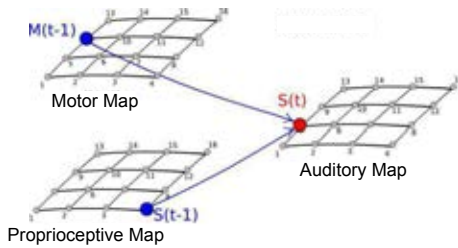
Figure 2: An example of a forward model consisting of three maps.

Figure 2 illustrates a forward model (as described in Figure 1) implemented using the proposed architecture composed of three SOMs: a proprioceptive map, encoding the initial joint configuration of the robot, a motor map, i.e. encoding the rotation applied to the joints from the initial positions, and an auditory map, encoding the noise produced by the movements. An inverse model can be implemented with two sets of directional Hebbian links: the first starting from the proprioceptive map and ending at the motor map, and the second starting from the auditory map and ending at the motor map.

Ego-noise representation

We represent the ego-noise produced by the robot movements using Mel-frequency cepstral coefficients (MFCCs), which are features derived from a type of cepstral representation of the auditory signal commonly used in speech recognition (Sahidullah and Saha, 2012). In this work, MFCC features are derived performing the following steps:

- Calculate the Fourier transform of an audio chunk. In the experiments reported here, we used a single channel audio signal, recorded from the robot with a sampling rate of 48 kHz. Audio chunks of 40 ms are extracted from the signal using a rectangular window. Chunks are extracted every 20 ms (that is, with a 50% overlap between subsequent chunks). FFT size is 2048 samples. 32 triangular overlapping filters are used in the Mel filterbank, with a mel filter width of 200. The frequency range of the filterbank goes from 0 to 16 kHz;

- Apply the Mel filterbank to the power of the spectrum and sum up the energy in each filter;

- Calculate the Discrete Cosine Transform of the logarithm of the filterbank energies;

- Keep the first 26 or 32 coefficients of the DCT as MFCC features.

For implementing the MFCC feature extraction process, we adopted and extended an existing open source and cross-platform digital signal processing library, named Aquila DSP (http://aquila-dsp.org/). Before being processed, input data streams are aligned in time, to ensure that the auditory stream matches the actions executed by the robot. We use the NAOqi and experimental NAOqi-Modularity frameworks provided by Aldebaran Robotics, which allow us to combine asynchronous data collection and data processing using filter chains in the humanoid robot Nao.

Experiments

We report here two experiments. Firstly, we present a simple classification experiment with the aim of demonstrating the learning and predictive capabilities that the proposed model can provide to artificial agents. In particular, we adopt the proposed framework for allowing a humanoid robot to learn the ego-noise that it is producing when performing a motor behaviour consisting of periodical horizontal head rotations (see Figure 3). Thus, we implement a classification experiment where the robot has to classify a behaviour it is executing in terms of velocity profile, by comparing the produced ego-noise to simulations of the ego-noise produced by imaginary executions of all the behaviours in the repertoire. In addition, we show how the model can deal with the situation when input information are missing, for example due to a damage in the system, resulting in the model still being able to classify, although with poorer performance.

Secondly, we describe an experiment on ego-noise attenuation with the aim of showing that the computational processes implemented by our framework resemble those

Figure 3: The robot behaviour executed during the recordings consisted in periodical rotations of the head on the yaw axis.
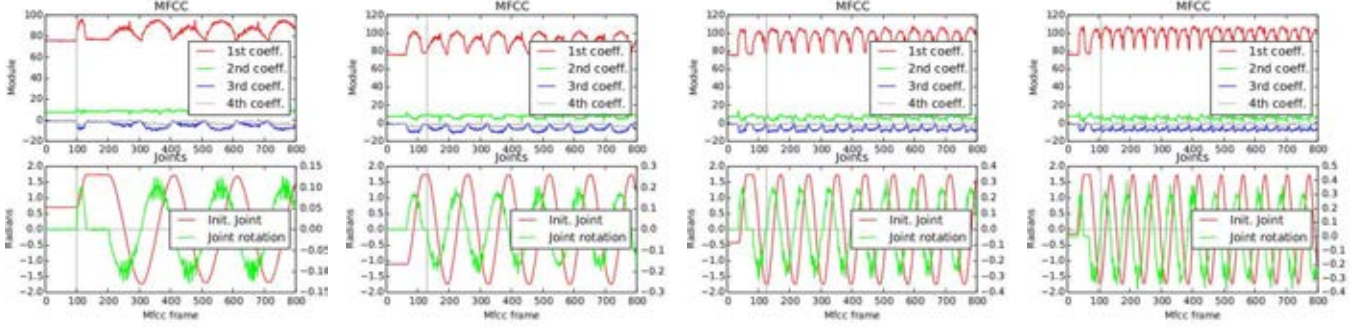
Figure 4: Example of trajectories of synchronised audio-motor data from the four velocity profiles. The upper plot shows the trajectories of the first 4-MFCC coefficients extracted from the single channel auditory signal recorded while executing the head rotations. The plot in the bottom shows the head yaw joint position (red line) and the head yaw rotation over $40ms$ (green line). The columns represent the different velocity profiles. From left to right: *slow*, *medium*, *fast* and *very fast*.

proposed by the behavioural studies mentioned in the introduction of this work, which would explain the mechanisms behind the sense of agency (Weiss et al., 2011; Blakemore et al., 2000a). In particular, we report an experiment on ego-noise attenuation based on sensorimotor predictions, where the quality of the attenuation is dependent on the degree of congruence vs. incongruence between predicted and actual sensory consequences of self-generated actions. In line with the behavioural studies reported above, we show that prediction errors generated by internal sensorimotor simulations are smaller when the proprioceptive and motor information are coherent with the events that are perceived from the external environment. We reported similar results in a different robotic experiment in the context of visuo-motor coordination (Schillaci et al., 2013).

Ego-noise classification

In the first experiment, we trained four different models with sensorimotor data gathered while executing a robot behaviour consisting of periodical horizontal head rotations with four different velocity profiles. We implemented the four velocity profiles using the original Aldebaran NAOqi controller with gradually increasing velocity thresholds, here named as *slow*, *medium*, *fast* and *very fast*. Figure 4 shows sample trajectories of aligned auditory and motor training data for each of the four velocity profiles. Training

of the models has been tested online and runs in real-time on an Aldebaran Nao v.5 robot. However, the classification results reported here are taken from models trained and tested offline. Sensorimotor data was gathered from the robot executing for ca. 200 seconds each of the four velocity profiles, resulting in 9449 training samples for the *slow* velocity profile, 9449 for the *medium*, 9459 for the *fast* and 9459 for the *very fast*. Each training sample consisted of the following sensorimotor information: S(t), encoding the MFCC features extracted from a single audio chunk (see Section "Ego-noise representation" for more details); S(t-1), encoding the initial position of the head yaw joint, that is the closest position in time to the first audio sample of the MFCC chunk; M(t-1), encoding the rotation of the head yaw joint over 40 ms, from S(t-1).

Four internal models have been trained with the different datasets (slow, medium, fast and very fast velocity profiles). Each internal model consisted of three maps (see Figure 2): a proprioceptive map, encoding a mono-dimensional feature space representing the initial head yaw joint position, that is $S(t-1)$; a motor map, encoding a mono-dimensional feature space representing the head yaw joint rotation, that is the motor command $M(t-1)$; an auditory map, encoding a 26-dimensional MFCC feature space representing the robot ego-noise. Each internal model encoded both the inverse and the forward models, as these are implemented by

the Hebbian tables containing the proper directional links, as explained in the previous section. Each SOM consisted of a 10x10 lattice of nodes, whose weights are randomly initialised and sampled from a Gaussian distribution $\mathcal{N}(0, 1)$. The weights of the Hebbian links connecting each pair of SOMs were all initialised to 0.

The classification task consisted in feeding the four internal models (slow, medium, fast and very fast) with test data samples gathered from the different datasets which stored sensorimotor data produced with each of the four velocity profiles, and in comparing the predicted auditory outcome with the actual one. Auditory chunks are classified as the velocity profile belonging to the forward model that produced the smallest prediction error (calculated as the Euclidean distance between the predicted and the observed MFCCs). Figure 5 illustrates the classification process using internal simulations.
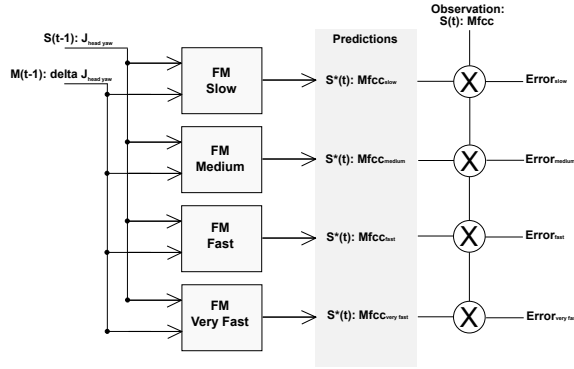

Figure 5: Diagram of the classification process.

Classification performance was measured for each trained model and on 5 different runs (thus, different test datasets). Table 1 shows the confusion matrix for the best run, when using only forward predictions with full input information.

| | | Classified as | | | |
	Slow	Medium	Fast	Very fast	# samples
	Slow	**94,00%**	5,50%	0,50%	0,00%	200
Executed	Medium	6,00%	**89,50%**	4,00%	0,50%	200
velocity	Fast	0,50%	14,00%	**85,5%**	0,00%	200
	Very Fast	4,00%	4,50%	7,50%	**84,00%**	200

Table 1: Confusion matrix showing the performance of the classification using only forward predictions.

We simulated a damage in the system, which was implemented as a lack of proprioceptive and motor information, during the predictive process. Internal simulations with partial inputs - in this case, only the auditory modality - were performed. The first step consisted in estimating a prediction of the motor command needed to generate the auditory outcome specified as input to the model, using an inverse prediction. The predicted motor command is fed into the corresponding forward model, which anticipates the sensory

outcome of the intended action. Table 2 shows the confusion matrix of the best of 5 classification runs, where we executed full internal simulations using only partial information as input. As expected, predictions estimated with missing proprioceptive inputs produced a degradation of the classification performance. However, the system is still able to classify correctly with at least 50% success rate.

| | | Classified as | | | |
	Slow	Medium	Fast	Very fast	# samples
	Slow	**85,5%**	11,50%	2,50%	0,50%	200
Executed	Medium	7,50%	**77,50%**	13,00%	2,00%	200
velocity	Fast	1,00%	16,00%	**80,0%**	3,00%	200
	Very Fast	1,50%	7,50%	37,00%	**54,00%**	200

Table 2: Confusion matrix showing the performance of the classification using both the inverse and forward predictions with missing input data (proprioceptive joints information).

Ego-noise attenuation as a cue for sense of agency

We performed a second experiment on ego-noise attenuation based on ego-noise predictions. In the experiment, we simulated that the robot is listening to an ego-noise signal (previously recorded from the robot itself) and, in the meanwhile, performing a motor behaviour. Along these movements, a forward model - trained with a periodical head rotation behaviour with slow velocity profile, as in the previous experiment - was used in executing sensorimotor simulations aimed at predicting the robot ego-noise generated by the current motor behaviour of the robot. We tested three conditions. In the first one, we simulated that the robot is executing a motor behaviour that is coherent with the observed ego-noise. In a second one, we simulated that the robot is not moving, thus holding the head in an initial position (applying a null motor command). In a third condition, we simulated that the robot is performing a periodical head rotation that is not aligned in time with the observed ego-noise. In each of the three conditions, we predicted the auditory outcomes of the movements by feeding the forward model with the joints information corresponding to the current motor behaviour. Thus, we subtracted from the original auditory information the one of the estimated noise.

Ego-noise suppression is performed in the log-filterbank energy domain. An inverse DCT (Discrete Cosine Transform) is applied to the 32-MFCC feature vectors representing the predicted and actual ego-noise chunks, producing two 32-D vectors (log-filterbank energies). Therefore, the vector representing the predicted ego-noise is subtracted from the one representing the actual ego-noise. In the event that the subtraction result in a dimension is negative, spectral flooring is applied, that is the attenuated signal is computed as the original one multiplied with a factor of 0.1.

Figure 6 qualitatively illustrates the results of the ego-noise attenuation. As evident from the plots, ego-noise attenuation is more pronounced when the input data fed to

the forward model is coherent with the auditory output (left graphs in the Figure - dark blue colour corresponds to total suppression of the ego-noise). The quality of the attenuation is worse, when there is incongruence between predicted and actual sensory consequences of self-generated actions, as in the case of the second and third condition. In particular, the second behaviour (head holding an initial position) generates a constant ego-noise prediction. The difference between the original and predicted ego-noise (bottom row, central column) is thus higher than in the case when the motor behaviour matches the observed ego-noise. The same effect is observed in the third condition, where the motor behaviour does not match the observed ego-noise.

In line with the studies reported in the introduction of this study, our experiment shows that prediction errors generated by sensorimotor simulations are smaller when the proprioceptive and motor information are coherent with the perceived ego-noise. Simply put, sensory attenuation is more pronounced when the robot is the owner of the action, as it has "a privileged access to internally generated efferent information during its own action" (Weiss et al., 2011), as simulated in the first condition of this experiment. The second and third condition simulated the situation where the robot is listening to another artificial agent performing a periodical horizontal head rotation behaviour, that *sounds exactly* as it would have been produced by the robot itself. However, the fact that the observed proprioceptive and motor information were incoherent with the observations of the ego-noise did constitute an element of surprise, as the forward model fed with such input data produced worse egonoise prediction than in the first condition of the experiment - as evident from the bigger prediction errors illustrated in Figure 6, bottom plots of the second and third columns.

Conclusion

We presented an implementation of a biologically inspired model for coding internal body representations that can generate predictions of auditory and motor experiences. The predictive capabilities provided by the models are tested in two experiments: a simple ego-noise classification task, where we also showed the capabilities of the model to produce predictions even in the absence of input modalities; an ego-noise suppression experiment, where we showed the effects in the ego-noise prediction, and thus suppression, performance of the input data to the forward model, when they are coherent or incoherent with the auditory observations. In line with the behavioural studies reported in the introduction of this paper, our experiment shows that prediction errors generated by sensorimotor simulations are smaller when the proprioceptive and motor information are coherent with the perceived ego-noise. Simply put, sensory attenuation is more pronounced when the robot is the owner of the action. When this is not the case, sensory attenuation is worse, as the incongruence of the proprioceptive and motor informa-

tion with the perceived ego-noise generates bigger prediction errors, which may constitute an element of surprise for the agent and allow it to distinguish between self-generated actions and those generated by other individuals.

In a separate study, we implemented a self-monitoring mechanism in a humanoid robot for the prediction and attenuation of visually detected consequences of self-generated actions. During a training phase consisting of a self-exploration behaviour, the system trained a forward model with motor data and visual data encoding movements detected from the robot camera. Consistently with the study reported here, sensory attenuation resulted to be more prominent in areas in the visual input where movements from the robot were expected. Instead, no attenuation was observed in areas of the visual input where the movements of an external object were detected. Again, this demonstrates that sensory attenuation processes can be adopted as a cue for distinguishing movements produced by external agents to those produced by the agent itself.

Therefore, we argue that equipping artificial agents with internal body representations and with the capability to perform sensorimotor predictions based on previous experience can represent a promising research direction towards the development of a sense of agency in artificial systems.

References

Blakemore, S.-J., Smith, J., Steek, R., Johnstone, E. C., and Frith, C. D. (2000a). The perception of self-produced sensory stimuli in patients with auditory hallucinations and passivity experiences: evidence for a breakdown in self-monitoring. *Psychological Medicine*, 30:1131–1139.

Blakemore, S. J., Wolpert, D., and Frith, C. (2000b). Why can't you tickle yourself? *NeuroReport*, 11(11):11–16.

Cang, J. and Feldheim, D. A. (2013). Developmental mechanisms of topographic map formation and alignment. *Annual Review of Neuroscience*, 36(1):51–77. PMID: 23642132.

Escobar, E., Hermosillo, J., and Lara, B. (2012). Self body mapping in mobile robots using vision and forward models. In *Electronics, Robotics and Automotive Mechanics Conference (CERMA)*, pages 72–77.

Holmes, N. and Spence, C. (2004). The body schema and multisensory representation(s) of peripersonal space. *Cognitive Processing*, 5(2):94–105.

Ince, G., Nakadai, K., Rodemann, T., Hasegawa, Y., Tsujino, H., and Imura, J. (2009). Ego noise suppression of a robot using template subtraction. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2009)*, pages 199–204.

Kaas, J. H. (1997). Topographic maps are fundamental to sensory processing. *Brain Research Bulletin*, 44(2):107 – 112.

Kaiser, A. (2014). *Internal visuomotor models for cognitive simulation processes*. PhD thesis, Bielefeld University.
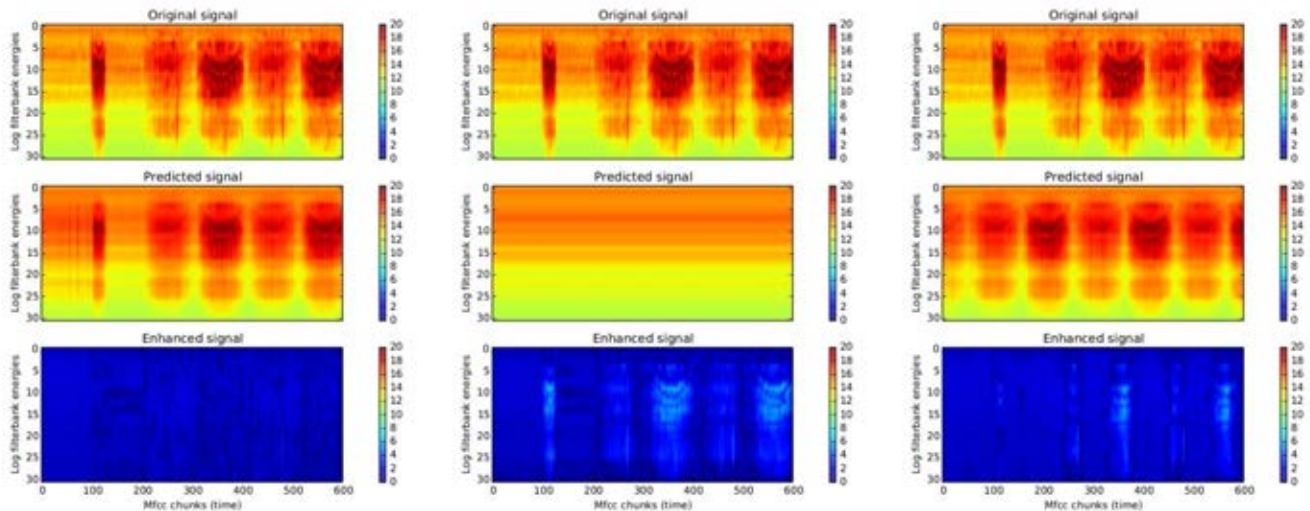
Figure 6: Ego-noise prediction and suppression tests. In the left graphs, input joint states and motor commands are coherent with the auditory outcome of the movements. In middle and right graphs, these inputs to the forward models are not coherent with the auditory information. In particular, in the middle graph the head is in an idle position (random joint position and motor command equals to 0); in the right graphs, joint and motor values follow periodical head movements shifted in time, compared to the actual auditory signal. In each of the three tests, the upper plots show the Mel log-filterbank energies extracted from the original auditory signal. The plots in the central row show predicted log-filterbank energies, where the input state and motor information vary according to the test. The bottom plots show the output of the ego-noise suppression (predicted signal subtracted from the original one).

Kajić, I., Schillaci, G., Bodiroža, S., and Hafner, V. V. (2014). Learning hand-eye coordination for a humanoid robot using soms. In *ACM/IEEE Int. Conf. on Human-robot Interaction (HRI2014)*, pages 192–193.

Knoblich, G. and Flach, R. (2001). Predicting the effects of actions: Interactions of perception and action. *Psychological Science*, 12(6):467–472.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69.

Loula, F., Prasad, S., Harber, K., and Shiffrar, M. (2005). Recognizing people from their movement. *Journal of Experimental Psychology: Human Perception and Performance*, 31(1):210.

Maravita, A. and Iriki, A. (2004). Tools for the body (schema). *Trends in Cognitive Sciences*, 8(2):79 – 86.

Miikkulainen, R. (1990). *DISCERN: A Distributed Artificial Neural Network Model Of Script Processing And Memory*. PhD thesis, University of California.

Möller, R. and Schenck, W. (2008). Bootstrapping cognition from behavior — a computerized thought experiment. *Cognitive Science*, 32(3):504–542.

Morse, A. F., Greef, J. D., Belpaeme, T., and Cangelosi, A. (2010). Epigenetic robotics architecture (ERA). *IEEE Transactions on Automnomous Mental Development*, 2(4):325–339.

Pitti, A., Mori, H., Kouzuma, S., and Kuniyoshi, Y. (2009). Contingency perception and agency measure in visuo-motor spiking neural networks. *IEEE Transactions on Autonomous Mental Development*, 1(1):86–97.

Sahidullah, M. and Saha, G. (2012). Design, analysis and experimental evaluation of block based transformation in {MFCC} computation for speaker recognition. *Speech Communication*, 54(4):543 – 565.

Schillaci, G., Hafner, V., and Lara, B. (2012). Coupled inverse-forward models for action execution leading to tool-use in a humanoid robot. In *ACM/IEEE Int. Conf. on Human-Robot Interaction (HRI2012)*, pages 231–232.

Schillaci, G., Hafner, V., and Lara, B. (2014). Online learning of visuo-motor coordination in a humanoid robot. a biologically inspired model. In *IEEE Int. Conf. on Development and Learning and Epigenetic Robotics*, pages 130–136.

Schillaci, G., Hafner, V., Lara, B., and Grosjean, M. (2013). Is that me? sensorimotor learning and self-other distinction in robotics. In *ACM/IEEE Int. Conf. on Human-Robot Interaction (HRI2013)*, pages 223–224.

Weiss, C., Herwig, A., and Schütz-Bosbach, S. (2011). The self in action effects: Selective attenuation of self-generated sounds. *Cognition*, 121(2):207–218.

Wilson, M. and Knoblich, G. (2005). The case for motor involvement in perceiving conspecifics. *Psychological Bulletin*, 131:460–473.

Wolpert, D. M., Ghahramani, Z., and Flanagan, J. R. (2001). Perspectives and problems in motor learning. *Trends in Cognitive Sciences*, 5(11):487 – 494.

Higher Order Cognition using Computers: Learning Abstract Concepts with Recursive Graph-based Self Organizing Maps

Peter J. Bentley[1, 2], Alexander Kurashov[1] and Soo Ling Lim[1, 2]

[1]Braintree Limited, London, United Kingdom
[2]Department of Computer Science, University College London, United Kingdom
p.bentley@cs.ucl.ac.uk

Abstract

Abstract concepts are rules about relationships such as identity or sameness. Instead of learning that specific objects belong to specific categories, the abstract concept of same/different applies to any objects that an organism might encounter, even if those objects have never been seen before. In this paper we investigate learning of abstract concepts by computer, in order to recognize same/different in novel data never seen before. To do so, we integrate recursive self-organizing maps with the data they are processing into a single graph to enable a brain-like self-adaptive learning system. We perform experiments on simple same/different datasets designed to resemble those used in animal experiments and then show an example of a practical application of same/different learning using the approach.

Introduction

Living organisms facing the challenges of survival must distinguish food from poison, friend from foe. In many simple organisms this may be a simple case of classification or categorical perception. For example, it is claimed that an earthworm can learn to avoid harmful stimuli by classifying the stimuli as harmful after some period of learning and avoiding them (Wilson et al., 2014), or a damselfish can recognize "faces" (Siebeck et al., 2015). But in more complex organisms, there is a need for a more complex form of learning: the recognition of abstract concepts.

Perhaps one of the most fundamental abstract concepts is same/different (S/D). Instead of learning that specific objects belong to specific categories, the abstract concept of S/D applies to any objects that the organisms might encounter, even if those objects have never been seen before. Four identical apples are the same, just as five identical cars are the same. A variety of species of mammals and birds are different, just as a variety of colors are different.

In theory, such a skill might have advantages – perhaps detecting differences in a group might indicate the presence of a predator, or differences within a nest might detect if eggs have been taken or replaced with those of another. However, experiments have shown that while organisms such as pigeons are capable of being taught the higher-level concepts of same/difference, this ability comes far more easily to animals with greater intelligence, such as chimpanzees, baboons, capuchin and rhesus monkeys (Katz et al., 2007).

Abstract concept learning is thus considered to form the basis of higher order cognition in humans (Katz et al., 2007). For many decades researchers have used the ability to judge same/different as a core theme in cognitive development, cognition, and comparative cognition (Goodman & Melinder, 2007; Mackintosh, 2000; Shettleworth, 2009; Thompson & Oden, 2000). The abstract concept of same/different is also thought to be necessary within mathematics, and learning language (Marcus et al., 1999; Piaget, 1970).

In computer science, the majority of research on learning focuses on classification or prediction, i.e., learning how to categorize or cluster specific types of data, or learning patterns and regularities within specific examples of data (Michie et al., 1994). Conventional machine learning typically does not study how abstract concepts can be learned.

In this paper we investigate learning of abstract concepts by computer, in order to recognize same/different in novel data never seen before. The study explores what type of information processing is required in order to perform this task, evaluating the algorithm with experiments similar to those performed with animals, and assesses whether same/different can truly be called higher order cognition. We also show an example of a more practical application of same/different learning using the approach.

The remainder of the paper is organized as follows. The next section describes existing work. The sections after that describe the method, experiments, and results. The final section provides our conclusions.

Background

Abstract Concept Learning

Abstract concepts are rules about relationships such as identity or sameness, and are considered to form the basis of much of our so-called higher order cognitive processing (Katz et al., 2007). Children develop cognition in stages and expand their abstract concept of sameness to include number, length, area, and volume (Piaget, 1970). In the laboratory, the abstract concept of sameness is studied in same/different (S/D) tasks where subjects view stimuli and then make one of two responses to indicate whether the stimuli are the same or different (Katz et al., 2007). The determination of abstract-concept learning is accurate performance with novel test stimuli, i.e., the subject learned an abstract rule that transcends the particular training stimuli. Such transfer performance

makes abstract-concept learning unique and different from other forms of concept learning (Katz et al., 2007).

It is important to differentiate abstract concepts from natural concepts. Natural concepts are categories of items which share specific features, such as cars, chairs, flowers, person, water, or trees. In contrast, abstract concepts do not involve learning specific stimulus features. Instead, they involve learning the relationships between items (Katz et al., 2007). Thus, abstract concepts involve relational learning as opposed to the item-specific learning of natural-concept learning. S/D experiments have been run on pigeons (Katz & Wright, 2006) and rhesus monkeys (Katz et al., 2002).

Self-Organizing Maps (SOMs)

This work uses graph-based SOMs in order to achieve abstract concept learning. The SOM is a biologically inspired brain-map model (Kohonen, 2013) which is often used for visualization of data to obtain a more abstract view (Kohonen, 1998). It is an automatic data-analysis method resembling the classical vector quantization, with the addition that more similar models will be associated with nodes that are closer in the grid, and less similar models will be situated gradually farther away in the grid (Kohonen, 2013).

SOMs have been used extensively for a variety of clustering, classification and visualization applications. For example. Merelo et al. (1994) used SOMs to develop a protein classification algorithm. Aly et al. (2008) and Lawrence et al. (1997) used SOMs for face recognition. Kamimoto (2005) used SOMs to evaluate the vibration of motor-operated electric tools. Teranishi (2009) used supervised SOM to estimate the bending rigidity of real bills using only the acoustic energy pattern. (Fatigued bills affect the daily operation of automated teller machines). Okada et al. (2009) used multiple SOMs for control of a visuo-motor system that consists of a redundant manipulator and multiple cameras in an unstructured environment. Tateyama et al. (2004) proposed a pre-teaching method for reinforcement learning using a SOM in order to increase the learning rate using a small amount of teaching data generated by a human expert. Amor and Rettinger (2005) developed a genetic algorithm that uses SOMs to enhance the search strategy and confront genetic drift. They found that representing the search history by the SOM provides visual insights into the state and course of evolution. De Buitleir et al. (2012) created an artificial life population where the agents have sufficient intelligence to discover patterns in data and to make survival decisions based on those patterns using diploid reproduction, Hebbian learning, and SOMs. Saunders and Gero (2001) used SOMs in their artificial life society of agents, enabling the agents to determine the novelty of new artifact without sampling the entire design space.

In this work we propose a recursive SOM, where the results from lower level SOMs are fed into higher level SOMs in order to enable abstract concept learning. Similar recursive SOMs have been proposed in the past, for example SOM^2 (Furukawa, 2009) and ASSOM (Kohonen, 1996), which use lower level SOMs to find the building blocks used by higher level SOMs, when visualizing similar data. In this work we propose a novel approach whereby datatype-independent interpretations or summaries of many distinct and different lower level SOM results are fed as input into parent SOMs.

In our work we also make extensive use of graph databases. Graph databases use graph structures for semantic queries with nodes, edges and properties to represent and store data (Angles & Gutierrez, 2008). SOMs have been used with semantic networks in the past with good results (Allinson et al., 2001) but in this work we also introduce another novel design feature: the SOMs are integrated into the network they are processing. By representing both data and the SOM in the same graph-based network the system becomes a dynamic "brain-like" network that integrates data and learning in the same structure, automatically reconfiguring itself when it needs to learn new concepts, and providing complete provenance for all nodes.

Method

Abstract concepts are not absolute concepts. In real life, the abstract concept of same/different means "degree of difference". Objects with few or no apparent differences can be called "same". Objects with many distinguishing features can be called "different". Organisms may have a different concept of same/different depending on their experiences. For example, if we see many different species, then we learn to detect the differences between species – a flock of birds is different from a herd of buffalo. If we see many animals from one species, then we learn to detect the difference between individual animals – each buffalo is different from its companions.

Therefore, to achieve the learning of same/different, it is not enough to learn the concept from one set of data. One must learn from multiple sets of data that some things within each set are the same and others within each set are different, and the exact meaning of the concept same/different will depend on the data sets presented. To achieve this, a recursive learning approach is needed:

1. For each data set: the data is clustered into groups containing similar items.
2. The clustered data sets are then clustered according to overall datatype-independent features of each set (in this case the number of clusters).
3. (Optional) Step 2 is repeated for even higher-level concept learning.

After performing second-order clustering, the abstract concepts of same and difference can be constructed. Those clusters containing data sets with fewer clusters, have items more similar to each other. Clusters containing datasets with more clusters, have items more different to each other.

By feeding the results of the learning algorithm back into itself in this way, the higher-order, abstract concepts can be learned (and indeed ever-higher level, more abstract concepts can be learned by recursing further). Uniquely to this work, the input data, the SOM grids, and the resulting clusters derived from the SOMs, are all stored within the same graph structure. In order to achieve this while making every data item traceable from high-level concept to input vector, this work uses a graph database to store all data, learned models, and concepts. Unlike traditional relational databases, graph databases enable *provenance*. The provenance of a data item is the lineage of that data item, describing what it is and how it came to be. The provenance about a data item includes details about processes and input data used to create the item. This is

of great benefit when constructing new networks of concepts and abstract concepts, as it becomes possible to trace exactly which data resulted in which abstract concepts. Graph databases also enable fast traversals of large-scale network-based data - not something that traditional relational databases support well.

These approaches are also chosen because of their similarity to neural systems and their power and scalability to large datasets and large numbers of features per data item (Kohonen, 2013). The system is implemented in Java and the Neo4j graph database[1] is used, also chosen for its ability to scale and support high-speed queries.

In more detail, the method works as follows.

Build Dataset Graph

Each dataset is encoded as a graph, see Figure 1. A new Vector node is created for each data item in the set, with data attributes encoded as a list of numeric properties in each Vector. The Dataset node links to each Vector node (with a HAS_MEMBER link). The Dataset node also links to a Datatype node, containing a type variable with unique value for each dataset (with HAS_TYPE link).
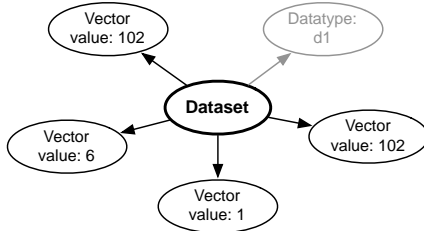
Figure 1. Each dataset is stored in a graph database (HAS_MEMBER links are in black, HAS_TYPE in grey).
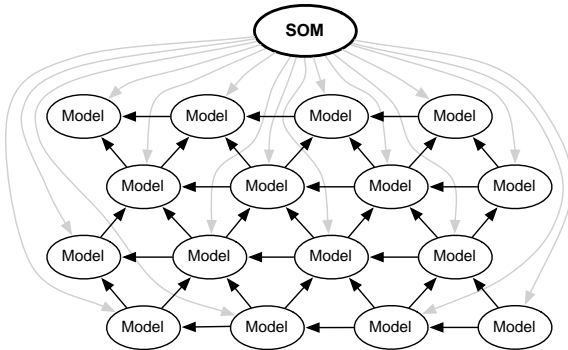
Figure 2. 4x4 SOM graph. HAS_MODEL links are shown in grey. NEIGHBOUR links are shown in black.

Build SOM graph

An SOM graph is created, see Figure 2. The SOM is encoded as an SOM node connected to a grid of $n{\times}n$ set of Model nodes (using a HAS_MODEL link), that are connected to their neighbors (using a NEIGHBOUR link). Each Model node

[1] http://neo4j.com/

contains a list of properties corresponding to the attributes of the data. Each HAS_MODEL link has integer properties "row" and "col", which represent the position of the Model within the grid.

First Order Learning

To classify the Dataset using the graph-based SOM, Algorithm 1 is used. Where no existing Models exist for a dataset of this type, each SOM is initialized by setting half of the attributes in each Model to *r/(N-1)* and the other half to *c/(N-1)*, where *(r, c)* denote the rectangular coordinate of the Model, and *N* is the size of the grid. This common method of initialization was chosen after preliminary experiments on standard datasets (e.g., Iris, Car, Wine, and Zoo from the UCI Machine Learning repository[2]) showed it was more effective than other alternatives. It also has the advantage of being deterministic and fast. Where a dataset of this type has been seen before, there will exist Models linked to the Datatype node; these are used to initialize the SOM to make use of past learning and improve speed. (Preliminary experiments showed that the reuse of previously learned Models can reduce the number of iterations needed for the SOM to converge.)
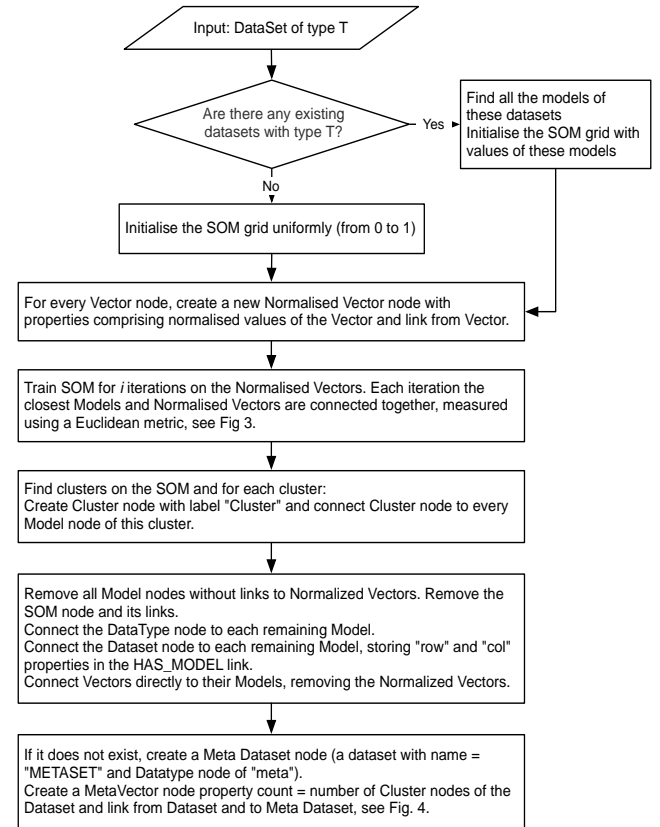
Algorithm 1. Learning and abstracting concepts from a dataset using the graph-based SOM.

[2] http://archive.ics.uci.edu/ml/
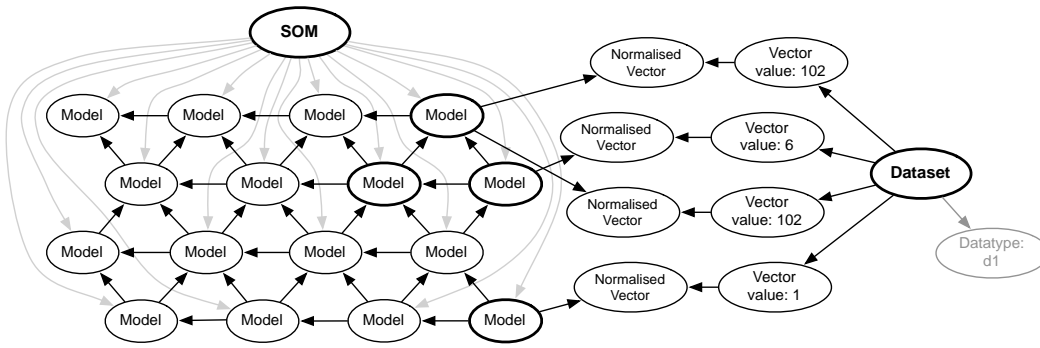
Figure 3. A graph-based SOM is trained on normalized vectors from the Dataset.

The SOM training procedure is taken from (Kohonen, 2013). On every iteration, each Model value is updated:

$$m'_i = \frac{\sum_j n_j h_{ij} x_j}{\sum_j n_j h_{ij}}, \text{ where } h_{ij} = \frac{1}{gd(i,j)+1} \tag{1}$$

and where: x_j denotes the mean of the vectors that are closest (according to the Euclidean metric) to the Model j; n_j denotes the number of those vectors; $gd(i, j)$ denotes the grid distance between Models i and j; h_{ij} is a neighbourhood function (Kohonen, 2013).

The grid distance gd between Models is the distance between the cubic coordinates (ax, ay, az) and (bx, by, bz) of each Model M and is defined in Eqn. 2.

$$gd(M_{ax,ay,az}, M_{bx,by,bz}) = \frac{|ax-bx|+|ay-by|+|az-bz|}{2} \tag{2}$$

The cubic coordinate of a Model (x, y, z) is a transformation of its rectangular coordinate (r, c) defined in Eqn. 3.

$$(x, y, z) = (c - \frac{(r-(r \text{ XOR } 1)}{2}, -x-z, r) \tag{3}$$

This method to compute the grid distance is chosen to optimize performance of a heavily used neighborhood function calculation.

After every Model is updated, the mean energy of the iteration is calculated (Eqn. 4).

$$E^* = \frac{\sum_i d(m_i, m'_i)}{N^2} \tag{4}$$

where m_i denotes the Model value before update; m_i' denotes the Model value after update; $d(x, y)$ denotes the Euclidean distance between vectors x and y; N denotes the size of the grid, and N^2 thus denotes the number of Models.

Each SOM is trained for a maximum of 50 iterations or until the mean energy E^* is less than 10^{-8}. The choice of neighborhood function and other values were also chosen after performing preliminary experiments on standard UCI datasets. Figure 3 illustrates the graph during the SOM learning phase. After learning is complete, the clusters are identified (see Algorithm 2) and Cluster nodes are linked to Models. Any unused Models and all normalized vectors are then removed from the SOM graph. This helps reduce the space taken for learning, while retaining useful Models in case the same type of data is presented again in the future. A MetaVector is then created, with a value that equals the number of clusters found. Figure 4 illustrates the graph after learning is complete and the SOM has been removed.
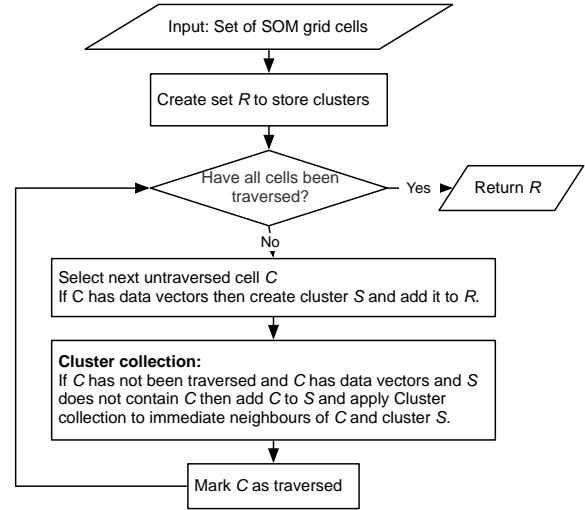
Algorithm 2. Finding clusters within the graph-based SOM.
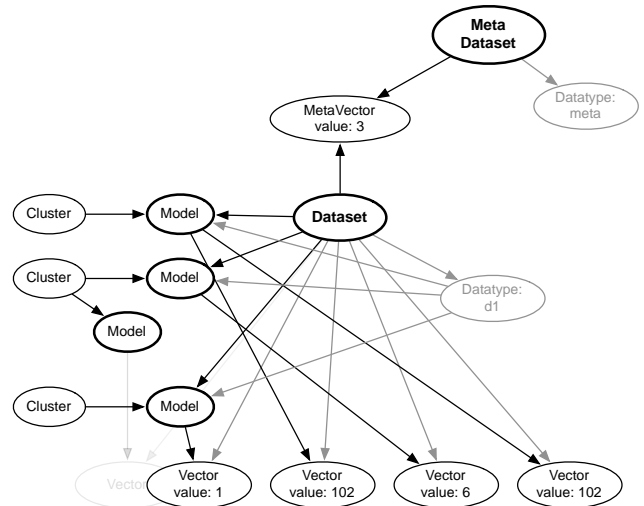
Figure 4. Similar Models are grouped into Clusters, the SOM and its unused Models are discarded, the normalized vectors are removed, and a MetaVector is created, summarizing the number of clusters in the Dataset, and added to the Meta Dataset. (A vector connected to the third Model is shown in pale grey for clarity.)
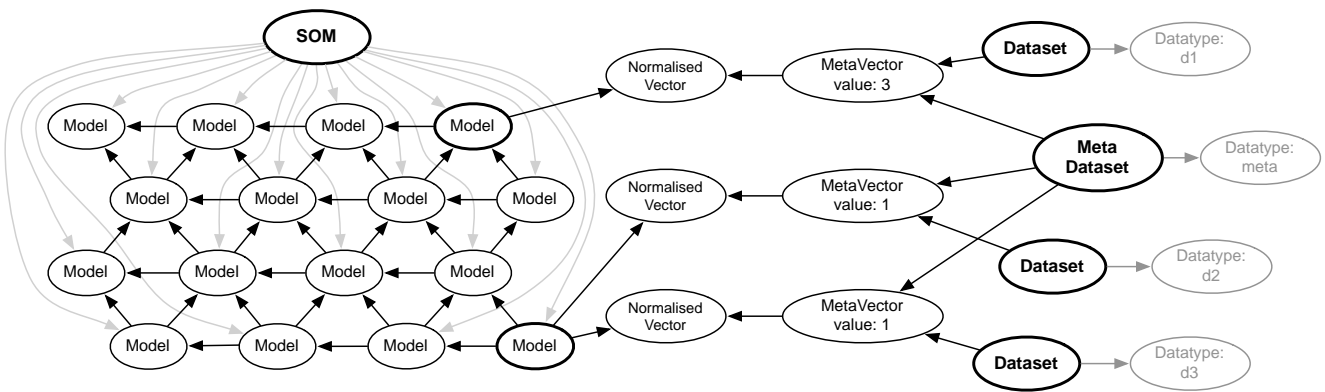
Figure 5. A new SOM is created and trained on the normalized MetaVectors from multiple different Datasets, which may comprise different types of data. (Graphs linked to each Dataset are not shown for clarity.)
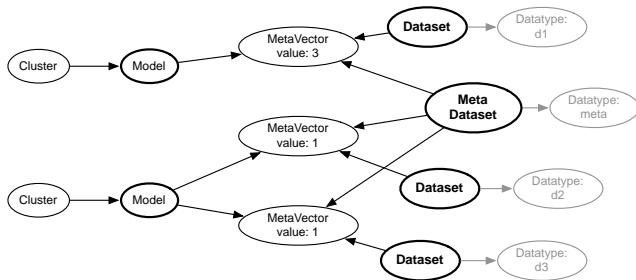
Figure 6. Similar Models in the SOM are linked to Clusters, the SOM and its unused Models are discarded, and the Normalized MetaVectors are removed as before. (New links between Models and Meta Datasets and Datatypes are not shown for clarity.)

Abstract Concept Learning

Each time First Order Learning is performed on a new Dataset, a new MetaVector node (linked to its corresponding Dataset) is created and linked to the Meta Dataset. Abstract Concept Learning is then performed by creating a new SOM to train on the set of MetaVectors. To perform Abstract Concept Learning, Algorithm 1 is used again on this Meta Dataset, where T = "meta", and Vectors are MetaVectors. Figure 5 illustrates the learning phase as the SOM is trained on the Normalised MetaVectors. Figure 6 illustrates the graph after learning is complete and the SOM has been removed. The resulting emergent clusters represent abstract concepts created – in Figure 6, one cluster represents the concept "same" as it points to Datasets containing only single clusters; the other concept represents "different" as it points to a Dataset containing multiple clusters (the MetaVector value is 3, representing 3 clusters). The final step of Algorithm 1 is optional (its execution would produce an even higher level of abstraction). If desired, such recursion could continue until everything is grouped into a single cluster.

Experiments

Experiment 1

The first experiment investigates whether the system can learn the higher order concept of "sameness" when presented with simple data similar to the simple symbols shown to animals during S/D experiments, see Table 1 (Katz et al., 2007). Table 2 shows the six simple sets of data, each set containing no value that appears in an earlier set. These sets were presented to the system in turn, allowing learning to complete before the next set was presented. SOMs were 4x4 in size with settings as described earlier.

Results 1

The results were fascinating. As each new dataset was presented, the first level SOM clustered the data, the clusters were identified, and the MetaVectors summarizing the clusters were then clustered to produce abstract concepts. Initially the first two datasets were clustered together – the system had correctly identified that both A and B contain items that are the Same. After dataset C was presented, the system created a new cluster containing C, representing the abstract concept Different. Datasets D and E were also placed into the Different cluster, see Figure 7 (Left). However, after the final dataset F was presented, the system had gained enough insight to reorganize the datasets, finally clustering A and B together (Same), C and D (Slightly Different), and E and F together (Very Different), see Figure 7 (Right). This seems a valid view of the data given that datasets E and F contain 10 and 9 unique values whereas C and D only comprise 4 values, of which only 1 or 2 are different.

– – – – – – – –	Same Display
+ U + + + + + +	Different Display

Table 1. Typical animal experiment data (Katz et al., 2007).

Data set	Values	S/D
A	33, 33, 33, 33	S
B	7.1, 7.1, 7.1, 7.1	S
C	102, 6, 102, 102	D
D	31, 1.5, -3, 31	D
E	1, 2, 3, 4, 5, 6, 7, 8, 9, 10	D
F	1.0001, 1.0002, 1.0003, 1.0004, 1.0005, 1.0006, 1.0007, 1.0008, 1.0009	D

Table 2. Input data sets for Experiment 1.
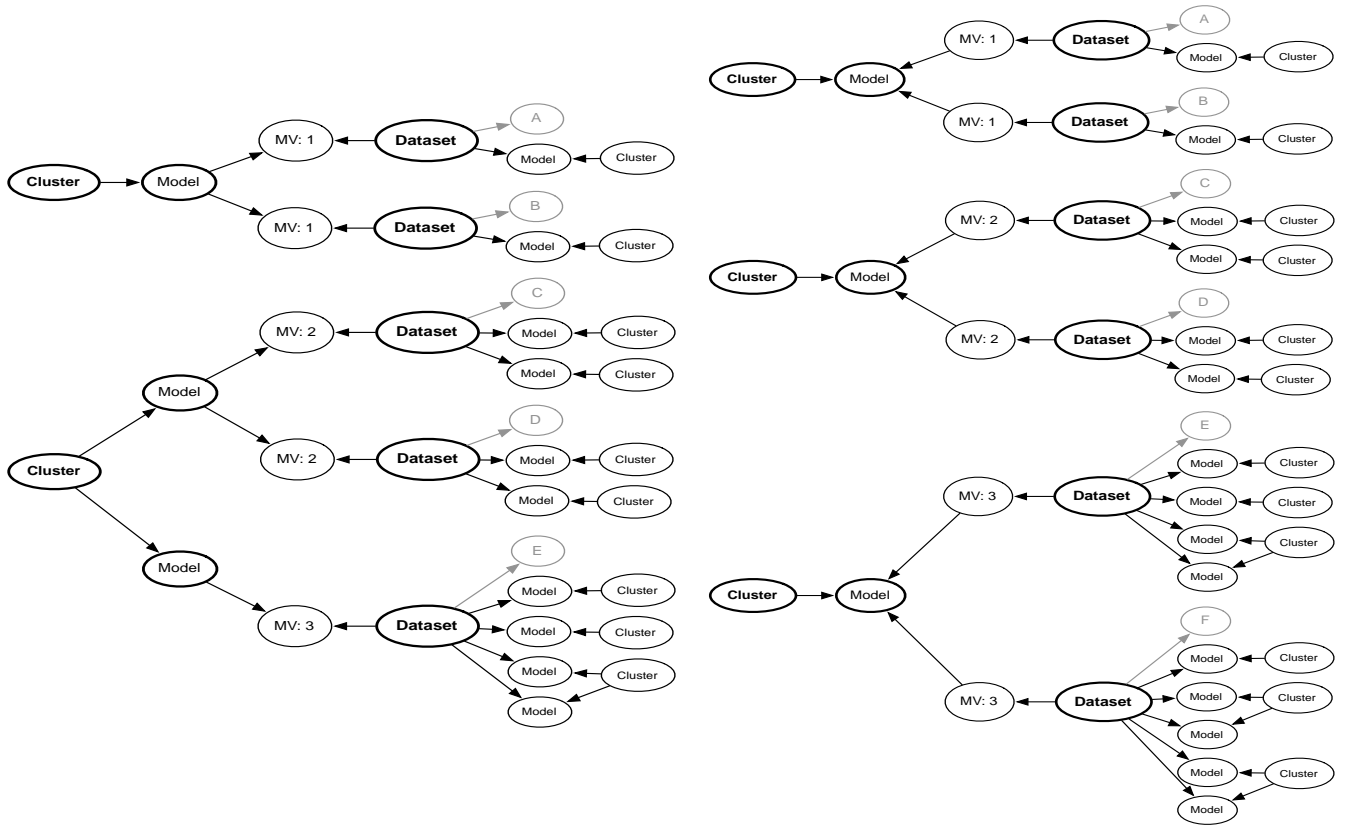
Figure 7. Left: the graph after presenting the first 5 datasets A to E. MetaVectors are shown as MV. The second-order SOM has discovered that there are two types of dataset: A,B (corresponding to Same) and C,D,E (corresponding to Different). Right: the graph after presenting the final dataset F. With more experience, the second order SOM now better distinguishes between the different types of dataset, by identifying three kinds: A,B (corresponding to Same), C,D (corresponding to Slightly Different) and E,F (corresponding to Very Different).

Experiment 2: Visual data

In many animal S/D experiments, instead of showing simple symbols, actual images of objects such as cars or flowers are shown (Katz & Wright, 2006). In the second experiment we take inspiration from these experiments and allow the system to "watch" an animation in which an event occurs after a specific period of time. Here we wish to determine if the system can understand when the frames of the animation are mostly the Same, discover the point in time when they become Different, and understand that new, unseen frames after the event may also be considered the Same once again. This is a more practical application of the ability to detect S/D and could, for example, be used to detect some anomalous event occurring in a video feed from a security camera. It is a more challenging task as the data presented to each SOM is considerably larger compared to Experiment 1. To enable the input to be carefully controlled, in Experiment 2 we use an 8x8 10-frame color animation of the video game character Mario, see Figure 9. Each frame was converted to a vector of 8x8x3 = 192 RGB values. An 8x8 SOM grid was used, and all other settings were the same as in Experiment 1. Three different scenarios were presented to the system.

Scenario 1: Jump after 10 frames: 100 frames of animation are presented, 5 at a time (e.g., 2-3-4-2-6, 7-2-3-4-2). The first

10 frames are Mario walking. The next 5 frames depict Mario jumping out of the picture. The remaining 85 are black.

Scenario 2: Jump after 50 frames: 100 frames of animation are presented, 5 at a time. After 50 frames of walking, the next 5 depict Mario jumping, and the remaining 45 are black.

Scenario 3: Jump after 90 frames: 100 frames of animation are presented, 5 at a time. After 90 frames of walking, the next 5 depict Mario jumping, and the final 5 are black.

Results 2

Figure 8 shows a summary of the clusters for each scenario. In the fist scenario the frames are considered Same (one meta cluster) until the Event on chunk 3 (frames 10-15) when Mario jumps. Here the first SOM finds two clusters, resulting in the higher level SOM forming a new meta cluster corresponding to the abstract concept Different. Following the Event, new identical black frames are presented, which are correctly classified as Same. This pattern occurs in the other two scenarios, except that during the Event the first level SOM finds three clusters. In all cases, the system learns that most of animation frames in each chunk can be considered the Same, but during each Event, one chunk of frames comprises Different frames.
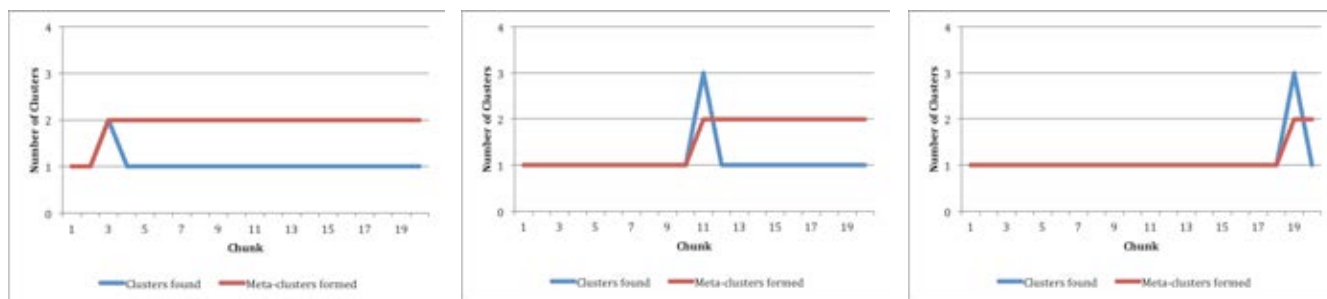
Figure 8. The number of clusters found by the first level SOM for each 5-frame chunk and meta clusters found so far by the second level SOM. The Event (Mario jumping out of shot) occurs during chunk 3 for Scenario 1, chunk 11 for Scenario 2 and chunk 19 for Scenario 3. On each Event the system learns and remembers that there are two classes of animation frames: Same and Different.
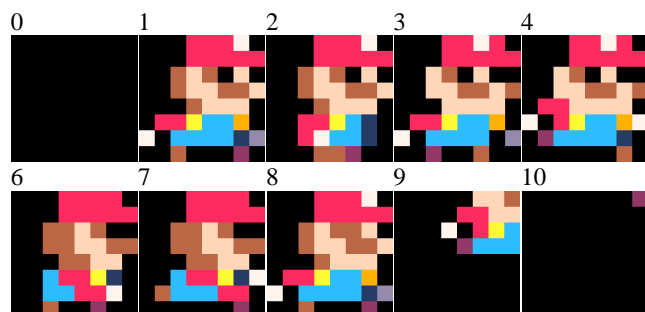
Figure 9. The 10 different frames of the Mario animation, comprising the walking sequence of frames: 2-3-4-2-6-7-2-3…, blinking sequence: 1-8-1-8-1…, and jumping sequence: 4-9-10-0-0-0…

Figure 10. (A) examples of a set of stored Models corresponding to the learned abstract concept Different. (B) and (C) show two examples of stored Models corresponding to the learned abstract concept Same, where (B) is a "hazily remembered Mario" and (C) comprises black frames.

A further test was performed by introducing the blinking Mario chunk (frames 1-8-1-8-1) at various times. This chunk of new frames could be regarded as Same (since each consecutive frames is different by a single pixel) or as Different (there are two clear groups of different frames: 1 and 8). Fascinatingly, the system seemed ambivalent towards this chunk of animation. It generally regarded the chunk as Same if it was presented after seeing many other Same chunks. It regarded the chunk as Different if it was presented after seeing the Event, which it considers Different.

Discussion

Abstract concept learning is considered to form the basis of higher order cognition in humans. This work has shown that in a very real sense, the notion of the abstract concept Same/Different is indeed higher order. For an SOM to discover this datatype-independent abstract concept, the output from one SOM must be fed into a higher-level SOM, which must learn about the general, datatype-independent features (in our case, the number of clusters) found by the first. This learning of datatype-independent features is an important requirement. Unlike previous work on recursive SOMs which build improved views of the same kind of data by using lower level SOM clusters as building blocks, here we show that it is necessary for the higher level SOM to learn

about sets of overall interpretations or summaries of the datasets derived from the lower-level SOMs. By doing so, the system is able to accept radically and completely different datasets that share nothing in common, and still determine whether each dataset comprises items that should be considered largely Same or Different. With enough examples, the system can also start to differentiate further, and find Same, Slightly Different and Very Different. It should be stressed that these abstract concepts are automatically generated and they will change depending on what kinds of data are presented. Like the S/D abstract concept of living creatures, the system here only understands Same and Difference in terms of its experience of different data sets, not in terms of any absolute comparisons of values.

The use of a graph database for this approach also enables the system to explain its understanding. We are able to query the database and ask it for examples of Same or Different that it has experienced. Even when the original datasets have been removed, the stored SOM Models are able to provide a "hazy memory" of its notion of each abstract concept, see Figure 10. The connections from each Model via MetaVectors and Models to Dataset types enables the tracking of the provenance of each Model – it is possible to know which dataset resulted in each abstract concept or which "hazy memory".

Conclusion

In this work we have presented a novel recursive SOM, where a datatype-independent summary of the output from lower-level SOMs that have been applied to different datasets is fed into a higher-level SOM in order to learn the abstract concepts of Same/Difference. The implementation exploits graph-based computing, with data, SOM grid, learned clusters and all concepts represented in the same graph database. This provides the advantage of provenance for all nodes, enabling details about processes and input data used to create the item to be found highly efficiently. The graph representation reorganizes itself during and after learning, adding new concepts as they are discovered, removing nodes when they are no longer needed, and reusing stored nodes to improve efficiency of learning. The use of the graph database also enables this approach to scale.

The method was tested on simple same/different datasets designed to resemble those used in animal experiments and then a more practical application of same/different learning was investigated – finding anomalous frames within a short animation. In all cases the system demonstrated a clear ability to learn the datatype-independent abstract concepts of Same/Different correctly, and this "skill" was refined and improved as new data was presented.

There are many potential applications for this system, where learning of Same/Different is non-trivial. Examples include the identification of similar profiles in large databases, anomalous events in video streams, or the identification of other higher-level concepts such as homographs and synonyms. Future work will explore some of these ideas.

References

Allinson, N., Yin, H., Allinson, L., & Slack, J. (2001). *Advances in self-organising maps*. Springer, London.

Aly, S., Sagheer, A., Tsuruta, N., & Taniguchi, R.-i. (2008). Face recognition across illumination. *Artificial Life and Robotics, 12*(1-2): 33-37.

Amor, H. B., & Rettinger, A. (2005). Intelligent exploration for genetic algorithms: Using self-organizing maps in evolutionary computation. *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, pages 1531-1538.

Angles, R., & Gutierrez, C. (2008). Survey of graph database models. *ACM Computing Surveys, 40*(1): 1-39.

de Buitléir, A., Russell, M., & Daly, M. (2012). Wains: A pattern-seeking artificial life species. *Artificial Life, 18*(4): 399-423.

Furukawa, T. (2009). SOM of SOMs. *Neural Networks, 22*(4): 463-478.

Goodman, G. S., & Melinder, A. (2007). Child witness research and forensic interviews of young children: A review. *Legal and Criminological Psychology, 12*(1): 1-19.

Kamimoto, N., Yamada, Y., Kitamura, M., & Nishikawa, K. (2005). Evaluation of vibration in many positions by SOM. *Artificial Life and Robotics, 9*(1): 7-11.

Katz, J. S., & Wright, A. A. (2006). Same/different abstract-concept learning by pigeons. *Journal of Experimental Psychology: Animal Behavior Processes, 32*(1): 80-86.

Katz, J. S., Wright, A. A., & Bachevalier, J. (2002). Mechanisms of same-different abstract-concept learning by rhesus monkeys (Macaca mulatta). *Journal of Experimental Psychology: Animal Behavior Processes, 28*(4): 358-368.

Katz, J. S., Wright, A. A., & Bodily, K. D. (2007). Issues in the comparative cognition of abstract-concept learning. *Comparative Cognition & Behavior Reviews, 2,* 79-92.

Kohonen, T. (1996). Emergence of invariant-feature detectors in the adaptive-subspace self-organizing map. *Biological Cybernetics, 75*(4): 281-291.

Kohonen, T. (1998). The self-organizing map. *Neurocomputing, 21*(1): 1-6.

Kohonen, T. (2013). Essentials of the self-organizing map. *Neural Networks, 37,* 52-65.

Lawrence, S., Giles, C. L., Tsoi, A. C., & Back, A. D. (1997). Face recognition: A convolutional neural-network approach. *IEEE Transactions on Neural Networks, 8*(1): 98-113.

Mackintosh, N. J. (2000). Abstraction and discrimination. In C. Heyes & L. Huber (Eds.), *The Evolution of Cognition* (pp. 123-141). Cambridge, MA, US: The MIT Press.

Marcus, G. F., Vijayan, S., Rao, S. B., & Vishton, P. M. (1999). Rule learning by seven-month-old infants. *Science, 283*(5398): 77-80.

Merelo, J. J., Andrade, M. A., Prieto, A., & Morán, F. (1994). Proteinotopic feature maps. *Neurocomputing, 6*(4): 443-454.

Michie, D., Spiegelhalter, D. J., & Taylor, C. C. (1994). *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, Upper Saddle River, NJ, USA.

Okada, N., Qiu, J., Nakamura, K., & Kondo, E. (2009). Multiple self-organizing maps for a visuo-motor system that uses multiple cameras with different fields of view. *Artificial Life and Robotics, 14*(2): 114-117.

Piaget, J. (1970). *Science of Education and the Psychology of the Child. Trans. D. Coltman*. Orion, New York.

Saunders, R., & Gero, J. S. (2001). A curious design agent. *CAADRIA (The Association for Computer-Aided Architectural Design Research in Asia)*, pages 345-350.

Shettleworth, S. J. (2009). *Cognition, Evolution, and Behavior*. Oxford University Press, USA.

Siebeck, U. E., Parker, A. N., Franz, M. O., & Wallis, G. M. (2015). Face discrimination in fish. *Behaviour*, pages 1.

Tateyama, T., Kawata, S., & Oguchi, T. (2004). A teaching method using a self-organizing map for reinforcement learning. *Artificial Life and Robotics, 7*(4): 193-197.

Teranishi, M., Omatu, S., & Kosaka, T. (2009). Continuous fatigue level estimation for the classification of fatigued bills based on an acoustic signal feature by a supervised SOM. *Artificial Life and Robotics, 13*(2): 547-550.

Thompson, R. K., & Oden, D. L. (2000). Categorical perception and conceptual judgments by nonhuman primates: The paleological monkey and the analogical ape. *Cognitive Science, 24*(3): 363-396.

Wilson, W. J., Ferrara, N. C., Blaker, A. L., & Giddings, C. E. (2014). Escape and avoidance learning in the earthworm Eisenia hortensis. *PeerJ 2:e250,* https://doi.org/10.7717/peerj.250.

How long did it last? Memorizing interval timings in a simple robotic task

Julien Hubert and Takashi Ikegami

The University of Tokyo, Ikegami Laboratory, Tokyo, Japan
{jhubert,ikeg}@sacral.c.u-tokyo.ac.jp

Introduction

Time perception is the capacity to sense the passing of time, but in most living creatures it also involves memorizing how much time passed, and eventually acting when it reaches a specific amount. The later is referred as interval timing. This capacity allows animals to detect temporally repeating events in their environment, avoid them if necessary, or exploit them if beneficial(Saigusa et al., 2008).

While the research in animals has focused on interval timing (Connor, 1985; Durstewitz, 2003), research in artificial life has limited itself to time perception (Maniadakis et al., 2014; Trianni, 2008). Indeed, alife models rely on the intrinsic temporal properties of neural networks to encode the passing of time and, therefore, cannot estimate how much time passed since the onset of a stimulus. Our work attempts to make one step closer to interval timing by designing an agent which must learn the duration of a stimulus, but also replay it later on.

Experimental Setup

Our task takes place inside an unbounded arena with two differently colored circular areas: the Stimulus Area (SA) where a stimulus is played, and the Replay Area (RA) where the agent must replay the duration of this stimulus. At the start of a trial, the agent is placed in SA with a random location and a random orientation. After 1s, a stimulus is played for a duration of either 2s or 4s. The agent is free of its movement and can leave SA at any time while the stimulus is played. Once it leaves SA, the stimulus is stopped (if still playing), SA disappears, and RA becomes visible. The agent must move to RA and remain on it for the duration of the initial stimulus. The agent has 30s to complete the task before the experiment is terminated. Because the goal and the stimulus are never shown simultaneously, the agent must maintain the duration in memory until it reaches the goal.

The agent is a simulated e-puck robot equipped with floor sensors, to detect the color of the two areas, and a compass indicating the direction of RA when visible. The controller of the agent is a CTRNN with 30 neurons (determined experimentally), including three inputs and two outputs (Beer,

1995). The first two inputs encode the floor sensors (one input for each area), and the last one receives the compass value. The outputs encode the speed of the left and right motors.

The parameters of the CTRNN are tuned using a genetic algorithm (GA) with tournament selection, one-point crossover and mutation (Holland, 1975). The fitness function is given by

$$fitness = \frac{tosa}{stimdur + 1} + \left(\frac{tora}{stimdur}\right)^{sign(stimdur - tora)}$$

(1)

where $tosa$ and $tora$ are respectively the time the agent spent on the areas SA and RA, and $stimdur$ is the duration of the initial stimulation. The first term promotes the capacity to listen to the complete stimulus before leaving SA. It is artificially limited to a maximum of 1. The second term increases linearly the fitness while the agent remains on RA less than $stimdur$, but decreases it afterward.

The simulation is physics based and accurately encode time with a timestep of 0.01s. Uniform noise is added to the position and orientation of the agent, but also on the duration of the initial stimulus ($\pm 5\%$ uniform noise). The distance between SA and RA is fixed, but the random initial position of the agent prevents the evolution to exploit it. The initial activation values of the neurons in the CTRNN are drawn from a uniform random distribution with a maximum value of 0.1.

Results

Behavior

Six evolutionary runs of 20.000 generations with a population of 100 individuals provided two successful CTRNN for this task (fitness > 1.95). While their dynamics are slightly different, the evolved strategies are similar.

Figure 1 shows the position of the two agents during one typical run for both durations. Despite individual differences, both agents remain in SA until the end of the stimulus, then exit the area to move toward RA. In RA, the agents set up their passage to last the duration of the stimulus.
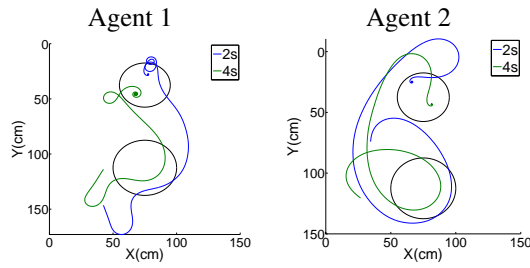
Figure 1: Positions of two agents for durations 2s and 4s. The upper circle is SA, and the lower one is RA.

To determine if the memory can be forgotten, we let the experiment run for 1000s and record the behavior of each agents. We observed that the agents return to RA quickly after leaving it. The duration of each stay within RA is shown in figure 2. For both agents, we can observe that the initial trial is successful, i.e. each agent remain on RA for the expected duration. Later on, the duration on RA increases progressively to stabilize at a fixed value (around 5s for agent 1, 4.5s for agent 2). This indicates two properties of the evolved agents. First, the memory is not fixed and disappears progressively with the time spent exploring the environment. Second, each network possesses a natural tendency to remain on RA for a specific duration. Interestingly, this duration is none of the ones we evolved the agents with.
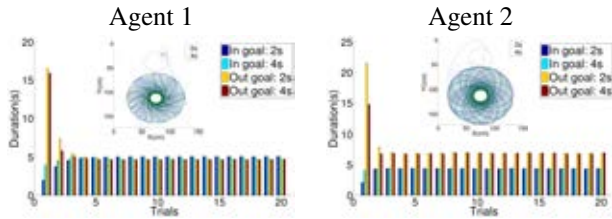
Figure 2: Time spent in and out of RA for each agent when the duration of the experiment is extended to 1000s. The inset shows the trajectory of the robot during the experiment.

Neural Dynamics

In order to get a better understanding of how the CTRNN solves the task, we computed the principal component analysis (PCA) on the activation of the neurons. The data compiled to compute the PCA is from 50 trials with 2s and 50 trials with 4s. Figure 3 shows the two strongest principal components (the 3rd one does not change the trajectory, and the others contain little information). In both agents, the dynamics while in SA for any duration remains the same. The dynamics of the networks for both durations diverge when the stimulus disappears, and move toward a similar region in the dynamical landscape. Between the end of the stimulus, and the end of the experiment, the state of the network

moves toward the final attractor which is the end of the trial. The important aspect of this analysis is that it does not show any fixed attractor connected to the memory of the duration. The memory is implemented within the transition toward the final attractor.
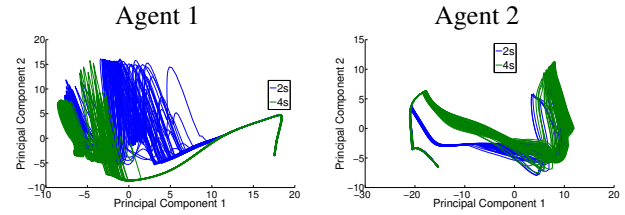
Figure 3: PCA of the neural activity of both agents.

Conclusion

The two agents presented in this work are capable of evaluating and memorizing the duration of 2s and 4s stimuli, move in their environment and replay this memory when needed. Our analyses determined that the agents are not implementing a stable attractor for one duration combined with a natural tendency for the other one. The memory of the duration is encoded through the transient dynamics of the network, which allows the network to forget what it learned previously. From our observations, it also allows the network to memorize the small variations of the duration due to the noise, indicating it memorizes the duration of the stimuli, and not just a cue indicating 2s or 4s.

References

Beer, R. D. (1995). On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behavior*, 3(4):471–511.

Connor, J. O. (1985). Neural pacemakers and rhythmicity. *Annual Review of Physiology*, 47:17–28.

Durstewitz, D. (2003). Self-organizing neural integrator predicts interval times through climbing activity. *The Journal of Neuroscience*, 23(12):5342–5353.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor : The University of Michigan Press.

Maniadakis, M., Hourdakis, E., and Trahanias, P. (2014). Robotic interval timing based on active oscillations. *Procedia - Social and Behavioral Sciences*, 126:72 – 81. International Conference on Timing and Time Perception, 31 March – 3 April 2014, Corfu, Greece.

Saigusa, T., Tero, A., Nakagaki, T., and Kuramoto, Y. (2008). Amoebae anticipate periodic events. *Physical Review Letters*, 100:018101.

Trianni, V. (2008). *Evolutionary Swarm Robotics: Evolving Self-Organising Behaviours in Groups of Autonomous Robots*, chapter Decision-Making Mechanisms through the Perception of Time, pages 147–159. Springer Berlin Heidelberg, Berlin, Heidelberg.

The Prisoner's Dilemma, Memory, and the Early Evolution of Intelligence

Mikaela Leas[1,4], Emily L. Dolson[2,3,4], Riley Annis[2,4], Joshua R. Nahum[2,3,4],
Laura M. Grabowski[1,4] and Charles Ofria[2,3,4]

[1]College of Engineering and Computer Science, University of Texas Rio Grande Valley
[2]Department of Computer Science and Engineering, Michigan State University
[3]Ecology, Evolutionary Biology, and Behavior Program, Michigan State University
[4]BEACON Center for the Study of Evolution in Action
mikaela.leas01@utrgv.edu

Abstract

Memory is an essential component of intelligence as it enables an individual to make informed decisions based on past experiences. In the context of biological systems, however, what selective conditions promote the evolution of memory? Given that reliable memory is likely to be associated with costs, how much is it actually worth in different contexts? We use a genetic algorithm to measure the evolutionary importance of memory in the context of the Iterated Prisoner's Dilemma, a game in which players receive a short-term gain for defection, but may obtain greater long-term benefits with cooperation. However, cooperation requires trust; cooperating when an opponent defects is the worst possible outcome. Memory allows a player to recall an opponent's previous actions to determine how trustworthy that opponent is. While a player can earn a high payout by defecting, it will likely lose the trust of an opponent with memory, yielding a lower long-term payout. We determined the value of memory in the Iterated Prisoner's Dilemma under various conditions. When memory is costly, players reduce their available memory and use short-term greedy strategies, such as "Always Defect". Alternatively, when memory is inexpensive, players use well-known cooperative strategies, such as "Tit-for-Tat". Our findings indicate that organisms playing against a static opponent evolve memory as expected. However, memory is much more challenging to evolve in coevolutionary scenarios where its value is uneven.

Introduction

Biological evolution has produced our only examples thus far of general intelligence. As such, understanding the evolutionary process–both how it occurred in nature and how we can replicate it in a computer–may prove important on the path to developing artificial intelligence. One important component of such research is understanding the role of memory. Memory is the foundation of learning, allowing an individual to alter its future behavior based on prior stimuli (Sherry and Schacter, 1987). As such, memory is critical for such behaviors as navigating, tracking, foraging, avoiding predators, hunting prey and cooperating with others (Dunlap and Stephens, 2009; Grabowski et al., 2010; Liverence and Franconeri, 2015; Kraines and Kraines, 2000; Soto et al., 2014). These behaviors are sufficiently beneficial to fitness

that memory is advantageous to many individuals despite the associated biological costs (Barton, 2012; Dukas, 1999; Mayley, 1996). Understanding the importance of memory and the conditions under which memory evolves is crucial as it is a fundamental component to both real and artificial organisms.

To study the selective pressures that lead to the early evolution of memory, we need a way to measure their impact on memory's value. Here, we propose a technique for performing a cost-benefit analysis of memory via a simple evolutionary simulation. As an environment for this simulation, we will use the the classic game theoretic problem, Iterated Prisoner's Dilemma (IPD). Game theory provides a tractable framework for studying the value of memory in social contexts. IPD specifically is an ideal choice, because it is well-understood, requires memory for optimal performance, and is commonly used as a model system for studying cooperation (Axelrod, 1987; Crowley et al., 1996; Kraines and Kraines, 2000; Golbeck, 2002). In this game, two players repeatedly interact; at each step, they may cooperate with or defect from each other, and are rewarded according to the Prisoner's Dilemma payout matrix (see Table 1). The fact that IPD is so well-studied allows us to thoroughly validate this approach to studying memory. At the same time, we can gain useful insights into a relatively intuitive system before tackling more complex ones.

To assess the value of memory in this environment, we use a genetic algorithm to evolve strategies for playing IPD. Strategies in this algorithm are allowed to use memory, but at a cost. They must sacrifice part of their payout to have and use memory. By imposing a series of different memory costs and observing under which memory-using strategies evolve, we can measure the value of memory in this evolutionary context. Allowing evolution to generate novel strategies, rather than hard-coding in well-known strategies and allowing them to compete, ensures that we are not inadvertently introducing our own biases to the study system. To further ensure the validity of our system, we initially test it in a static environment where all players compete against a fixed set of three strategies. Overall, this system should

allow the evolution of individuals that use successful strategies in IPD, allowing us to determine the value of memory.

Iterated Prisoner's Dilemma

Three commonly-employed strategies for IPD are Always Defect, Always Cooperate, and Tit-for-Tat (Brunauer et al., 2007). The first two are the repetition of one action (defect or cooperate, respectively), while Tit-for-Tat is a strategy that repeats whatever action a player's opponent performed last. Always Defect and Always Cooperate do not require memory, as they do not rely on the history of a player's actions or those of its opponent. Tit-for-Tat, however, does require memory. In a single iteration of Prisoner's Dilemma, the best possible strategy is Always Defect; regardless of the opponent's decision, defecting will always yield a higher payout on a given iteration than cooperating would have (see Table 1). This consistent benefit makes Always Defect a selfish/greedy strategy (Axelrod, 1987).

	C	D
C	R = 3	S = 0
D	T = 5	P = 1

Table 1: **Payouts To Row-Player for Prisoner's Dilemma** Fitness is determined based on this matrix. Four payouts are possible: Reward (R), Sucker (S), Temptation (T), and Punishment (P). These payouts are a result of whether the player and the opponent each cooperate (C) or defect (D). In a single iteration, T is the highest payout for a single player. However, when playing repeated iterations of Prisoner's Dilemma, players can retaliate against each other, yielding lower payouts for both than if they had cooperated consistently.

When playing multiple iterations, cooperative strategies, such as Tit-for-Tat, outcompete the Always Defect strategy by allowing for the higher rewards associated with long-term cooperation (Axelrod, 1987; Crowley et al., 1996; Golbeck, 2002). To be successful, cooperative strategies must, among other things, be **forgiving** and **retaliating**; both of these attributes require memory (Axelrod, 1987). **Forgiving** strategies (eventually) cooperate in response to their opponents cooperating, even if the opponent defected in the past. Conversely, **retaliating** strategies (eventually) defect in response to their opponents defecting. Both of these strategies are only possible if the player is able to remember the opponent's actions. Thus, we can reasonably expect memory to be worth sacrificing some percentage of a player's payout, an assumption which is born out by prior research (Crowley et al., 1996).

Methods

Our system is a genetic algorithm, where fitness is based on the cumulative payout of IPD. A genetic algorithm is a

Strategy	AD	TFT	R	Average
AD (0)	1.00	1.06	3.00	1.69
TFT (1)	0.98	3.00	2.24	2.07
TTFT (2)	0.98	3.00	2.60	2.19

Table 2: **Payouts for Optimal Strategies for the Static Environment** A player's payout is determined from the Prisoner's Dilemma matrix (Table 1). In our static environment, the player competes against three static strategies: Always Defect (AD), Tit-for-Tat (TFT), and Random (R) over 64 iterations. The player's optimal strategy is dependent on the size of its memory. The AD strategy uses zero bits of memory, while the TFT strategy uses one bit of memory. When the player has one bit of memory, the best strategy is Tit-for-Tat. In this environment, the optimal strategy when a player has two bits of memory is to first cooperate and then defect any time the opponent has defected in the player's memory. This strategy is called Two-Tits-for-Tat (TTFT).

method for computationally solving problems that maintains and generates a population of potential solutions by selecting the most successful ones and allowing them to reproduce (Goldberg and Holland, 1988). There are four important components within a genetic algorithm: representation of a genotype, the initialization of the population, mechanism for selecting the next generation, and mutation operators (Mitchell, 1996). To facilitate validation of our approach via comparison to the results of previous research, we based our system off of systems that have successfully been used to study IPD in the past (Axelrod, 1987; Crowley et al., 1996; Kraines and Kraines, 2000). Crowley et al.'s set-up was a particularly strong influence, as their system allowed for flexible evolution of memory-using strategies. Our implementation is open source and available on GitHub: https://github.com/mikaelaleas/ChangingEnvironmentGA.

Representation of Genotype

Genotypes in our system are closely based off of those used by Crowley et al. (1996). An individual's genotype has three components: (1) the amount of memory it uses, (2) the initial state of its memory, and (3) its decision list. (1) The size of an individual's memory is the number of previous iterations for which it can remember its opponent's actions (as a simplification, organisms are unable to remember their own actions). Each bit of memory can hold information about one iteration. Since the decision list grows exponentially with the amount of memory used, we limit individuals to have no more than four bits of memory; that is, individuals can remember up to four iterations of their opponent's actions. (2) Next, since the memory is supposed to be a list of the opponent's actions, its initial state (before the opponent has actually played any iterations) biases the early decisions

made by an individual. The initial state of this memory is allowed to evolve. The individual's memory is subsequently updated every iteration of IPD, with the oldest past action being removed and the most recent action being added (see Figure 1). (3) Finally, the decision list is used to specify which action an individual will take, given a particular memory state (see Figure 2). The length of the decision list is 2^n, where n is the number of bits of memory, with an entry corresponding to every possible state of memory. The initial population, of size 500, was composed of individuals with each of these components randomly selected. Populations were allowed to evolve for 500 generations.

Figure 1: **Single Iteration of Prisoner's Dilemma** The player has three components: size (in bits), initial memory, and decision list. In a single round, a player will use the initial memory and decision list to decide whether to cooperate (C) or defect (D). A player's initial memory is updated every round to store the opponent's last action. The decision list does not change during an individual's lifetime. Here, player 1 cooperates with player 2 and player 2 cooperates with player 1. Player 1's initial memory is updated to reflect player 2's cooperation.

Selection of Next Generation

To select which individuals contribute offspring to the next generation: (1) a fitness score is generated for each individual, and (2) the population participates in a tournament. To determine a fitness score, individuals play 64 iterations of Prisoner's Dilemma (1 game) against competitors. In the static environment that we use to validate this approach, these competitors have three predetermined strategies: Always Defect, Tit-for-Tat, and Random. These three strategies were chosen to keep simplicity of the model, allowing for a focus on the evolution of memory-using strategies. In the coevolutionary environment, these competitors are randomly chosen from the population. Based on the IPD payout matrix, each individual is awarded a payout. This payout is

Figure 2: **Initial Memory and the Decision List** During a single iteration of Prisoner's Dilemma, a player chooses to cooperate (C) or defect (D) based on its decision list. Defect is represented with a 0 and cooperate with a 1. In this example, the initial memory is CD, which is represented as the binary number 10 (i.e. 2, in decimal). This points to index 2 in the decision list, which contains a C, so this player will cooperate in this iteration.

multiplied by the difference between 1 and the total cost of memory (accounting for all of the bits). The result is the fitness score.

$$fitness = payout(1 - cost * size) \qquad (1)$$

The fitness score calculation determines how the cost of memory affects the fitness of an individual. The cost of memory is fixed prior to the experiment. Finally, an average fitness for each individual is calculated. The next generation is produced through a tournament-style selection. The population is divided into subgroups of 10 individuals. The best half of the group–those with the highest fitness scores–are selected for the next generation. Note that this is a slightly gentler selection scheme than the one used by Crowley et al. (1996); we chose it because we felt that the reduced elitism was a better analog for the biological systems we are ultimately interested in understanding.

Mutations

Mutations occur probabilistically after the next generation is selected. There are three classes of mutations that can occur, corresponding to each of the portions of the genome: (1) size mutations, (2) initial memory mutations, and (3) decision mutations. All three types of mutations have a fixed probability of 0.01 of occurring when offspring are created. (1) A size mutation will increase or decrease the size of an individual's memory by 1 bit. This change affects the length of the decision list and the initial memory state of the individual. If the size of the memory is increased, the decision list will be duplicated meaning that increasing memory has no immediate effect on behavior unless one of the other types of mutations also occurs. However, if the size of the
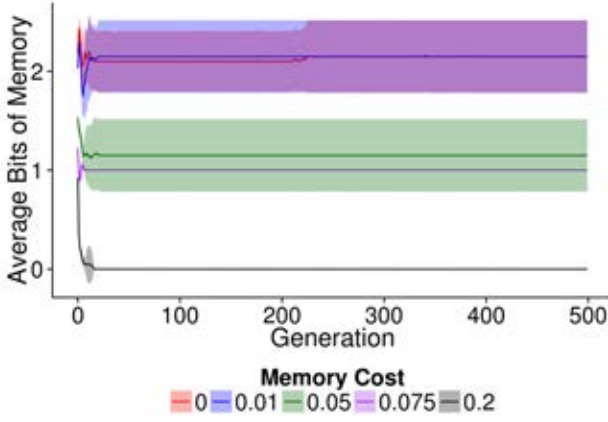
Figure 3: **Average Number of Bits of Memory Used By Cost of Memory** Shaded area represents standard deviation for each line. The cost of memory had a strong impact on the average number of bits of memory used by the population (Kruskal-wallis test, chi-squared = 168.44, df =8, p < .0001). When the cost of memory increases, the average number of bits of memory decreases (Post-hoc Wilcoxon Rank-sum test with Bonferonni correction). The average number of bits used at each memory cost are consistent with the predicted values from Table 3.

memory is decreased, the decision list is halved by removing the least significant bits (most distant in the past memory position). (2) The memory mutation affects the initial state of memory. This mutation will randomly choose an index of the initial memory and toggle the action (cooperate or defect) at that position. (3) The decision mutation targets the decision list. This mutation will randomly choose an index of the decision list and toggle the action (cooperate or defect) at that position.

Results and Discussion

Static Environment

To verify this system's efficacy, we started out by allowing strategies to evolve in a static environment in which each player competed against three static strategies: Always Defect, Tit-for-Tat, and Random. In this scenario, we can deterministically calculate how much a bit of memory should be worth in each context. The expected fitness and the highest memory cost for which players evolve to use memory is calculated from the Prisoner's Dilemma payout matrix, the individual's size, and the memory cost. The individual plays 64 iterations of IPD against each of the three strategies and receives payouts accordingly. The payouts are then adjusted according to the individual's size and the cost of memory, to determine the individual's fitness (Equation 1). Using more bits of memory allows the player to recall more previous actions of the opponent and thus determine which

Strategy	Cost	AD	TFT	R	Average
AD (0)	0.01	1.00	1.06	3.00	1.69
TFT (1)	0.01	0.97	2.97	2.22	2.05
TTFT (2)	0.01	0.96	2.94	2.55	2.15
AD (0)	0.075	1.00	1.06	3.00	1.69
TFT (1)	0.075	0.91	2.78	2.07	1.92
TTFT (2)	0.075	0.83	2.55	2.21	1.86
AD (0)	0.2	1.00	1.06	3.00	1.69
TFT (1)	0.2	0.78	2.40	1.79	1.65
TTFT (2)	0.2	0.59	1.80	1.56	1.32

Table 3: **Expected Average Fitness by Cost of Memory in the Static Environment** This table shows the expected average payout per iteration for the optimal strategies for 0, 1, and 2 bits of memory, adjusted by various costs of memory. The parenthetical next to each strategy name denotes the number of bits of memory that it uses. Here, we show three costs, each of which favors a different strategy: Always Defect, Tit-for-Tat, or Two-Tits-for-Tat.

strategy the opponent is using. Once an individual is able to determine its opponent's strategy, it may alter its future actions to increase its payout. This enables the evolution of better strategies that are able to retaliate against opponents if exploited. For example, an individual using the Always Defect strategy receives an average payout per iteration of 1.69 (see Table 2). If the cost of memory were 0.01 and the individual had one bit of memory, that payout would be reduced to 1.67. Using two bits of memory would further decrease the payout to 1.65. When there is no fitness cost, the optimal strategy is to start out cooperating, use the maximum allowed amount of memory, and defect any time an opponent has defected within memory. This will result in an individual always defecting after the first iteration against Always Defect, cooperating with Tit-for-Tat, and recognizing Random as frequently as possible. However, there are diminishing returns to adding additional bits of memory (see Table 2); in this simple setup, the greatest fitness improvement comes from adding the first bit, making Tit-for-Tat a possible strategy.

When a cost is applied to memory, the optimal strategy may change (see Table 3). If our system is accurately measuring the value of memory, we would expect to see Always Defect be the dominant strategy when the cost per bit of memory is 0.18 or greater, Tit-for-Tat be dominant when the cost is between 0.18 and .065, and so on. This result is almost exactly what we see in practice (see Figure 3). As predicted, this shift seems to be driven by an increase in Tit-for-Tat-style strategies as the cost of memory decreases (see Figure 4).

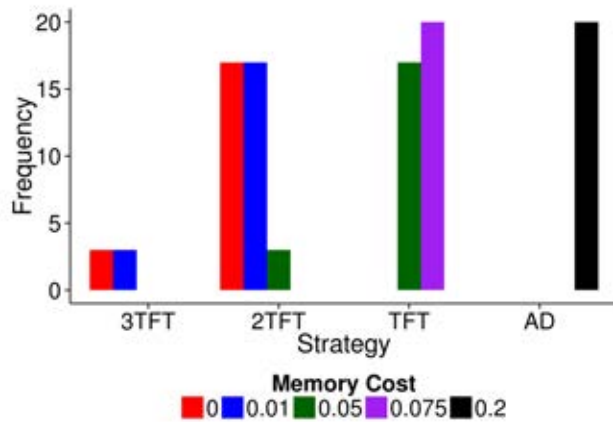The one slightly unexpected result is that, even when

Figure 4: **Most Common Strategies By Cost of Memory** We calculated the most commonly used (dominant) strategy in each of the 20 replicates within each of the 5 memory cost conditions. The four dominant strategies we observed were 3TFT (Three-Tits-for-Tat, the optimal strategy with three bits of memory), 2TFT (Two-Tits-for-Tat, the optimal strategy with two bits of memory), TFT (the optimal strategy with one bit of memory), and AD (the optimal strategy with no memory). As expected, the dominant strategy depended on the cost of memory. Increasing the cost of memory increases the frequency with which less memory-intensive strategies are dominant.

Figure 5: **Memory Usage in Coevolutionary Environment (Low Mutation Rate)** Shaded area represents standard deviation for each line. Memory use consistently evolved only when memory had no cost; the average amount of memory used in this condition was significantly different from the amount used in all of the other conditions (Kruskal-wallis test and post-hoc Wilcoxon rank-sum test with Bonferonni correction, chi-squared = 55.93, df =5, p < .0001). In all of the other conditions, the average amount of memory used gradually declines over time.

memory has no cost, strategies don't tend to use much more than three bits of memory. We hypothesize that this is due to the following mechanism: Every additional bit of memory doubles the size of an individual's decision list. An excessively large decision list is at increased risk of experiencing genetic drift away from the optimal values. Thus, the potential fitness gain from adding a fourth bit of memory may not be worth the increased risk of the lineage making incorrect moves later on. Such a scenario would be consistent with the decreased recognition accuracy found by Crowley et al. (1996).

mutation type) and high (.1 for each mutation type).

At a low mutation rate, memory proves far less useful in this more complex environment, as evidenced by the fact that it is not consistently used if it has any cost associated with it (see Figure 5). As in the previous experiment, increasing the memory cost increases the percentage of replicates in which Always Defect, rather than Tit-for-Tat, becomes the dominant strategy. When examining individual runs, a common pattern takes place. The initial population of Tit-for-Tat is frequently invaded by Always Cooperate. Always Cooperate can displace Tit-for-Tat (in the absence of other competitors) because it receives the same payout, but does not have to pay any cost for memory. Once Tit-for-Tat is extinct (or nearly so), Always Defect arises and quickly displaces Always Cooperate. In the low mutation rate replicates, Tit-for-Tat rarely is generated via mutation from Always Defect, leading to a stable population that is trapped at a sub-optimal strategy. Although Crowley et al. did not analyze the strategies that evolved in their system, these results are consistent with theirs in that they too observed that applying a cost to memory resulted in decreased cooperation (Crowley et al., 1996).

Interestingly, memory use in a coevolutionary context increases at the higher mutation rate (see Figure 6). When memory is free in this treatment, strategies quickly evolve to use the maximum allowed amount of memory, suggesting that the implicit costs of making use of a large memory are

Coevolutionary environment

Having demonstrated that our methodology accurately measures the value of memory in a system, we can now move on to a more interesting case. Instead of placing solutions in a static environment, we can allow them to compete against each other. This scenario introduces complex coevolutionary dynamics that would normally confound attempts to measure the value of memory. In this setup, the population is initially populated with Tit-for-Tat (one bit of memory) and each individual plays IPD with each other individual in its tournament to determine its fitness. Like before, the top-half of each tournament is allowed to reproduce. We ran this treatment at two different mutation rates: low (.01 for each
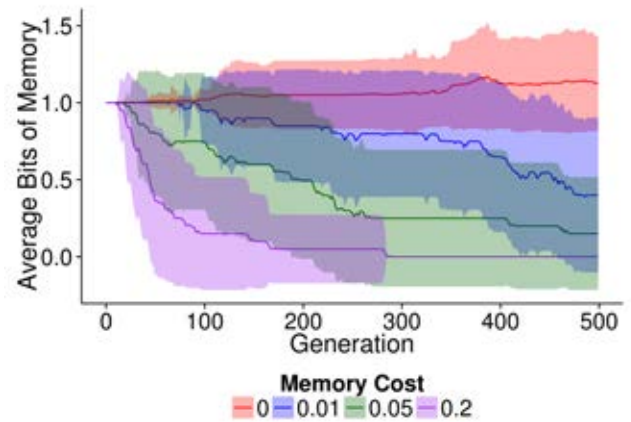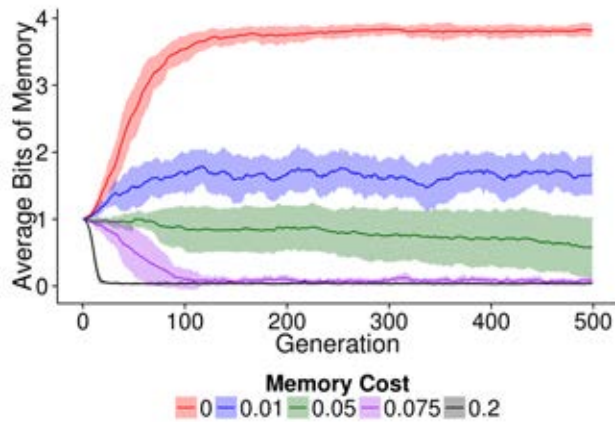
Figure 6: **Memory Usage in Coevolutionary Environment (High Mutation Rate)** Shaded area represents standard deviation for each line. Populations evolved to use more memory at lower costs. At memory costs of .075 and higher, the average amount of memory used by the population after 500 generations was not significantly different from 0 (Kruskal-wallis test and post-hoc Wilcoxon rank-sum test with Bonferonni correction, chi-squared = 95.24, df =5, p < .0001).

overwhelmed by coevolutionary selective pressures. Alternatively, the large decision lists that individuals with a lot of memory have may serve to increase mutational robustness. This effect would be in contrast to the results observed in the static environment and in previous research (Crowley et al., 1996). Understanding the relationship between these factors would be an interesting direction to explore in the future.

In the condition with no memory cost, Tit-for-Tat is the most common strategy in approximately half of the replicates, a finding which is consistent with Tit-for-Tat's dominance in the Axelrod tournament (Axelrod, 1987). Among the other half of the replicates there is an incredible diversity of most common strategies - only two of the other replicates have the same most common strategy. Applying any cost to memory causes the population to converge to well-known strategies (see Figure 7). These results align with Mayley's finding that applying a cost to learning (analogous to memory, in our case) substantially inhibits the exploration of strategies that would require it (Mayley, 1996).

Conclusion

We demonstrated the evolutionary value of memory by using a genetic algorithm that awards fitness based on the results of many iterations of the Iterated Prisoner's Dilemma. Under static environmental conditions, the population often evolved to use memory, despite it being costly, as long as it provided a substantial gain in payout. In fact, the extent to which memory was used aligned nearly perfectly with theoretical predictions about the costs and benefits of memory
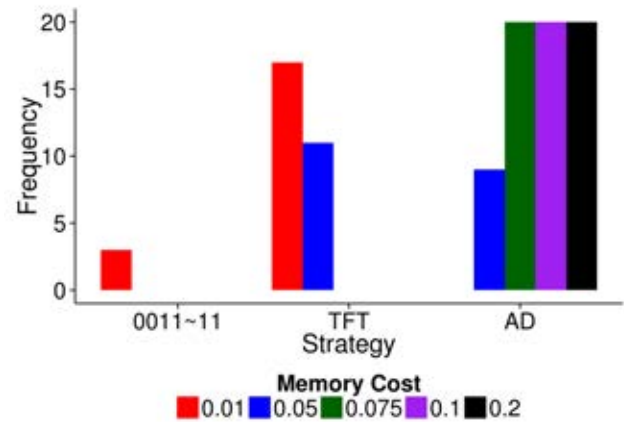
Figure 7: **Most Common Strategies in Coevolutionary Environment (High Mutation Rate)** Again, Tit-for-Tat is more frequently the dominant strategy at lower memory costs. Note that this figure does not include the strategies used when there was no cost to memory, because there were too many of them. Approximately half of the replicates in the 0 cost condition of this treatment used Tit-for-Tat, and the other half each had a different dominant strategy (although most of the dominant strategies were not dramatically more prevalent than other strategies in the population). Also note that the strategy on the far left, 0011~11 is denoted only by its genotype (decision list~initial memory), as it does not correspond to a well-known named strategy. It cooperates initially, and any time its opponent cooperated two iterations ago.

in this system. This result demonstrates that the technique proposed here is an effective way to quantify the value of memory in evolutionary contexts. By simply giving memory a fitness cost and observing whether memory evolves we can assess its importance in complex scenarios.

In more dynamic environments, we observed that memory was valuable when there were no costs because it enabled cooperation. However, it was easily evolved away under high memory costs (where Always Defect could rapidly overtake Tit-for-Tat) or low mutation rates (where Always Cooperate could outcompete Tit-for-Tat and subsequently be outcompeted by Always Defect). While this phenomenon illustrates the difficulty of measuring the value of memory in an environment where that value keeps changing, our results were consistent with the findings of prior research and we were able to more fully investigate the mechanisms behind them.

While we were able to show the value of a single bit of memory, the evolutionary dynamics explored here generally did not provide a substantial benefit to having larger amounts of memory. In light of these early findings, we plan to extend this research, both in static environments (to test our

analytical predictions of the value of memory) and in dynamic coevolutionary environments (to study the practical evolution of memory in realistic scenarios).

For static environments, we plan to explore the evolutionary response of players to imperfect opponents, such as those that attempt to engage in Tit-for-Tat, but make occasional errors. A single mistake can spiral into a high level of defection and much lower overall payouts, but if a player uses larger amounts of memory, it will be able to recognize and forgive mistakes for a longer-term benefit. We will also explore introducing longer-term memory that the player can set as it chooses. We will provide these players with combinations of opponents that require long-term memory to receive optimal payouts, such as Always Cooperate and Tit-for-Tat. In such cases, a player with long-term memory will be able to initially probe to determine whether its opponent responds negatively to a defection. If so, it can play Tit-for-Tat from then on (starting with a cooperation). On the other hand, if the opponent does not retaliate, the player knows that it can play Always Defect from then on out for a larger payout.

Dynamic environments have an even wider potential for helping us learn more about the evolution of memory. As of now, it is challenging to evolve cooperative strategies *de novo*. They require memory to increase–immediately incurring a cost–but no gain is realized until a cooperative strategy is in place and multiple players are using it and interacting. We plan to explore structured populations with smaller, local groups where kin selection effects can dominate and selection is weaker, allowing these strategies to more easily come into play. We plan to also explore more stabilizing forces once players are engaging in cooperation so that it doesn't evolve away as easily as we saw here.

Overall, this work is an important step in studying the early evolution of memory utilization, and insights from it are likely to be valuable in informing other real and artificial life studies involving the evolution of intelligence.

Acknowledgments

References

Axelrod, R. (1987). The evolution of strategies in the iterated prisoners dilemma. *The dynamics of norms*, pages 1–16.

Barton, R. A. (2012). Embodied cognitive evolution and the cerebellum. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 367(1599):2097–2107.

Brunauer, R., Lcker, A., Mayer, H. A., Mitterlechner, G., and Payer, H. (2007). Evolution of Iterated Prisoner's Dilemma Strategies with Different History Lengths in Static and Cultural Environments. In *Proceedings of the 2007 ACM Symposium on Applied Computing*, SAC '07, pages 720–727, New York, NY, USA. ACM.

Crowley, P. H., Provencher, L., Sloane, S., Dugatkin, L. A., Spohn, B., Rogers, L., and Alfieri, M. (1996). Evolving cooperation: the role of individual recognition. 37(1):49–66.

Dukas, R. (1999). Costs of memory: ideas and predictions. *Journal of Theoretical Biology*, 197(1):41–50.

Dunlap, A. S. and Stephens, D. W. (2009). Components of change in the evolution of learning and unlearned preference. *Proceedings of the Royal Society B: Biological Sciences*, 276(1670):3201–3208.

Golbeck, J. (2002). Evolving strategies for the prisoners dilemma. *Advances in Intelligent Systems, Fuzzy Systems, and Evolutionary Computation*, 2002:299.

Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99.

Grabowski, L. M., Bryson, D. M., Dyer, F. C., Ofria, C., and Pennock, R. T. (2010). Early Evolution of Memory Usage in Digital Organisms. In *Artifical Life XII*, pages 224–231.

Kraines, D. P. and Kraines, V. Y. (2000). Natural Selection of Memory-one Strategies for the Iterated Prisoner's Dilemma. *Journal of Theoretical Biology*, 203(4):335–355.

Liverence, B. and Franconeri, S. (2015). Human cache memory enables ultrafast serial access to spatial representations. *Journal of Vision*, 15(12):1292.

Mayley, G. (1996). Landscapes, learning costs, and genetic assimilation. 4(3):213.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA.

Sherry, D. F. and Schacter, D. L. (1987). The evolution of multiple memory systems. *Psychological Review*, 94(4):439–454.

Soto, D., Rotshtein, P., and Kanai, R. (2014). Parietal structure and function explain human variation in working memory biases of visual attention. *NeuroImage*, 89:289–296.

Ecology

Social Systems and Ecosystems: History Matters

Inman Harvey

Evolutionary and Adaptive Systems Group, University of Sussex, Brighton, UK
inmanh@gmail.com

Abstract

In both social systems and ecosystems there is a need to resolve potential conflicts between the interests of individuals and the collective interest of the community. The collective interests need to survive the turbulent dynamics of social and ecological interactions. To see how different systems with different sets of interactions have varying degrees of robustness, we need to look at their different contingent histories. We analyse abstract Artificial Life models of such systems, and note that some prominent examples rely on explicitly a-historical frameworks; we point out where analyses that ignore a contingent historical context can be fatally flawed. Real life studies highlight the role of history, and Artificial Life studies should do likewise.

Introduction

In both ecosystems and social systems there are at least two levels at which, speaking loosely, 'lifelike' processes can be observed. There is one level at which the individual organisms, animals, humans are interacting with each other and pursuing their individual interests. But also there is a second level of ecosystem organisation, or social organisation, which provides the context within which they exist. In principle the same individuals could function, perhaps more or less successfully, if the ecosystem/social organisation was changed. One extreme version of such a change would be for the ecosystem/social organisation to break down in chaos, which is often against the interests of the individuals concerned. Systems survive or die, just as individuals do.

Ecosystems versus Social Systems. Social organisation can be the outcome of a social contract where individuals have chosen to agree to a set of rules. Ecosystems do not involve such explicit choice. Regardless of such differences, in both cases one can analyse individual behaviour in terms of self-interest potentially clashing with the interests of others around. In social systems we may call some actions 'cheating' and some consequences 'punishment'. In ecosystems we tend to avoid such moral overtones and merely discuss 'actions' and 'consequences'; the analyses may nevertheless be similar.

How do they Persist? If a specific ecosystem/social system survives for a long time, explanation is called for. If no external authority is responsible for imposing this, then the organisation must be an emergent consequence of individual patterns of behaviour that are globally somewhat resilient to the perturbations of everyday life. We may ask how *one specific* ecosystem/social system manages to persist, or we may ask about *generic* properties needed for persistence.

What is their Origin? Each specific ecosystem/social system will have its own unique history, from origins up to the present day; just as each organism has its unique genetic and

	Real Life	**Artificial Life**
Social systems	Bitcoin (Nakamoto, 2008) Common pool resources (Ostrom, 1990)	Iterated Prisoner's Dilemma (Press and Dyson, 2012) (Stewart and Plotkin, 2013)
Natural systems	Ecosystems Niche construction (Clements, 1916) (Lewontin, 1969)	Daisyworld (Watson & Lovelock, 1983) (Harvey, 2015) Complex systems (May, 1972)

Table 1: Classes of decentralised social systems and natural (eco-)systems and their Alife counterparts analysed here.

developmental history. It is the main thesis of this paper that *generic* theories, that gloss over or average such *specific* histories, often fail to capture salient features of reality. Examples of such theories will be criticised.

Real Life. We consider both real systems and their artificial life counterparts, as in different columns of Table 1. Amongst real social systems we look at Bitcoin as a money transfer system, and common pool resource governance as studied by economists. Natural systems refer here to ecosystems as studied by ecologists in the field. It will be suggested that those studying such real life systems will have no problems agreeing with the thesis that history matters. Hence this paper is mainly targeted at those producing abstract models that explicitly leave out any consideration of history.

Artificial Life Models. Simple abstract models of social systems are illustrated here by examples from IPD, Iterated Prisoner's Dilemma. This is based on a classic two-person game where each player has simple choices and the interactions between them have consequences in terms of different payoffs. The basic dilemma of individual cheating versus cooperation is distilled into this simplest form. Ecosystem models discussed here include Daisyworld models where the organisms and environmental influences are characterised as variables interacting in a dynamical system.

Mathematical Summary. If processes are actually non-Markovian, modelling them as Markovian will lead to error.

Plain Language Summary. Real life takes place in a world of accumulated historical accidents that affect how social and ecological processes actually function. History matters.

Artificial Life Models of Governance

The Introduction to 'Leviathan' (Hobbes, 1651) gives the first known reference to artificial life under that name:

> NATURE (the art whereby God hath made and governes the world) is by the art of man, as in many other things, so in this also imitated, that it can make an Artificial Animal. For seeing life is but a motion of Limbs, the beginning whereof is in some principall part within, why may we not say that all Automata (Engines that move themselves by springs and wheels as doth a watch) have an artificiall life?

This introduces the metaphor of a nation state as an artificial man, Leviathan, with different components functioning together as mechanically deterministic parts but forming a living whole. What sort of governance can provide some form of global harmony ensuring cooperation and collaboration between component parts and reconciling any potential conflicts between them? How does some form of social contract arise (and continue to survive) from a natural state of anarchy? Hobbes' answer was for central rule by an absolute sovereign. Though such a sovereign is ultimately driven by his private interests, these are aligned with the public interests in so far as "The riches, power, and honour of a monarch arise only from the riches, strength, and reputation of his subjects".

Here we follow Hobbes' surprisingly modern notion of artificial life, and the use of models such as Automata in the study of real life governance. However we part company with him on his assumption of a need for a central sovereign. Leviathan is Hobbes' exemplar of central authority, but in all further examples we discuss below there is no central authority, no rules for behaviour are imposed from outside.

Distributed Social Systems, Choice, Attractors

When governance arises solely through interactions between individual participants, different styles of governance can only arise through the different choices they make. Strategies any one individual has for choosing are typically conditional in part on the choices the others have made. These individual choices bind into a social system when there is a stable pattern that persists despite potential disturbances from within or without. In dynamical systems terms, we are looking for the attractors of such systems. There may be many possible such attractors, some more congenial than others to the participants — e.g. with higher payoffs in utilitarian terms.

Ecosystems, Choice, Attractors

With natural ecosystems we may not be considering the same type of explicit strategies or choices seen in social systems. Nevertheless, a different type of 'choice' is available, a choice of where to locate, which environment to inhabit. Animals may move from valley to hilltop; even plants, over generations, can shift their habitat. In this subtly different sense of choice, the component members of an ecosystem have 'chosen' to coexist in a specific locale where their various interactions (including their own knock-on effects on the environment) allow them to thrive. In the theoretical space of all conceivable ecosystems, there is a multitude of such viable and robust locales that act as potential attractors.

Real Social Systems

We focus here on two classes of real social systems with distributed governance, chosen to illustrate different facets of dependence on history: Bitcoin and common pool resource systems. Such systems are only stable if they are indeed currently near an attractor, which is another way of saying that they recover from small disturbances. We look at how such systems may adapt to changing circumstances over the longer term, and hence the role of historical contingency in how they come to be in one attractor rather than another.

Bitcoin

In commerce there needs to be common agreement about who paid what to whom; this is often centrally regulated by banks maintaining records. One version of this governance problem is that of verifying money transfers over the internet, and a very different style of solution is provided by Bitcoin (Nakamoto, 2008). Here the maintenance of book-keeping records is distributed, not centralised. The protocol used has to reconcile private interests with public interests; an individual would (dishonestly) benefit by spending the same money twice, but a money transfer system only works if such double-spending is prevented. Roughly speaking, this replaces trust in a sovereign central bank with trust in a majority consensus of multiple independent record-keepers distributed across the internet. This may be compared with a simple natural biological example where consensus amongst bacteria can be achieved via 'quorum sensing' (Miller and Bassler, 2001).
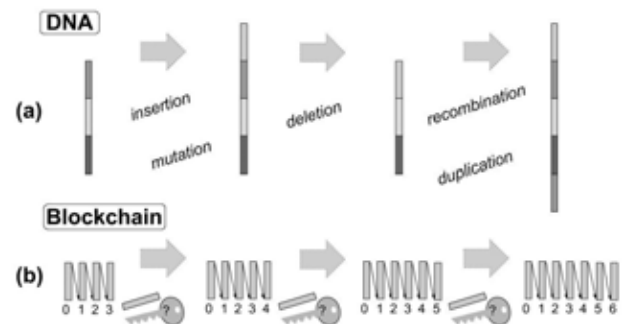
Figure 1: (a) DNA has statistical continuity over phylogenetic history, with noise. Older and newer data both matter. (b) Blockchain is built up systematically with new blocks added at the end, verified by consensus via 'key-finding' for each addition.

Blockchains, DNA and Costs. In Bitcoin the official record of all transactions in recorded history is maintained in a data object called a blockchain. Somewhat like DNA, this is a linear string of digits, meaning it is virtually free and instantaneous to copy. Multiple copies can be distributed widely. Like DNA, it can grow incrementally over time. Unlike DNA, the blockchain of accounting records cannot mutate or have parts excised; the protocol has to maintain accuracy and integrity across all copies of the blockchain as it is updated with new transactions (Figure 1). New transactions

are bundled together into a block to be added on to the end of the blockchain; then a deliberately lengthy and computationally expensive process is undertaken by each record-keeper to find a 'key' to validate it. This cryptographically-based key must identify both the old (mutually agreed) transaction history and the new block of transactions. Different record-keepers may have different updates to add (i.e. different new blocks), but the protocol must ensure agreement on just one of these as authoritative.

With DNA, the stored data is expensive to accumulate. Evolution has selected *for* what is transmitted and preserved, at the expense of numerous other versions that were selected *against*. With blockchains there is also expense, but arising differently in the key-finding exercise; the need for this expense arises from the need to avoid cheating. From a dynamical systems perspective, both with blockchains and with DNA (within a single species) there is maintenance of a steady state far from equilibrium, a steady state that preserves the information across multiple copies. But this is a metastable steady state with possibilities for change and growth, as the history is not only maintained but added to.

Though blockchain technology is not near cost-free (as some mistakenly think) and has weaknesses discussed below, its subtle use of history makes it a powerful tool with uses that go beyond the financial record-keeping of Bitcoin.

Common Pool Resources

Bitcoin has distributed, not centralised control. At a different scale, practical working examples of decentralised control can be seen in societies across the world ranging from water authorities in California to shared forest usage in Nepal and Switzerland and shared fisheries in Turkey. These are maintained and policed by the participants themselves rather than imposed by some external sovereign authority. Such 'common pool resources' have been the focus of economist Ostrom (1990). She proposes a list of design principles or 'best practices' that are common to such robust institutions:

(1) Clear identified boundaries between those people and resources *within* the institution and those *outside*.
(2) Appropriation rules congruent to local social and environmental conditions.
(3) All (or most) members share in making or changing rules.
(4) People who are users (or accountable to them) monitor the appropriation and resource management.
(5) Sanctions for rule violations are graduated from low to high according to the severity or persistence of violations.
(6) Conflict-resolution mechanisms are local and rapid.
(7) External authority, e.g. higher government, does not enforce its own rule contrary to that of the local institution.
(8) Where there are multiple levels of governance they are organised in multiple nested layers.

In such common-pool scenarios, unlike Bitcoin, anonymous entry or participation is not possible. The potential for the system to adapt itself according to changing local circumstances further differentiates this from what may be a serious weakness of Bitcoin. All participants have not only a stake in *maintaining* the rules (principles 4 and 6) but also in *changing* them (principle 3). Such adaptation in the governance system needs to be congruent with local social and environmental conditions (principle 2); and the social conditions may include further higher or lower level layers of governance, overlapping in a nested fashion (principle 8). Within the generic constraints of these 8 principles there is scope for a multitude of possible governance systems each adapted, more or less, to local circumstances and fashioned through a historical succession of contingencies.

ALife Models of Social Systems: IPD

We move from stability, contingency, history in *real* systems to the same issues in ALife models. Recent innovations in IPD (Iterated Prisoner's Dilemma) provide a case study.

Motivation for IPD Models. These provide a minimal model of 2 agents ('prisoners') interacting. They must decide on actions independently, but the payoff to each depends on what they both decide, and is designed to provide a conflict between individual and collective gains.

The supposed story is that they have agreed beforehand to deny everything about some joint crime, but now they are interviewed separately by the police. Each has to decide whether to keep quiet as promised ('Cooperate' with the other prisoner) or make some deal with the police ('Defect'). In terms of utility, they both receive R (say 3) if both Cooperate; both receive P (1) if both Defect; and if one Defects, the other Cooperates the payout is T (5) to the defector and S (0) to the other. The choice of (T, R, P, S) = (5, 3, 1, 0) (Figure 2a, following Press and Dyson, 2012) meets the PD condition T>R>P>S that implies whatever agent 2's decision is, agent 1 would gain more by Defecting than Cooperating. The further condition 2R>T+S implies the total payout for both Cooperating, 2R, is higher than the total payment when one Cooperates and the other Defects.

The rules treat each agent symmetrically, so any difference in outcome depends solely on how their strategies interact. In a single game with no further consequences, each agent maximises their payoff by Defecting, irrespective of the other agent's choice. Hence they both Defect (D), receiving 1 each, whereas if both Cooperated (C) each would have received 3.

If such games are iterated indefinitely, in the IPD, then each agent's actions may influence future responses. Under some circumstances a regime of Cooperation for mutual benefit can arise; IPD studies usually focus on just what conditions allow this and discourage cheats (i.e. Defectors). Such conditions provide counter-examples to Hobbes' intuition that only a sovereign authority can guarantee a mutual Commonwealth.

Tit for Tat, TFT. A typical class of IPD strategy depends (either deterministically or probabilistically) on memory of the previous choices made by each agent in the previous N rounds, N≥1. Tit for Tat (TFT), for example, is the memory-1 strategy where an agent copies the action that the other agent took in the previous round (Axelrod, 1984; Figure 2b). Tit-for-Two-Tats is the memory-2 strategy where an agent only defects if the opponent defects twice in a row. More generally a memory-N strategy can be specified as a table with 4^N cells, relating to 4 possibilities CC, CD, DC, DD (for own +opponent choices) on N previous rounds, each cell specifying the probability that C will be chosen by this agent in the new round. For example, TFT with memory 1 has these probabilities of Cooperating, dependent on the previous round: CC 100%, CD 0%, DC 100%, DD 0%.
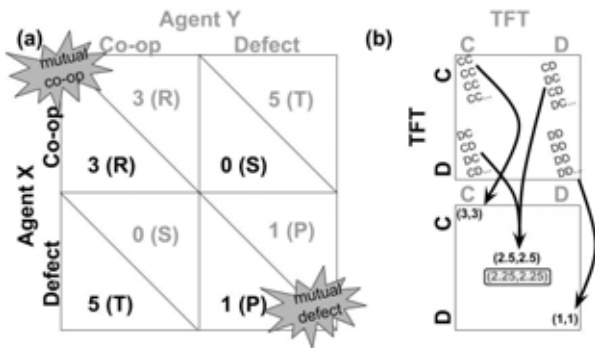
Figure 2: (a) The IPD payoff table (Press and Dyson, 2012). (b) Tit-for-Tat players differentiate into TFT$_C$ or TFT$_D$ (opening play C or D). Different varieties meeting (upper square) lead to 3 different end-attractors, average scores of (3,3), (1,1) and (2.5,2.5) (i.e. average of (5,0) and (0,5), lower square). The weighted (25%, 25%, 50%) average of all these attractor scores is different yet again, (2.25,2.25).

Historical and A-Historical Agents

Such memory-1 strategies depend explicitly on short-term memory of the previous round; but they also depend on long-term history of starting conditions, since the very first move makes a difference, say C for TFT$_C$ or D for TFT$_D$. There are two possible routes to finesse this issue, the first being to acknowledge that TFT$_C$ and TFT$_D$ are indeed two different strategies with different consequences (Figure 2b). Only when TFT$_C$ meets another TFT$_C$ does the virtual circle of Cooperation take off. If both are instead TFT$_D$ then a vicious circle of Defection takes over. A TFT$_C$ meeting TFT$_D$ results in alternating CD, DC choices. The starting conditions have a permanent effect on which basin of attraction is entered.

A second way to finesse this issue is to arrange affairs so that initial conditions eventually become irrelevant, and this could be the case with sufficient noise in the system. If with high enough probability a choice is accidentally reversed, then over enough iterations of IPD all possible basins of attraction will be visited. In a classic early Alife paper (Lindgren, 1991) explicitly used this method. There is a cost to be paid for finessing matters this way, however: TFT$_C$ and TFT$_D$ are now indistinguishable in such a theory, despite the fact that over any finite run they typically have totally different behaviours.

In principle the IPD game iterates for an arbitrary number of rounds, not known in advance. If both players know it to be the final game, this becomes a one-shot PD where both must rationally Defect. In turn, the penultimate game falls to the same analysis, and so on back to the first. An infinite series of rounds avoids this trap, but is impossible in practice. But we can have a finite, non-predetermined, number of rounds by arranging *after* each iteration a small (e.g. 1%) chance that it is *then* deemed to be the last. Then if any noise (as introduced by Lindgren) is small in comparison to this 1%, strategies such as TFT$_C$ and TFT$_D$ will be visibly seen to operate in different basins of attraction. Real world scenarios typically resemble this pattern rather than the infinite-iteration limit. For such real world scenarios, the history will matter.

This distinction between historical and a-historical agents is the central focus of this paper. Behaviour of the latter depends only on recent short-term events held in 'memory', whereas the former also depends on one-off longterm origins in history. Crudely, this can be related to different perspectives from Biology and Physics. Typically many biologists are interested in a specific species or ecosystems with a specific evolutionary history (which we can relate to TFT$_C$ or TFT$_D$). In contrast physicists, broadly speaking, may be happier making broad generalisations across some arbitrary range of entities (which we can relate to Lindgren's TFT); often this makes the mathematics more tractable. Taken to extremes, this can result in broad statements that are generically true about "all possible organisms" assuming ergodicity, thus including extant organisms on this planet together with all extinct organisms, and indeed all conceivable organisms on all conceivable planets; but nevertheless misleading about any one specific non-ergodic organism. What is true about generic a-historical IPD agent TFT can be false about TFT$_C$ or TFT$_D$.

Press and Dyson

A recent ground-breaking IPD paper (Press and Dyson, 2012), displayed a novel class of memory-1 ZD (Zero Determinant) strategies. These allow an agent — provided it no longer had the simple ambition to maximise its own payout that traditionally is expected in IPD — to tailor its strategy to guarantee that the opponent's payout will average some value such as 1.5 (between P and R) regardless of how the opponent responds. Or such an extorting agent can guarantee that the excess of payoff above P will be shared in unequal proportions such as 3:1. The details of these ZD strategies are not discussed here. They are highly novel and counter-intuitive and are acknowledged by others to be valid, given the context; but many of the conclusions Press and Dyson drew have been shown to be misplaced (Stewart and Plotkin, 2013). We summarise these points, then go even further in questioning the validity of their Markovian assumptions..

Extortionate ZD Strategies. Suppose agent_X chooses an extortionate ZD strategy that gains a bigger proportion of the excess rewards (above a base-level of P) regardless of agent_Y's responses. Then if agent_Y is an optimising player that adjusts strategy so as to increase its own payoff (Press and Dyson call this an evolutionary player) the result is that agent_X scores even higher. The erroneous implication Press and Dyson draw is that in an evolutionary scenario where multiple strategies are competing against each other, such extortionate strategies will triumph and dominate. As Stewart and Plotkin (2013) and other commentators point out, this is not so. If extortionate players came to dominate an evolutionary scenario, they will typically be competing with similar extortionate strategies. If agent_X and agent_Y are both forcing their excess payout (above P=1) to be 3 times greater than their opponents, this is neatly resolved by the excess being 0 for each, the (1,1) score of mutual Defection.

Generous ZD Strategies. It turns out that so-called Generous ZD strategies — that roughly speaking do the opposite of extortion in making sure that differential benefits mostly accrue to their opponents — will dominate in an evolutionary scenario. Such Generous strategies behave optimally against other Generous strategies, and also replace non-cooperative ZD strategies (Stewart and Plotkin, 2013).

Such ZD Strategies Ignore Historical Contingency

The main contribution of this paper to this novel development in IPD studies is to point out what other commentators have apparently missed: this whole class of ZD strategies, whether extortionate *or* generous, has been set up to be a-historical and hence to be largely irrelevant as models of human (or animal) strategies; since these are typically historical, contingent and contextual. Press and Dyson (2012) explicitly set up their ZD strategies to use the same finesse Lindgren (1991) uses, as discussed above, to average over all possible contingent longterm histories; they focus on generic strategies dependent on short-term memory alone. Indeed, they go further than Lindgren in showing that such Markovian assumptions allow any memory-N strategy to be generically equivalent to (some other) minimal memory-1 strategy.

Their proof covers the TFT strategy averaged over all possible histories, but fails cover a TFT$_C$ strategy, even with its short history of a single first move. *A fortiori*, such IPD results have even less relevance to the real world when e.g. analysing the mating behaviour of *this* specific butterfly, with its long evolutionary and ecological history of multiple overlapping constraints as context; or when analysing the governance system for *that* Turkish communal fishing arrangement, with its long social and cultural history of multiple over-lapping polycentric social contracts. Ostrom (1990) explicitly mentions congruence with local social and environmental conditions among her design principles observed in long-lasting common pool governance systems, and this historical contingent contextually is what is stripped away in such generic mathematical proofs. Mathematically, one cannot analyse non-Markovian processes as if they were Markovian.

Real Ecosystems

We now consider the systems in the lower row of Table 1, starting with a minimal overview of real ecosystems.

Ecological Succession. This is the observed process of change in structure of an ecological community over the medium to long term. For instance after a mass extinction a typical sequence is for a few species of plants and animals to initially return; then successive new organisms arrive, building on what is already there in what Ostrom might want to call multiple nested polycentric layers in analogy to her social systems. In some cases this may be a somewhat predictable succession towards a final 'climax community' (Clements, 1916); but more recent ideas tend to take account of the many historical contingencies involved, including the varied feedbacks through knock-on environmental effects, and see a more unpredictable picture of 'alternative stable states' (Lewontin, 1969). In the short-term an ecosystem is in a stable steady state, but in the longer term it is somewhat accidental which one of many such possible equilibria it is, and what range of fellow organisms it contains.

Niche Construction. Such theories emphasise that organisms may not be merely accepting or selecting (through moving to) their specific environment; they may also have an active role in changing it (Laland and Sterelny, 2006).
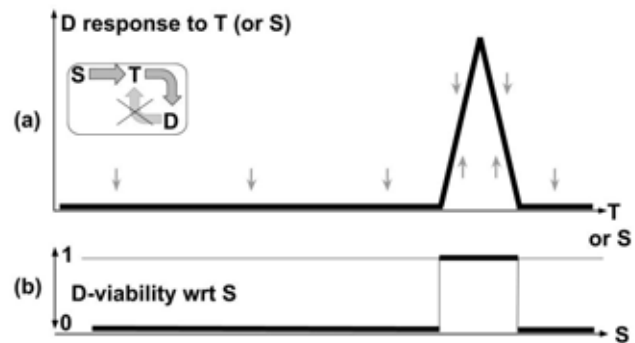

Figure 3: No-feedback scenario: environmental perturbation S (solar output) directly affects local env. T (temperature) which directly affects organism D (daisies). (a) D assumed to have steady-state dependency, 'hat-shaped' function of T, giving limited zone of viability. (b) D-viability (binary yes/ no) plotted against perturbation S (here scaled to match T).

ALife Models of Ecosystems

Daisyworld (DW) models (Watson and Lovelock, 1983; Harvey, 2015) offer a simplified vision of how organisms and environment interact in some sense cooperatively. This can be compared to a very basic form of niche construction.

Motivation for Daisyworld Models. These are not widely known, and where known largely misunderstood (Harvey, 2015). The rationale is to model a number of types of organisms (e.g. one being 'daisies' D) that can survive within a limited range of local environmental conditions (e.g. one being 'temperature' T). Collective survival of an ecosystem of different organisms means *all* of them are currently viable in their local environment; *robustness* of an ecosystem is measured in terms of how wide a range of perturbing environmental conditions it can survive; e.g. an external 'sun' S creating hotter/colder conditions. An organism may have some local environmental effect (e.g. the albedo of a black daisy may raise local temperature), and *complexity* is measured as the number of such different effects within the ecosystem. The key DW result is that more such *complexity* leads to greater ecosystem *robustness*.

We demonstrate this, starting from the simplest ecosystem with a single species; for equations see Appendix A. Figure 3 shows schematically the basic influence of environment T on an organism D. Figure 4 shows the consequence of further adding an effect from the organism D onto the environmental variable T. The consequence is to extend, i.e. widen the range of solar forcing (perturbing external effect S) for which the organism is viable (Harvey, 2015). Here the effect is positive (e.g. the albedo of a black daisy increases local temperature) the solar viability range is extended towards lower values than otherwise. A negative effect (e.g. white daisies tend to reflect heat and decrease temperature) would extend the solar viability range towards higher values.

Plus and Minus, Rein Control. Further, if both variants are potentially available with both positive *and* negative effects
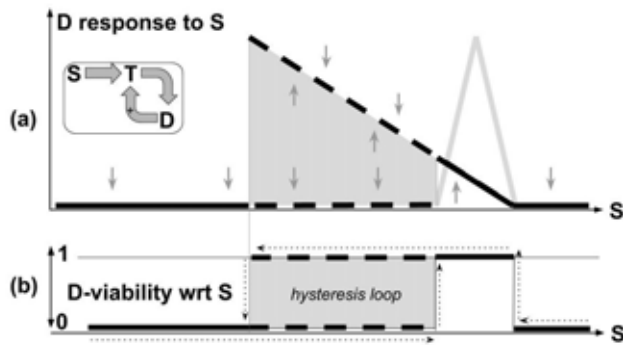
Figure 4: As Fig. 3 plus a further influence of D on T (here positive, black daisies increase temp.) (a) Peak response of D to changes in S is shifted, with a hysteresis loop. (b) D-viability zone is extended (here to the left) by a buffer zone, only effective if entered from high S values. and not low ones.

Figure 5: As Fig. 4 plus white daisies D_W affect negatively their local temp T_W as well as black daisies D_B affecting positively T_B. (a) shows steady-state values of each D (b) shows viability of D_B&D_W (simultaneously viable), against S.

on the local environmental variable, temperature, they will collectively expand their joint eco-niche, as seen in Figure 5. This phenomenon depends on some basic assumptions spelt out in Appendix A; each variant, black or white, largely determines its own local temperature but with some 'leakage' between them in their shared environment. In this model, interactions between different 'species' such as D_B and D_W are only mediated via environmental variables, rather than through e.g. direct predation of one on the other. The results here, and developed further in Harvey (2015), demonstrate that any changes in viability range (for D_B&D_W, or D_B, D_W individually) always increase the range and never decrease it.

The expanded viability range takes the form of hysteresis loops as in Figure 5b. If the external perturbing force, here S, changes slowly, then which of the upper (viable) or lower (non-viable) arms of such loops is followed depends on which direction they are approached. In this sense history matters.

This is an example of 'rein control' (Clynes, 1969; Harvey, 2004). Clones observed a pattern when natural organisms exhibit homeostasis in response to external forces threatening viability both from above and below (e.g. both 'too hot' and 'too cold'). Rather than one mechanism responding in two directions, he noted two mechanisms each responding in one direction only. Since reins of a horse have this same property, each pulls but does not push, he called this 'rein control'.

This is further related to Le Chatelier's principle (Le Chatelier and Boudouard, 1898) as known to chemists and economists. This principle asserts that when any system in equilibrium is disturbed the system will adjust itself so as to (at least partially) nullify the effect of the change. A practical application of this principle is the use of a *buffer solution* which resists changes in pH when acid or alkali is added. These can be designed by chemists (Scorpio, 2000), or seen naturally where the *bicarbonate buffering system* regulates pH of blood in humans or other animals (Krieg et al., 2014).

Multidimensional Daisyworld

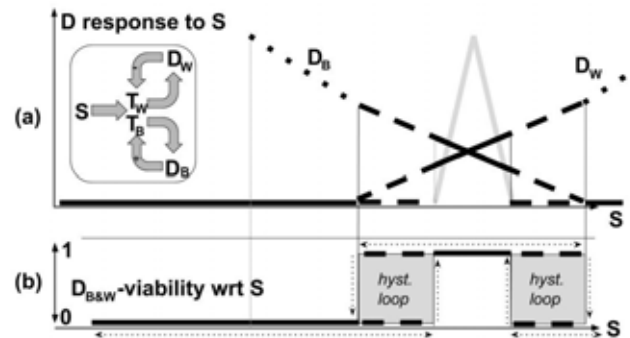So far we only considered one environmental variable at a time: say temperature in DW, pH in the buffering examples.

What if two or more such variables are simultaneously relevant, e.g. both temperature and pH?

We can answer this within any very simple, abstract class of ecosystem models where (any number of) 'organisms' are modelled by 'hat-shaped' viability functions of (any number of) environmental variables; and in turn the organisms have any effect of any kind, positive or negative, on each or all of the environmental variables. In such cases it has been shown in the 'Gaian Regulation Theorem' (Harvey, 2015) that hysteresis loops or buffer zones as illustrated above exist regardless of the dimensionality of any such system. Perturbations in any number of dimensions will tend to be countered so as to widen — and never lessen — the viability range of any disparate group of organisms in an ecosystem, or of individuals in a corresponding social system.

As an abstract example, Figure 6a shows 8 groups of 8 species in clusters of narrow preferences for 3 environmental variables. In the absence of DW feedbacks at most one such group could be viable since the small viability spheres do not intersect (only P, V spheres shown here). If we add DW effects, different for all 8 members within each group, then when an external perturbation happens to pass the neighbourhood the whole group of 8 becomes jointly viable with a viability radius greatly expanded (from 0.05 to 0.218 for effect size 0.4; details in Appendix B). The expanded viability spheres (as V-sphere in Figure 6b) may now overlap and (depending on environmental history) several such groups may become simultaneously viable. If the effect size were increased to 0.8, the mid-value perturbation C (0.5,0.5,0.5) would be within all 8 potentially expanded spheres, allowing all 64 (8x8) species with diverse environmental limits and diverse effects to be simultaneously viable.

This proof of principle still only has 3 dimensions of environmental variables, and is symmetrically set up to demonstrate an effect. Real systems will typically have more dimensions and be highly asymmetrical and locally varied, with convoluted overlaps of basins of attraction. Nevertheless we can see that different perturbation trajectories may result in very different ecosystems. Trajectories matter, history matters.

Such meandering paths through ecosystem-space can be compared with meandering evolutionary paths through DNA-
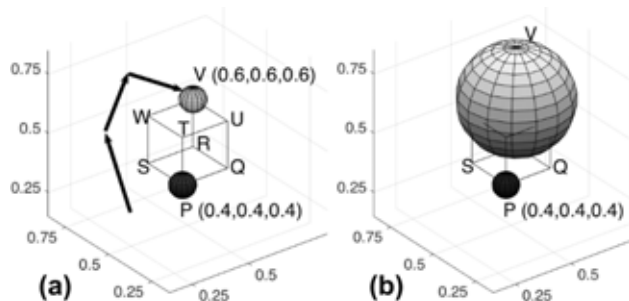
Figure 6. (a) 3 dimensions of external env. perturbations. P is group of 8 species, preferred env. (0.4,0.4.0.4), viable within radius 0.05 of this as shown by P sphere. Similar groups centred on Q, R…W, at corners of cube. Arrows show a possible trajectory of external perturbation. (b) This passes through viability zone of group V, DW effect consequently expands its viability radius to 0.218. See Appendix B.

space, and some degree of resemblance is not entirely accidental. From a high-level perspective the viability functions of DW can be related to the survival focus of Darwinian evolution. The natural settlement into attractors of the broad class of dynamical systems that is multidimensional DW relates directly to the natural selection of Darwinian evolution. Indeed the latter may be seen as a special case of the former. They both have surprising and counter-intuitive consequences; for instance an increase in an effect that *increases* the range of DW robustness usually *decreases* Darwinian fitness (Harvey, 2015).

Where an A-Historical Analysis Differs

The analysis of ecosystems in terms of DW, as presented here, is controversial (Harvey, 2015). One influential analysis (May, 1972) of an even broader class of 'any large complex system' (that includes multidimensional DW) purports to contradict it, proposing that, after some critical number of variables is exceeded, such systems are inherently unstable. Three mathematical flaws in this analysis have been previously exposed (Harvey, 2011). We here go further in identifying these flaws as arising from an a-historical analysis that resembles the a-historical analysis of IPD (Press and Dyson, 2012; Stewart and Plotkin, 2013).

May (1972) picks out an arbitrary equilibrium point of a large complex system and analyses its properties. This arbitrary choice, together with other explicit or implicit assumptions he makes, allows one to draw general conclusions; as the system gets larger, the chance that this specific equilibrium is stable tends towards the vanishingly small. This part of May's argument resembles creationists' arguments about the improbability of the 'irreducibly complex'. But the fact that the probability of an arbitrary lottery ticket being a winner becomes arbitrarily small as the lottery itself gets arbitrarily big does not stop there being a winning ticket, or indeed many such.

A dynamical system left to its own devices will naturally head towards a stable equilibrium, such a 'winning ticket'; any unstable equilibrium will only be briefly observed. As external conditions change, such a system inevitably passes through a sequence of metastable states. Thus any observed equilibrium is almost inevitably a stable one; which

equilibrium it is depends on the history of the system. May's analysis of a generic a-historical equilibrium state has little relevance for the analysis of specific, observed, historically contingent equilibria (Harvey, 2011). Likewise the analysis by Press and Dyson (2012) of extortionate ZD strategies for IPD, or of Stewart and Plotkin (2013) of generous ZD strategies, has little relevance for historical contingent strategies such as TFT_C or TFT_D.

Conclusions

Crudely speaking, biology equals physics (and chemistry) plus history — stability in the short term plus the contingent context arising from an extended history of stability. More elegantly put, "Biology has always occupied a middle ground between the determinism of classical physics and the uncertainties of history" (Smith and Morowitz, 1982). When the physics of short-term stability is the focus of attention to the exclusion of contingent history, key concerns that can characterise complex systems can be missed.

It may be more than a coincidence that Press, Dyson, Stewart, Plotkin and May, variously cited and criticised above, all come from physics backgrounds. Another physicist, Rutherford (Birks, 1962), is quoted as saying "All science is either physics or stamp-collecting". If the latter is interpreted as contingency, it need not be taken as derogatory; this is not only important for understanding real biology and social science but equally so for Artificial Life models of these.

In biological systems internal DNA is one obvious marker of a history, but other external markers may also be crucial. In polycentric social contracts (Ostrom, 1990) there may be multiple overlapping simultaneous systems of governance; likewise in polycentric organisms, polycentric ecosystems. Adaptations (and neutral changes) in any one system layer are within (and constrained by) the contingent current context of the others. Complexity of the whole arises through such adaptive/neutral trajectories through history, and cannot be explained a-historically.

A specific novel observation in this paper, apparently not noted by other commentators, is that the recently discovered extortionate ZD strategies in IPD (Press and Dyson, 2012) , together with their generous cousin strategies (Stewart and Plotkin, 2013), have very little relevance to any biological or social studies of cooperation because they are all avowedly a-historical. Their Markovian assumptions are mathematically powerful but implausible as models of reality. The same applies to May's (1972) analysis of large complex systems.

In passing we have noted that the blockchain of Bitcoin in its present form cleverly maintains the global history of transactions, and the full history is needed to establish the current state of accounts; in this sense the blockchain is historical. However the institutional framework of Bitcoin currently has no mechanism for adaptive change as per Ostrom's principle 3; Bitcoin itself is a-historical.

Successful real social systems and ecosystems have a history of adapting to circumstances, and this gives context to their current stability. Artificial Life models should reflect this, and there are currently many promising research areas that give scope for developing currently deficient a-historical models to take account of such contingency. History matters.

Appendix A

Figure 5 shows 'black' and 'white' daisies, D_B and D_W, and respective local temperatures T_B, T_W (Harvey, 2004). Figure 4, using D_B only, is similar except that D_W is clamped to 0.

Daisy viability w.r.t. local temperature is based on a 'hat-shaped' function $H(T)$ with (Figure 3) peak value 1.0 at T_{opt} reducing to zero outside some limited viability range. Results are not qualitatively changed by different hat shapes.

$$(1) \quad H(T) = \max(0, 1 - abs(T_{opt} - \alpha T))$$

Parameter α sets slope of hat. hence radius ($=1/\alpha$) of daisy-viability in terms of its local temperature. Parameter β sets the rate at which daisy-viability moves towards the hat-function:

$$(2) \quad dD_B/dt = \beta (H(T_B) - D_B)$$
$$(3) \quad dD_W/dt = \beta (H(T_W) - D_W)$$

The local temperature T_B, of black daisies D_B is based on the solar insolation S, altered (i) by *positive* influence from the black daisies, and (ii) by equilibration towards T_W. T_W is conversely affected, white daisies have *negative* effect. On the assumption that temperatures settle faster than rate of change of Daisies we can use the steady-state values as in (Harvey, 2004). Using T′ for intermediate values of T, phase (i) is:

$$(4) \quad T'_B = S + \gamma D_B$$
$$(5) \quad T'_W = S - \gamma D_W$$

where γ parameterises the effect size for black/white daisies increasing/decreasing their own local temperatures. Phase (ii) gives the final temperature T as a compromise between each individual T′ and their average current values; there is some 'leakage' (Harvey, 2004), here parameterised via δ (for $0 \leq \delta \leq 1$), between temperatures of black and white daisies:

$$(6) \quad T_B = \delta T'_B + (1 - \delta)(T'_B + T'_W))$$
$$(7) \quad T_W = \delta T'_W + (1 - \delta)(T'_B + T'_W))$$

If we choose $\delta = 0.5$, then algebraic manipulation shows that equations (4,5) together with (6,7) can be replaced by:

$$(8) \quad T_B = S + \varepsilon (3 D_B - D_W)$$
$$(9) \quad T_W = S + \varepsilon (D_B - 3 D_W)$$

where for convenience we substitute ε ($= \gamma/4$) for parameter γ.

Equations (1), (2,3) and (8,9) can be simulated computationally by choosing some specific value for S, and running these equations from starting values for D, T, until steady-state is reached. In hysteresis regions, the end-states reached will depend on the starting states. To plot one branch of each hysteresis loop, S should be initialised at a low value, and the computation run until D, T reach steady-state. Then S is incremented slightly, keeping *current* values of D, T as new starting values for the next run; this is further repeated, through to high values of S. If the process is then reversed, moving from high S to low S, the other branches of the hysteresis loops can be plotted. In Figure 5b, the viability of D_{B+W} is plotted as: IF ($D_B > 0$ AND $D_W > 0$) plot 1, ELSE plot 0.

Appendix B

Figure 8 shows 3 dimensions of external env. perturbations. Viability of group of 8 species at P is 1.0 at (0.4,0.4,0.4), decreasing linearly to 0.0 at radius (Euclidean distance) 0.05. Each species has different +/- effects on 3 respective env. variables ($2^3 = 8$ variants); signs differ, but effect size is always 0.4. The other 7 groups (Q,…,W) are formed similarly.

Effects of a P-species are multiplied by their viability and have two local contributions: half serves to shift the P-group

local env. away from the perturbing force (and is thus shared with other P-members; 'leakage'); and half shifts the species-specific env. away from the P-local env. Over a trajectory of env. perturbations, at each point 20,000 computational iterations altered viability by 0.001 and local env. variables by 0.005 of their indicated shift. This smoothing of dynamics, together with the inheritance of previous env. values as perturbations changed, avoided numerical instabilities. A species was considered extinct if viability<0.01.

An effect size 0.4 expanded viability radius of each group from 0.05 to 0.218; effect size 0.8 expanded it further to 0.35.

References

Axelrod, R., (1984). The evolution of cooperation. Basic Books, NY.

Birks, J. B. (1962). Rutherford at Manchester. Heywood, London.

Clements, F. E., (1916). Plant succession; an analysis of the development of vegetation. Carnegie Institute of Washington.

Clynes, M., (1969). Cybernetic implications of rein control in perceptual and conceptual organisation. *Ann. NY Acad. Sci.* 156:629-670.

Harvey, I., (2004). Homeostasis and rein control: from Daisyworld to active perception. In Pollack, J., Bedau, M., Husbands, P., Ikegami, T. and Watson, R. A. (Eds.), *Proc. 9th Int. Conf. on Sim. and Syn. of Living Systems, ALIFE 9.* MIT Press, Cambridge, MA.

Harvey, I., (2011). Opening stable doors: complexity and stability in nonlinear systems. In Lenearts, T. et al., (Eds.), *Advances in Artificial Life, ECAL 2011,* pp 805-812, MIT Press.

Harvey, I., (2015). The circular logic of Gaia: fragility and fallacies, regulation and proofs. In Andrews, P., Caves, D., Doursat, R., Hickinbotham, S., Polack, F., Stepney, S., Taylor, T. and Timmis, J. (Eds.), *Proc. Eur. Conf. on Artificial Life 2015*, MIT Press.

Hobbes, T., (1651). Leviathan. Andrew Crooks (publisher), at the Green Dragon in St. Pauls Church-yard, London.

Krieg, B. J., Taghavi, S. M., Amidon, G. L., Amidon, G. E., (2014). *In vivo* predictive dissolution: transport analysis of the CO_2, Bicarbonate *in vivo* buffer system. *J.Pharm. Sc.* 103(11):3473-3490.

Laland, K. N. and Sterelny, K., (2006). Perspective: seven reasons (not) to neglect niche construction. *Evolution,* 60(9), 1751-1762.

Le Chatelier, H. and Boudouard, O., (1898). Limits of flammability of gaseous mixtures. *Bull. de la Soc. Chim. de France,* 19:483-488.

Lewontin, R. C., (1969). The meaning of stability. *Brookhaven Symposia in Biology,* 22:13-23.

Lindgren, K., (1991). Evolutionary phenomena in simple dynamics. In Farmer, J. D., Rasmussen, S. and Taylor, C., (Eds.), *Artificial Life II.* Edison-Wesley, Redwood City, CA.

May, R. M., (1972). Will a large complex system be stable? *Nature* 238, 413-415.

Miller, M. B. and Bassler, B. L., (2001). Quorum sensing in bacteria. *Annu. Rev. Microbiol.* 55:165-199.

Nakamoto, S., (2008). Bitcoin, an electronic peer-to-peer cash system. Url: https://bitcoin.org/bitcoin.pdf

Ostrom, E., (1990). Governing the Commons: the evolution of institutions for collective action. Cambridge University Press.

Ostrom, E., Walker, J. and Gardner, R., (1992). Covenants with and without a sword: self-governance is possible. *American Political Science Review* 86(2), 404-417.

Press, W, H. and Dyson, F. J. (2012). Iterated Prisoner's Dilemma contains strategies that dominate any evolutionary opponent. *Proc. Nat. Acad. Sci.* 109(26), 10409-10413.

Scorpio, R. (2000). Fundamentals of acids, bases, buffers and their application to biochemical systems. Kendall Hunt, Dubuque, IA.

Smith, T.F. and Morowitz, H. J., (1982). Between history and physics. *J. Mol. Evol.* 18(4), 265-282.

Stewart, A. J. and Plotkin, J. B., (2013). From extortion to generosity, evolution in the Iterated Prisoner's Dilemma. *Proc. Nat. Acad. Sci.* 110(38), 15348-15353.

Watson, A. J. and Lovelock, J. E., (1983). Biological homeostasis of the global environment: the parable of Daisyworld. *Tellus* 35B:284-289.

How ecological inheritance can affect the evolution of complex niche construction in a 2D physical simulation

Naoaki Chiba, Reiji Suzuki and Takaya Arita

Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan
Email: chiba@alife.cs.is.nagoya-u.ac.jp

Abstract

Niche construction is a process in which organisms modify the selection pressure on themselves and others through their ecological activities. Various evolutionary models of effects of niche construction on evolution have revealed that they bring about unexpected evolutionary scenarios. However, little is still known about how niche-constructing behaviors of complex physical structures (such as nest-building) can emerge through the course of evolution, even though it is one of the most ubiquitous and significant niche-constructing behaviors. Our purpose is to obtain knowledge of the emergence and evolution of physically-grounded niche construction and the effect of its ecological inheritance on evolution. We construct an evolutionary model in which a virtual organism has to arrive at a goal by constructing a physical niche composed of objects in a physically simulated environment. In particular, we focus on effects of the degree of ecological inheritance, which is represented as a weathering probability of ecologically inherited objects from a parent to its offspring. We show that it has a nonlinear effect on the adaptivity of the population. In the case of no ecological inheritance, adaptive niche-constructing behaviors such as valley-filling or ramp-placing strategies emerged, which created complex structures composed of multiple objects. It also turned out that the stable ecological inheritance of constructed structures could increase the adaptivity of the population by allowing an organism to maintain the inherited and adaptive structures while the unstable ecological inheritance rather decreases the adaptivity of the population by making previously adaptive structures maladaptive obstacles.

Introduction

All creatures, to a greater or lesser extent, change their own and others' niches through their ecological activities, which modify the selection pressure on themselves and others. This process is called "niche construction" (Odling-Smee et al., 2003). The niche construction processes are seen in ecological activities of many species such as plants (photosynthesis), nonhuman animals (nest building) and human. Recently, the niche construction is also recognized as an important factor in considering an open-ended evolution (Taylor, 2015).

A typical example of a niche-constructing organisms are earthworms that change the structure and chemistry of soils
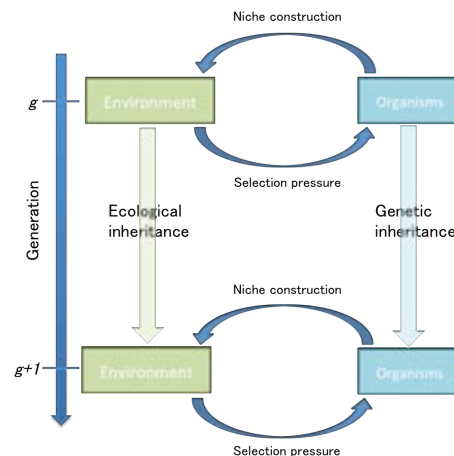

Figure 1: A diagram of evolution, niche construction and ecological inheritance (based on Odling-Smee et al., 2003).

through their burrowing behaviors. These changes are accumulated over generations, and then bring about different environmental conditions, which expose the successive population to different selection pressure. This effect is also called "ecological inheritance", as it makes the generation inherit a legacy of modified selection pressures from ancestral organisms (Odling-Smee et al., 2003).

The effects of niche construction on evolution have been mainly investigated using both mathematical and simple computational models, in which effects of niche-constructing behaviors are represented as changes in variables that represent the environmental states. The environmental state in Laland et al.'s model of population genetics (Laland et al., 1996) is represented as the amount of resource, which can be directly increased by the niche-constructing behavior. Han et al. extended a version of Laland et al.'s model to a patch occupancy model in which the amount of distributed resources can be modified through niche construction (Han et al., 2009). They showed that three different spatial patterns of metapopulation emerged depending on the different ecological imprint from niche

construction. Suzuki and Arita constructed an evolutionary model into which both learning (change in a phenotypic value) and niche construction (change in an optimal phenotypic value) were incorporated, and reported that a cyclic coevolution of genes for learning and niche construction was observed in the case of the low temporal locality of ecological processes (Suzuki and Arita, 2010). Harvey also constructed a simplified version of the Daisyworld model in order to understand interactions between the environment states (i.e., the temperature of the planet) and organisms (i.e., black and white daisies that increase and decrease the temperature, respectively) (Harvey, 2004). He reported that the homeostasis of the global temperature emerged through the change of the proportion of black and white daisies against the global warming. These studies clarified various effects of niche construction on evolution in the cases when the environmental state is represented as a quantity (e.g., resources, optimal phenotypic values, temperature).

On the other hand, an important feature of niche construction is that it can create physical and complex structures composed of many components, which cannot be represented quantitatively. A nest building is a typical and ubiquitous example of such a behavior. For example, a beaver makes a dam with branches: It stems the flow of a river and have an influence on many organisms (Odling-Smee et al., 2003). Taylor presented an individual-based model of complex niche construction that can make the shape of one-dimensional fitness landscape complex (Taylor, 2004). He showed that the evolved organisms that performed more complex niche constructions had more genes, which implies a continuous increase in the complexity of organisms. Kojima et al. constructed an evolutionary model in which individual has a strategy for a Prisoner's Dilemma and a trait for creating a physical structure that can limit social interactions between neighboring individuals (Kojima et al., 2014). They found that, when the degree of ecological inheritance was high, a stable pattern of the physical structure emerged. It enabled cooperators to reduce the number of interactions with defectors while keeping that number with cooperators moderate. Although these studies discussed complex or physical niche construction, they are still abstract in the sense that physically-grounded interactions between organisms and environments were not considered.

The purpose of this study is obtaining knowledge of the emergence and evolution of physically-grounded niche construction and the effect of its ecological inheritance on evolution. In particular, we focus on the effects of the degree of ecological inheritance, which could be different due to properties of niches and ecological dynamics. We discuss whether and how complex interaction processes between organisms and environments can bring about non-trivial evolutionary dynamics depending on the different degree of ecological inheritance.

We adopt a model of virtual organisms, which is rec-
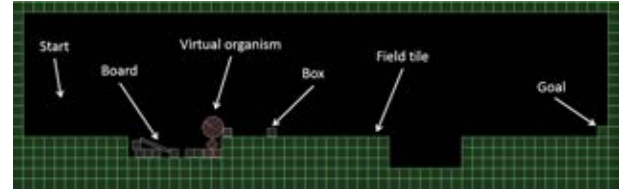

Figure 2: The field for fitness evaluation.

ognized as a novel platform to discuss recent topics in evolutionary research such as eco-evolutionary feedbacks (Ito et al., in press). We construct an evolutionary model in which a virtual organism has to arrive at a goal by constructing a physical niche composed of some objects. We adopt a physically simulated environment based on a physics engine for 2D games. We show that the degree of ecological inheritance can have a nonlinear influence on the adaptivity of the population, facilitating or retarding the evolution of niche-constructing behaviors of complex and physically-grounded structures.

Model

Field and task

We use the Box2D (Catto, 2016), which is an open source physics engine for 2D games, in order to introduce a physically simulated environment into our model. Box2D can simulate physical interactions between 2D objects such as friction and collision between rigid bodies.

In our model, we assume an x-y coordinate plane that represents a horizontal and vertical space, and there exists gravity along the y-axis toward the bottom. The simulation is updated every infinitesimal time step dt (second). Hence, the physical environment is updated $1/dt$ times in one second. We used the default parameters that define the properties of physical environment with a few modifications[1].

We assumed a 1160×360 virtual space as shown in Fig. 2. A field consists of squared "field tiles" with a side length of 20. There is a special field tile on the right end of the field, representing a goal. There are two valleys composed of field tiles in the field, and the left one is shallow and wide whereas right one is deep and narrow. The virtual organism is put on a starting point on the left end of the field at first. The task for the organism is to move from the start to the goal as many times as possible within the time limit T seconds under the assumption that the organism is returned to the start after the goal. Specifically, the fitness of the virtual organism is calculated by the following equation (Eq. 1):

$$fitness = g + \frac{disgoal - dis}{disgoal}, \quad (1)$$

---

[1]Gravity $g = 9.8\,(\mathrm{m/s^2})$, density $\rho = 1.0\,(\mathrm{kg/m^2})$, coefficient of friction of a virtual organism $\mu_i = 0.7$ and coefficient of friction except for a virtual organism $\mu = 0.3$.

where g is the number of times for which the virtual organism arrived at the goal, $disgoal$ is the distance between the start and the goal, and dis represents the distance between the goal and the position of the virtual organism at the end of fitness evaluation. Therefore, the more the number of times of arrival at the goal is and closer to the goal at the final step the virtual organism gets, the higher fitness it gets. We expect that a non-niche-constructing organism will get stuck in the first valley because it cannot climb the valley while niche-constructing organisms have a possibility to reach the goal by placing some objects in the field.

Virtual organism

In our model, a circular-shaped organism with the radius of 20 can move in the field by rotating its body to the left or right (Fig. 3). It also can place objects[2] in the field. This is a niche-constructing behavior in our model in the sense that constructed structures can affect the future adaptivity of the organism. There are two types of objects: "box" with a side length of 16 and "board" which is a 6×60 rectangle. A virtual organism has two areas around it: the visibility and the motion range of its arm (Fig. 3, right). The visibility is a round shaped region around the organism with the radius of FV and the organism can recognize objects and field tiles within this area. The motion range of its arm is also a round-shaped region with the radius of L and the organism can place objects within this area. Placed object will fall on other objects or field tiles due to the gravity if it is placed in the air. There is no cost for placing objects.

A three-layer neural network, of which weights are defined by the genotypes of an organism, determines the behavior of the organism (Fig. 3, left). We use a sigmoid function as a transfer function in the neural network. The values are inputted to the neural network every time when the physical environment is updated. The following values are inputted to the input layer: the number of field tiles, boxes and boards within the visibility; the relative x-y positions of their center of gravity from the organism; and the number of available objects, which will be explained later.

The output layer consists of one neuron which decides a direction of rotation and five neurons related to placing objects. The first neuron decides which direction the virtual organism rotates and moves. If its output value is higher than 0.5, the clockwise torque τ of which the magnitude is $100,000 \, (\mathrm{kgf} \cdot \mathrm{m})$ is applied to the virtual organism, otherwise anti-clockwise torque is applied to it. The second neuron decides whether the virtual organism places an object or not. If its value is larger than 0.5, the virtual organism places the object in the field. The third neuron decides which object the virtual organism places in the field if it does. If its value is higher than 0.5, the virtual organism places a box, otherwise it places a board. The fourth and fifth neurons decide

---

[2]In this paper, the term "objects" represents boxes and boards placed by an organism. It does not include field tiles.
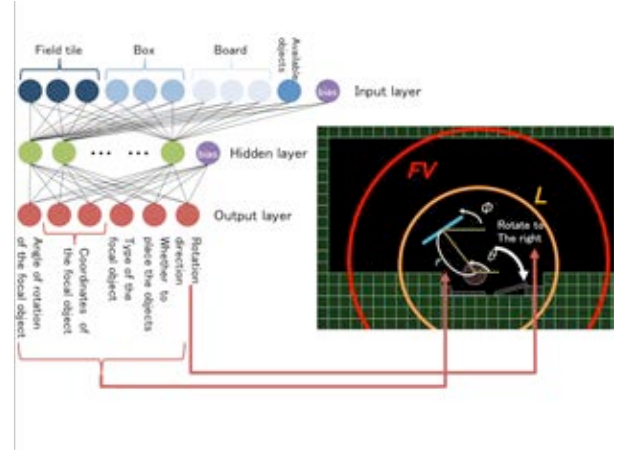

Figure 3: The neural network of a virtual organism.

the position on which the virtual organism places the object within the motion range of its arm. The position is represented by the polar coordinates $(r, \theta) = (L \times x_4, 2\pi \times x_5)$, where x_4 and x_5 represent the fourth and the fifth output values, respectively. The last neuron decides the angle of rotation of the object $\phi = 2\pi \times x_6$, where x_6 represents the sixth output value. If the focal object will interfere the existing field tiles or objects in the field, or will be outside of field, the action of placing the object is canceled and nothing happens.

Moreover, the parameter B determines the maximum number of the objects that can exist in the field. It reflects the amount of available resources for niche-constructing behaviors. The organism can make use of the number of available objects for decision making as an input to its neural network.

Evolution and ecological inheritance

A virtual organism has synaptic weights of its neural network of which values are determined by its own genes. The population of organisms evolves according to a genetic algorithm. In the initial generation, there are N virtual organisms and the values of their genes are randomly assigned between -1.0 and 1.0. After the fitness evaluation of all organisms, a pair of parents is selected by a roulette-wheel selection in accordance with the fitness. They produce a pair of two offspring with the same genotypes as themselves, and a two-point crossover occurs between the offspring with a probability PC. Each gene can mutate with a small probability PM. If a mutation occurs, a random number $\in [-DM, DM]$ is added to a value of the gene. This process will be conducted until the number of the offspring reaches N.

Furthermore, we introduce an ecological inheritance into the model in order to investigate its effect on the evolution of niche construction. In a pair of the offspring, the environmental state of one parent is inherited to the environment of
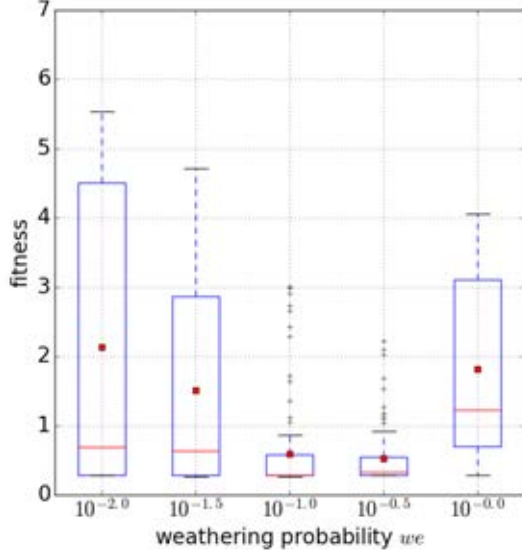
Figure 4: The fitness and the weathering probability we.

Figure 5: A valley-filling strategy.

Figure 6: A ramp-placing strategy.

Figure 7: Obstacles preventing an organism from crossing a valley.

one offspring, and the environmental state of the other parent is also inherited to that of the other offspring. Specifically, each offspring inherits the environmental state of the corresponding parent at the end of its fitness evaluation process. This means that the all objects in the parent's environment will be copied to the offspring's environment, keeping their types, positions and rotations the same.

In addition, the degree of such ecological inheritance can vary depending on environmental conditions in the real world. Thus, we also introduce a probability we into our model, which represents a probability of weathering of each object. Each inherited object vanishes according to the probability we. The higher the probability we is, the less the virtual organism inherits the objects. We conduct the whole process of evolution and ecological inheritance through G generations.

Result

We conducted evolutionary experiments using the following parameters: $N = 50$; $G = 2,000$; $dt = 0.02$; $T = 200$; $B = 25, 40$ and 55; $FV = 500$; $L = 125, 250$ and 500; $we = 10^{-2.0}, 10^{-1.5}, 10^{-1.0}, 10^{-0.5}$, and $10^{-0.0}$; $PC = 0.7$; $PM = 0.001$ and $DM = 0.003$. We conducted 10 trials for each combination of the parameters B, L and we. This is because B and L are related to the richness of the environment and the basic ability of organisms, respectively.

Fig. 4 shows the average fitness over the all $(3 \times 3 \times 10)$ trials for each case of we. A horizontal axis represents we and the red square represents the average fitness. We used the fitness values of the last 1000 generations for calculating t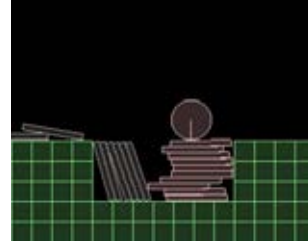he ave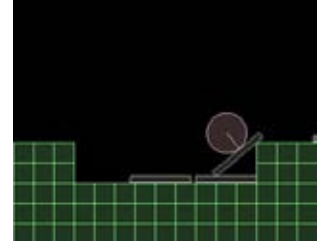rage fitness in each trial to eliminate effects of initial conditions. We also showed a box plot of each set of 90 trials. We see that the weathering probability we strongly affected the average fitness. There was a statistically significant difference in the fitness distribution among these cases (Kruskal-Wallis test, $H = 81.9$, $p - value < 1 \times 10^{-3}$).

Here, we particularly focus on the three typical cases of we: $0.01, 0.1$ and 1.0. In the case of the probability $we = 1.0$, there is no effect of the ecological inheritance because all objects vanish when they are handed over to next generation. On the other hand, the ecological inheritance occurs stably when the probability we is 0.01. When the probability we is 0.1, the ecological inheritance occurs but it is unstable.

In the case of $we = 1.0$, the average fitness was 1.8. It means that a virtual organism arrived at the goal one or two times on average in one trial. On the other hand, the average fitness was about 2.1 when we was 0.01. It indicates that the ecological inheritance of most of the objects from the parental generation contributed to the fitness increase. Comparing with this result, the average fitness was about 0.6 in the case of $we = 0.1$, which also indicates that unstable inheritance of objects can rather decrease the fitness. This implies that the degree of ecological inheritance has a nonlinear effect on the adaptivity of the population.

The high fitness values when $we = 0.01$ and 1.0 were obtained by the evolution of adaptive niche-constructing strategies. Fig. 5 and Fig. 6 show two typical strategies, which were commonly observed in successful trials irrespective of the parameter settings (except for $we = 0.1$). One is a valley-filling strategy that fills a valley with many objects
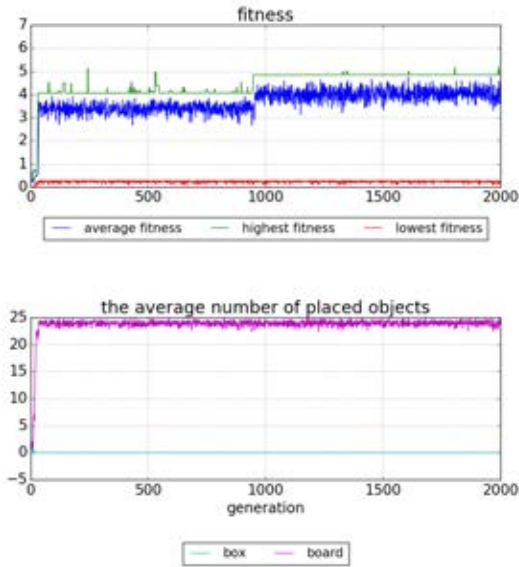
Figure 8: A typical example of evolution process when $we = 1.0$, $L = 500$ and $B = 25$.

(Fig. 5), which allows an organism to pass through the valley. The other is a ramp-placing strategy that creates a ramp of a board, which allows an organism to climb from the bottom of a valley. The former strategy was observed more often than the latter, which implies that the former was more easily acquired through the evolution process due to its simplicity and robustness against external perturbations such as collisions with organisms or other objects, compared with the latter. Fig. 7 shows an example of objects that worked as obstacles, preventing an organism from crossing a valley. We discuss how and why the degree of ecological inheritance affected the evolution of such niche-constructing behaviors in detail.

Experiments with no ecological inheritance
($we = 1.0$)

First, we analyze experimental results with the weathering probability $we = 1.0$ as a basic situation in which there is no effect of ecological inheritance. Fig. 8 shows an example evolution process of the fitness (top) and the average number of objects that existed at the end of fitness evaluation in each generation (bottom) in a trial when $L = 500$, $B = 25$ and $we = 1.0$, as a case in which an adaptive structure evolved successfully. While the average fitness was less than 1.0 in the initial generation, it increased to about 3.5 quickly. Then it further increased to about 4.0 at around the 950th generation. There were organisms with very low fitness through experiments. This is because offspring of organisms with higher fitness sometimes cannot reach a goal at all due to the negative effects of genetic operations. The average num-
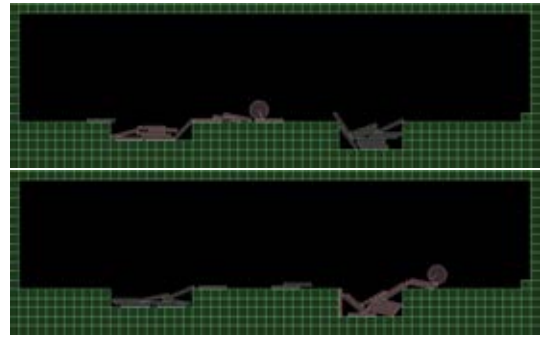

Figure 9: An emerged structure at the 900th (top) and the 2000th (bottom) generations in the trial in Fig. 8.

ber of the placed boards was around 24, which is close to the number of available objects B in this case, whereas that of placed boxes was 0. Fig. 9 shows an emerged structure at the 900th (top) and the 2,000th (bottom) generations in the trial in Fig. 8. We see that the organism could pass through valleys by using both valley-filling and ramp-placing strategies with many boards. We also see that the slight changes in the distribution of boards contributed to the fitness increase as mentioned above.

So as to analyze the general tendency of emerged niche-constructing behaviors and their adaptivity, we focus on the average fitness and the average number of the two types of placed objects in various experimental conditions of B and L, as shown in Fig. 10. Each point represents the average fitness (color) and the average number of placed boxes (x-axis) and boards (y-axis) during a fitness evaluation process over the last 1000 generations in a single trial. Each subfigure corresponds to the setting of B and each type of marker corresponds to the setting of L. We see that the trials in which many objects were placed tended to have the higher fitness, as the example above showed. This indicates that the niche construction, that is, placing many objects, contributed to the adaptivity of the evolved organisms in our model. We also see the virtual organism never arrived at the goal in some trials and thus got a low fitness. In this case, the organisms tended to evolve non-niche-constructing strategies, which do not place any objects at all, because it is better not to place any obstacles in order to get closer to the goal if objects do not contribute to pass through valleys. This strategy is expected to be a sub-optimal in the sense that once such a strategy occupied the population, adaptive niche-constructing strategies rarely evolved.

Next, we focus on the types of the placed objects in the field. It is seen from Fig. 10 that the number of the placed boards was larger than that of placed boxes especially when the fitness was high. The difference in the characteristic of boxes and boards appears to be the cause of this. A box is taller than a board. Thus, it is beneficial to use for filling in valleys. However, at the same time, it can become an
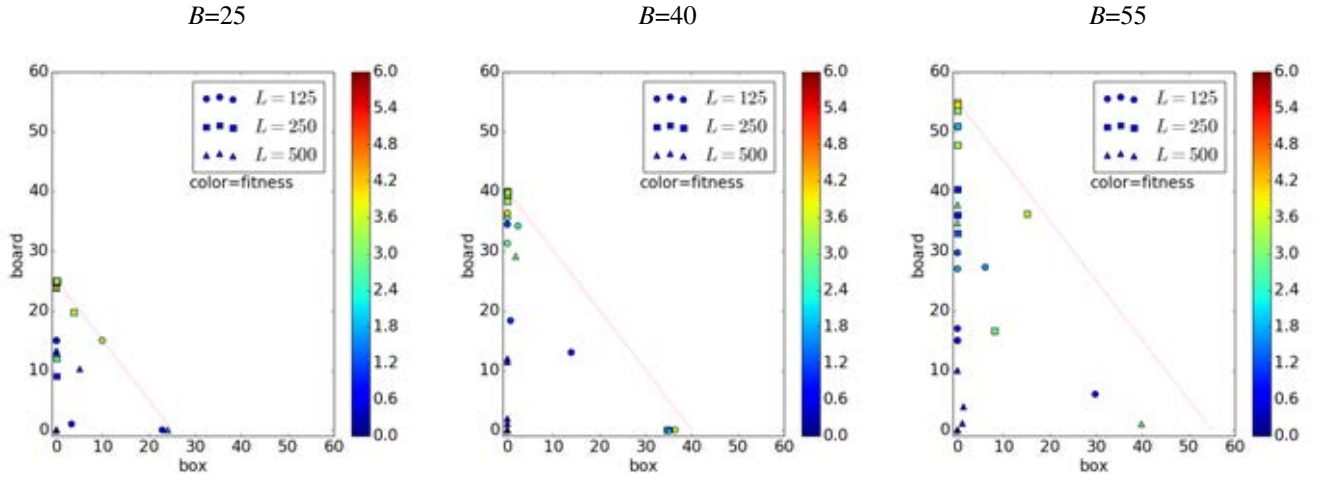
Figure 10: The fitness and the number of the objects that existed at the end of fitness evaluation when $we = 1.0$.
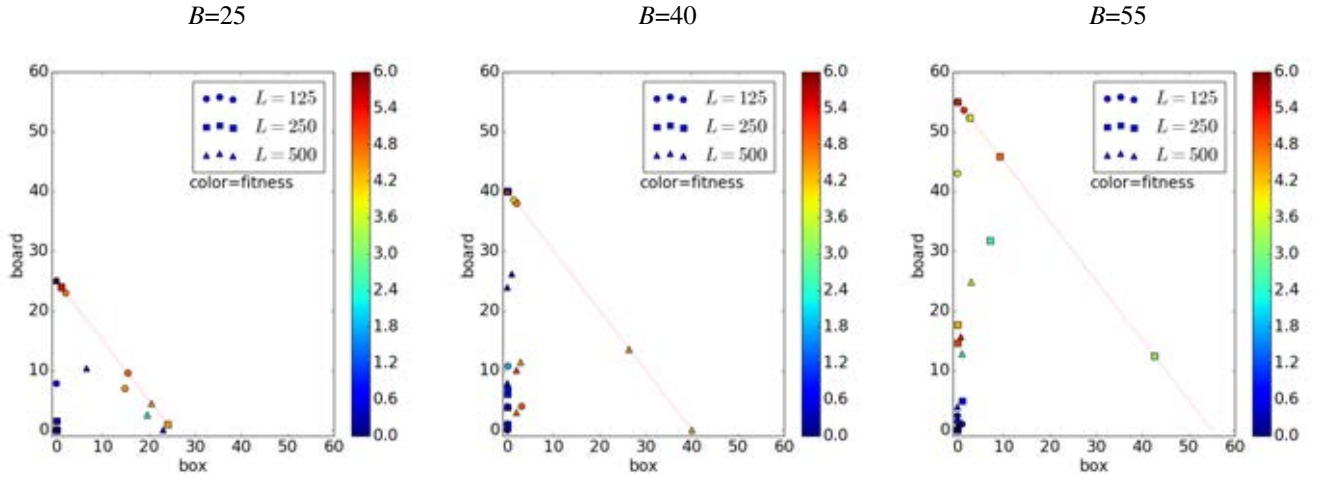

Figure 11: The fitness and the number of the objects that existed at the end of fitness evaluation when $we = 0.01$.

obstacle if it exists in front of an organism. On the other hand, a board is flatter than a box and a virtual organism can climb over it, thus, it does not cause such a problem, which allows organisms to obtain the higher fitness when they use boards. Furthermore, it is long enough to create a ramp structure. It is expected that boards were used more frequently than boxes because of these differences in their characteristic. In addition, we also see that the fitness of the trials with $B = 25$ was higher than one with $B = 40$ and 50 in general. This might be because objects tend to become obstacles if too many objects are placed in the field.

Experiments with stable ecological inheritance ($we = 0.01$)

Next, we discuss how the evolution of such an adaptive niche construction is affected if constructed structures are inherited to the next generation. Fig. 11 shows the relation between the fitness and the number of the objects that existed at the end of fitness evaluation in the case of stable ecological inheritance: $we = 0.01$. The overall trend did not change compared with Fig. 10, meaning that adaptive organisms tended to have many objects in their field. However, in this case, they inherited most of the objects due to the very low weathering rate, and they tended to add a few boards during their fitness evaluation process. The fitness tended to be higher than the cases with $we = 1.0$, especially when B was large (40 and 55).

Fig. 12 shows a typical example of an evolution process in the case of $L = 125$ and $B = 40$ in which the adaptive structure evolved successfully. The top, middle and bottom panels represent the fitness, the average number of the inherited objects from the previous generation and the average number of the placed objects by an organism at the current generation, respectively. Except for the initial few generations, organisms evolved to place a few boards at each generation, which resulted in the accumulation of many boards in the field. At the last generation, on average, 38.1 boards
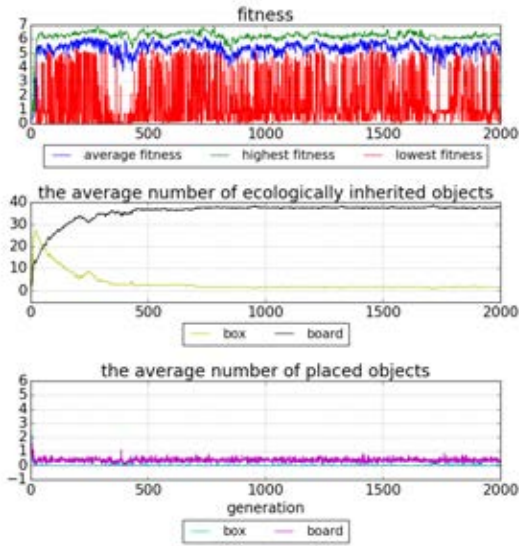
Figure 12: A typical example of an adaptive evolution process in the case of $we = 0.01$, $L = 125$ and $B = 40$.
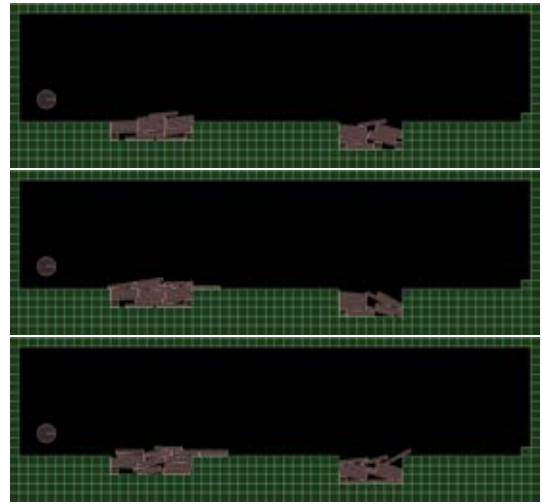

Figure 13: A typical example of an inherited environment at the 300th (top), 400th (middle) and 500th (bottom) generations in the case of $we = 0.01$, $L = 125$ and $B = 40$.

and 1.5 boxes were inherited from the previous generation, while 0.3 boards and 0.02 boxes were placed in the field. This means that organisms inherited the nearly maximum number of boards from their parents, and they compensated for the vanished ones by placing a few additional boards. The average fitness reached around 5.0 and this was higher than that of any cases with the probability $we = 1.0$.

Fig. 13 shows snapshots of the inherited environment in the same trial in Fig. 12. The top, middle and bottom panels represent the inherited environments at the 300th, 400th and 500th generation, respectively. We see that two valley-filling structures of boards were inherited. These structures allowed an organism to pass through the valleys and this is a main reason that this organism obtained the high fitness because it does not need to create such adaptive structures from scratch. We also see that there were a few changes in these structures across generations. This is due to the weathering of a few objects, and the organism maintained adaptive structures by placing additional boards as many as possible in the field. The reason the fitness tended be high especially when B was large is expected to be due to the fact that well-organized structures with many objects through ecological inheritance were more adaptive (e.g., easy to pass through, robust against the weathering of objects). Therefore, in the case of stable ecological inheritance, organisms evolved to maintain an inherited adaptive structure composed of many objects.

Experiments with unstable ecological inheritance ($we = 0.1$)

In this condition, in which 10% of the objects in the previous generation disappear, the fitness was very small in many trials, and the non-niche-constructing strategy evolved in such cases. A cause of it is expected to be large changes in the environmental conditions between generations. Even when adaptive niche-constructing strategies appear and begin to invade into the population, the emerged adaptive structures in the current generation tend to become obstacles (such as shown in Fig. 7) in the subsequent generations because of their irregular shapes due to the high weathering rate. This prevents such adaptive strategies to invade into the population, and further allows non-niche-constructing strategies to evolve.

Fig. 14 shows a typical but a bit complex example of such a situation when $L = 125$, $B = 40$ and $we = 0.1$. The fitness was high around the initial few generations, meaning that an adaptive niche-constructing strategy evolved. However, the fitness decreased drastically as soon as the organism began not to place objects, and it never increased until the last generation.

Once the adaptive structures emerged and inherited to the next generation, placing more objects might not contribute to the fitness increase or even have a negative effect on the fitness increase. In such a case, there can be no or negative selection pressure to place objects. However, if non-niche-constructing strategies evolved, the adaptive structure became obstacles very quickly in the case of the high weathering rate, which resulted in the rapid fitness decrease, as observed in Fig. 14.

In sum, the unstable ecological inheritance has a negative effect on the evolution of an adaptive niche-constructing be-
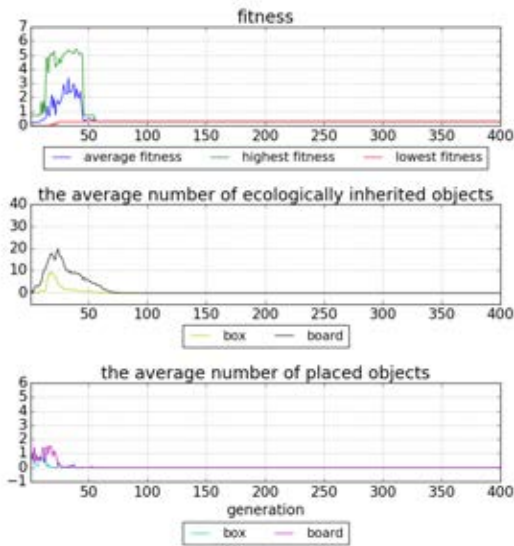
Figure 14: A typical example of an evolution process in the case of $we = 0.1$, $L = 125$ and $B = 40$.

havior by collapsing emerged adaptive niches.

Conclusion

In order to clarify the evolutionary dynamics of physical niche-constructing behaviors, we constructed an evolutionary and physically-grounded model of virtual organisms in which an organism has to arrive at the goal by performing a physical niche construction by placing objects in a two-dimensional field with valleys.

The results showed that the degree of ecological inheritance, which is represented as a weathering probability of inherited objects from a parent to its offspring, has a nonlinear effect on the adaptivity of the population. In the case of no ecological inheritance, niche-constructing behaviors such as valley-filling or ramp-placing strategies emerged. Furthermore, the stable inheritance of constructed structures from a parent contributed to the adaptivity of the population by allowing an organism to maintain the inherited and adaptive structures. On the other hand, when constructed structures were unstably inherited, the adaptive structures at a generation tended to become non-adaptive one at subsequent generations. This prevents the niche-constructing behavior from evolving and makes the adaptivity of the population lower. We think that these effects of ecological inheritance on the evolution of niche-constructing behaviors might be one of the typical properties of physical and complex niche construction.

The environmental state in Laland et al's model of population genetics (Laland et al., 1996) is represented as the amount of resource, which can be directly increased by the niche-constructing behavior of organisms. They focused on the number of previous generations of niche construction influencing the amount of resource in the current generation, and showed that the increase in the number of previous generation have a simple and monotonous effect on the evolution process, yielding the more considerable time-lag. In contrast, our result implies that such environmental parameters can have more complex effects on the evolution process when there are more complex interactions between organisms and environments.

Future work includes introducing other types of objects and the evolution of object shapes into our model, and conducting evolutionary experiments with different setting of the field.

References

Catto, E. (2016). Box2d. http://box2d.org/.

Han, X., Hui, C., and Zhang, Y. (2009). Effects of time-lagged niche construction on metapopulation dynamics and environmental heterogeneity. *Applied Mathematics and Computation*, 215:449–458.

Harvey, I. (2004). Homeostasis and rein control: from daisyworld to active perception. In Pollack, J. Bedau, M. A., Husbands, P., Watson, R. A., and Ikegami, T., editors, *Proceedings of Artificial Life IX*, pages 309–314. MIT Press, Cambridge, MA.

Ito, T., Pilat, M. L., Suzuki, R., and Arita, T. (in press). Population and evolutionary dynamics based on predator-prey relationship in 3D physical simulation. *Artificial Life*, 22(2).

Kojima, T., Suzuki, R., and Arita, T. (2014). Effects of ecological inheritance on coevolution of cooperative behaviors and physically niche constructing behaviors. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 18(3):391–400.

Laland, K. N., Odling-Smee, F. J., and Feldman, M. W. (1996). The evolutionary consequences of niche construction: a theoretical investigation using two-locus theory. *Journal of Evolutionary Biology*, 9:293–316.

Odling-Smee, F. J., Laland, K. N., and Feldman, M. W. (2003). *Niche Construction: The Neglected Process in Evolution*. Princeton University Press, Princeton, NJ.

Suzuki, R. and Arita, T. (2010). Effects of temporal locality of ecological processes on coevolution of learning and niche construction. In Fellermann, H., Dörr, M., Hanczy, M. M., Laursen, L. L., Maurer, S., Merkle, D., Monnard, P., Støy, K., and Rasmussen, S., editors, *Proceedings of Artificial Life XII (ALIFEXII)*, pages 471–477. MIT Press, Cambridge, MA.

Taylor, T. (2004). Niche construction and the evolution of complexity. In Pollack, J. Bedau, M. A., Husbands, P., Watson, R. A., and Ikegami, T., editors, *Proceedings of Artificial Life IX*, pages 375–380. MIT Press, Cambridge, MA.

Taylor, T. (2015). Requirements for open-ended evolution in natural and artificial systems. In *The Online Extended Abstract of the Satellite workshop "EvoEvo Workshop" of the 13th European Conference on Artificial Life (ECAL 2015)*, pages 1–8. arXiv:1507.07403.

The Effects of Evolution and Spatial Structure on Diversity in Biological Reserves

Emily L. Dolson[1,2,3], Michael J. Wiser[1,3] and Charles Ofria[1,2,3]

[1]BEACON Center for the Study of Evolution in Action
[2]Computer Science and Engineering, Michigan State University, East Lansing, MI, USA
[3]Ecology, Evolutionary Biology, and Behavior, Michigan State University, East Lansing, MI, USA
dolsonem@msu.edu

Abstract

Conservation ecologists have long argued over the best way of placing reserves across an environment to maximize population diversity. Many have studied the effect of protecting many small regions of an ecosystem vs. a single large region, with varied results. However, this research tends to ignore evolutionary dynamics under the rationale that the spatiotemporal scale required is prohibitive. We used the Avida digital evolution research platform to overcome this barrier and study the response of phenotypic diversity to eight different reserve placement configurations. The capacity for mutation, and therefore evolution, substantially altered the dynamics of diversity in the population. When mutations were allowed, reserve configurations involving a greater number of consequently smaller reserves were substantially more effective at maintaining existing diversity and generating new diversity. However, when mutations were disallowed, reserve configuration had little effect on diversity generation and maintenance. While further research is necessary before translating these results into policy decisions, this study demonstrates the importance of considering evolution when making such decisions and suggests that a larger number of smaller reserves may have evolutionary benefits.

Introduction

Protecting biodiversity is generally acknowledged to be an important conservation goal for a number of reasons, including biodiversity's role in maintaining various ecosystem services (e.g. carbon sequestration), and its potential as a reservoir of useful and undiscovered genetic innovations (Gaston and Spicer, 2004; Hassan et al., 2005; Loreau et al., 2001; Montoya et al., 2012). However, there is another reason that biodiversity is critically important, which is often overlooked – continued evolution requires diversity. Since adaptation to new environments will be a critical component of the long-term survival of many lineages in the face of climate change, it is important to consider conservation of biodiversity in the context of evolution (Stockwell et al., 2003; Mace and Purvis, 2008; Smith et al., 2014).

Most conservation biology research requires a broad spatial scale. Incorporating the long temporal scale required to study evolution makes this already challenging problem intractable in most cases. As a result, most attempts to factor evolution into conservation planning decisions have, out of necessity, been based on general evolutionary principles rather than empirical analysis of their likely outcomes (Cowling and Pressey, 2001; Sgro et al., 2011; Ferrire et al., 2004). In particular, little research on conservation schemes to date has taken evolution into account. Artificial Life techniques such as digital evolution have a lot of potential as an approach to overcoming these obstacles; they allow for the formation of interesting ecologies in a system with a fast enough generation time to do large scale evolution experiments.

Here, we use digital evolution to revisit the single-large vs. several small (SLOSS) debate from a perspective that incorporates evolutionary theory. The SLOSS debates emerged from the theory of island biogeography (MacArthur and Wilson, 1967). The original argument was that, since large islands have more species, larger reserves should be better for conserving biodiversity (Diamond, 1975). However, this effect is counterbalanced by the fact that placing more reserves might result in sampling from multiple different species pools (Simberloff and Abele, 1976). More recent refinements have considered the placement of the reserves relative to each other and interconnectivity between them (Saunders et al., 1991; Tjorve, 2010).

Evolutionary dynamics likely add additional weight to the argument for several small reserves for a number of reasons. First, transient fitness gains can result in a single lineage sweeping a reserve relatively quickly and wiping out standing diversity. Second, separating reserves decreases the colonization rate, giving other lineages time to gain beneficial mutations of their own (Whitley et al., 1998; Tomassini, 2005). Third, spatial isolation can increase the likelihood of speciation.

Many factors interact to bring about the complex spatial eco-evolutionary dynamics that we observe in biological ecosystems. Indeed, the interactions between various factors are a large part of the reason that the relative benefits of different reserve placement strategies have been so hard to untangle. Here, we seek only to lay the groundwork for addressing the impact of evolution on these questions. In

order to facilitate this, we will deal with the simplest possible case: a population of sessile, asexual organisms at the same trophic level. Movement, sexual recombination, and predation likely have dramatic impacts on the resulting dynamics. However, in order to understand these effects, we must first understand the behavior of a system without them. Additionally, for the purposes of this paper, we assume an entirely homogeneous environment, eliminating the possibility for complex interactions among habitat heterogeneity, species diversity, and reserve area (Kadmon and Allouche, 2007).

Methods

Study System

We conducted our experiments *in silico*, using the Avida Digital Evolution Platform version 2.12.4 (available at https://www.github.com/devosoft/avida) (Ofria and Wilke, 2004). Configuration files for this paper are available at https://github.com/emilydolson/conservationExperiment. The world of Avida is a two-dimensional grid of cells occupied by digital organisms. These organisms are actually computer programs; their genomes are sequences of simple computer instructions. At the beginning of the experiment, we seed the world with a single ancestor that contains the instructions necessary to copy itself. As organisms copy themselves, they periodically make mistakes, introducing mutations. Some of these mutations will improve the efficiency of self-replication, so the organisms that have them will copy themselves faster than the others. If there is no space available for an organism's offspring, the offspring will replace an existing organism. As a result, there is selection for organisms that can replicate themselves faster. Because there are mutation, inheritance, and selection, evolution by natural selection occurs.

To allow for the formation of more complex ecologies, we can also choose to reward organisms for performing various computational tasks by allowing them to execute their genomes faster. These tasks can be thought of as pathways for metabolizing various resources, and allow for different organisms to specialize on different survival strategies. To allow for the formation of a stable ecosystem, we can establish negative frequency dependence by linking each task to a limited resource, such that organisms are rewarded for a task in proportion to the amount of the relevant resource that they have access to (Chow et al., 2004).

Experimental Set-up

For this study, we started by evolving ten populations in the limited resource environment described above (specifically, we used the same configuration settings as in (Walker and Ofria, 2012) except where otherwise noted). Each population was started from the same hand-coded self-replicator, but was then allowed to diverge for 100,000 updates, a length of time roughly equivalent to 2000 generations. We

Reserve Configurations	Kill Rates
see Figure 1	100, 200, 300, 400, 500
Mutations Allowed?	**Initial Populations**
Yes, No	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Table 1: Parameter values. These values were combined in a full-factorial design, with 10 replicates per condition, for a total of 8000 runs of Avida. Kill rate indicates a number of grid cells that were randomly selected from the environment each update. If these cells were not in a reserve and contained an organism, that organism was killed

then placed these populations in one of eight environments for 100,000 more updates. Each environment had reserves placed across it in a different configuration. All environments had a total of 900 out of the 3600 grid cells placed in square reserves that tiled evenly across the environment. Reserve configurations varied from having many very small reserves to having a single very large reserve (see Figure 1). Because world size was held constant, configurations with more reserves necessarily involved them being placed closer together (there was always one reserve worth of space between reserves on all sides). Organisms living in areas outside of the reserves were at risk of being randomly killed each update (see Table 1). Offspring were placed probabilistically near their parents, according to a Poisson distribution, to create a spatial population structure roughly analogous to tree seed dispersal. In order to ascertain what effect allowing populations to evolve was having on our results, we also ran a series of controls in which mutations were disallowed for the second 100,000 updates. We ran 10 replicates per treatment in a fully factorial design across initial population, environment, five kill rates, and mutations being allowed vs. disallowed.

Analysis

There are three possible mechanisms by which our reserve schemes could drive changes in diversity over evolutionary time - a reserve configuration might: 1) sample from a different range of locations across the environment, 2) promote improved maintenance of existing diversity, and/or 3) promote improved generation of new diversity. To address possible mechanism (1), we measured the number of phenotypes in any reserve at the beginning of the experiment for each condition. To address possible mechanisms (2) and (3), we collected a variety of data on the phenotype-area relationships in our data: phenotype richness within each reserve, total phenotype richness captured across all reserves in a replicate, count of phenotypes lost over the experimental treatment, and count of new phenotypes that evolved over the course of the experimental treatment.

All analyses were conducted using the R Statistical Computing Language, version 3.2.3 (Team, 2013). For statistics in which the unit of replication was a single run of Avida
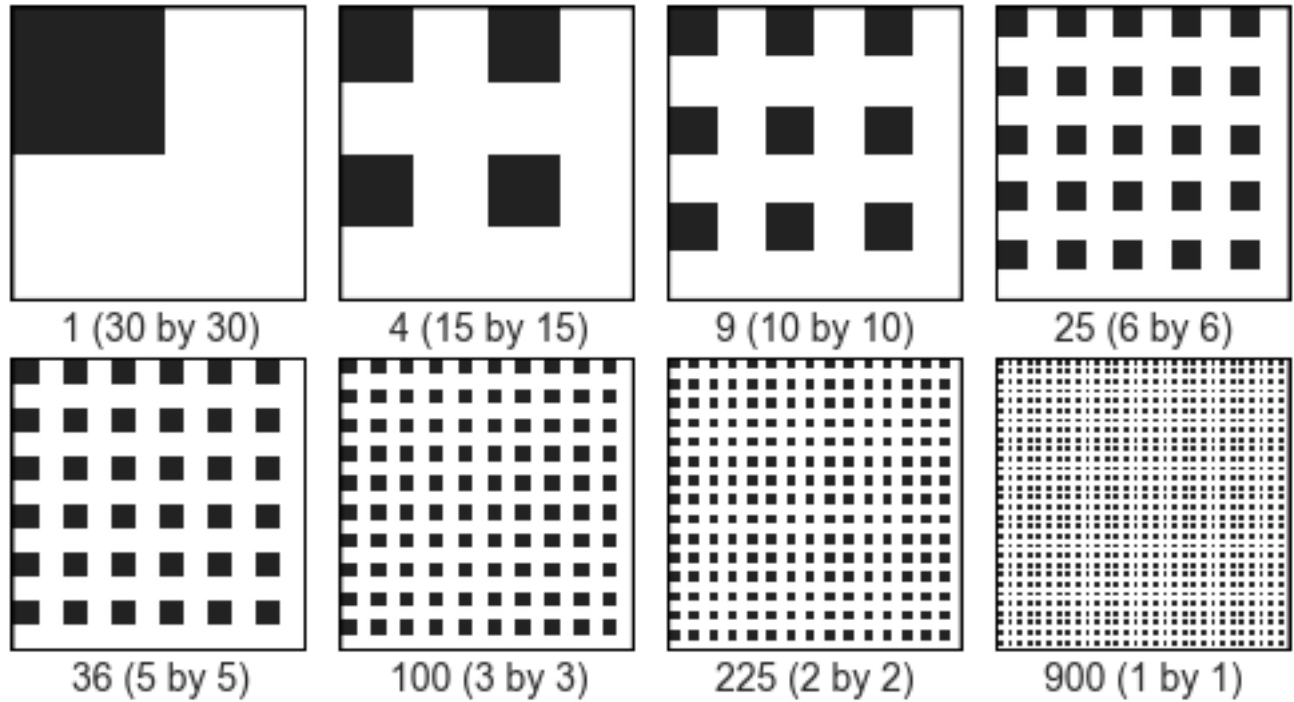
Figure 1: Reserve configurations. Black cells are part of reserves, while white cells are unprotected. Note that the world is toroidal, so spacing between all reserves within a condition is equivalent.

(phenotype loss and generation of novelty), we used a 2-way ANOVA in which initial population, reserve size, and their interaction were treated as random effects. Effect sizes were calculated as partial eta squared (Lakens, 2013). We also calculated some statistics (alpha richness) for which the unit of replication was individual reserves. To account for the non-independence that this set-up introduced, we used linear mixed models with random effects for initial population and replicate, as implemented in the lme4 R package (Bates et al., 2015).

Results and Discussion

Overall, the replicates in which mutations were allowed during the second 100,000 updates had substantially higher diversity (both richness and Shannon entropy) at the end of the experiment than replicates for which mutations were disallowed (see Figure 3). Runs in which mutations were disallowed had an average of 192.009 +- 1.345 fewer phenotypes remaining at the end than runs in which evolution was allowed to continue (Linear mixed model, Chi-squared = 10123, p <.0001). Most ecological models do not include this drop-off (MacArthur and Wilson, 1967; Kadmon and Allouche, 2007; Tjorve, 2010). This discrepancy is partially because ecological models of reserve placement are based on models of island biogeography, and so do not include a

phase prior to reserve placement. However, this is a superficial distinction. The more fundamental explanation is likely that most ecological models are built on the assumption of some sort of competitive equivalency between phenotypes, usually based off of the idea that all extant phenotypes are well-optimized to their environment. We make no such assumption. Instead, diversity in these experiments is stabilized through negative frequency dependence (due to limited resources, discussed above) and co-evolutionary arms races, resulting in a dynamic near-equilibrium. While these mechanisms are more realistic, they generally mean that advantages that one lineage has over another are unlikely to persist in the long-term. At any arbitrary point in time, there is probably a lineage with a slight advantage over other lineages. Removing mutations will eliminate the generation of novelty and leave this lineage at an advantage for the rest of the experiment. A related factor is the fact that diversity can only ever decrease if mutations are disallowed; a simple random walk under these conditions would also show some decrease in diversity.

Allowing ongoing mutations dramatically increased the extent to which a greater number of smaller reserves promote higher final phenotypic richness than configurations with a smaller number of larger reserves (see Figure 3). For the runs in which mutations were allowed during the sec-
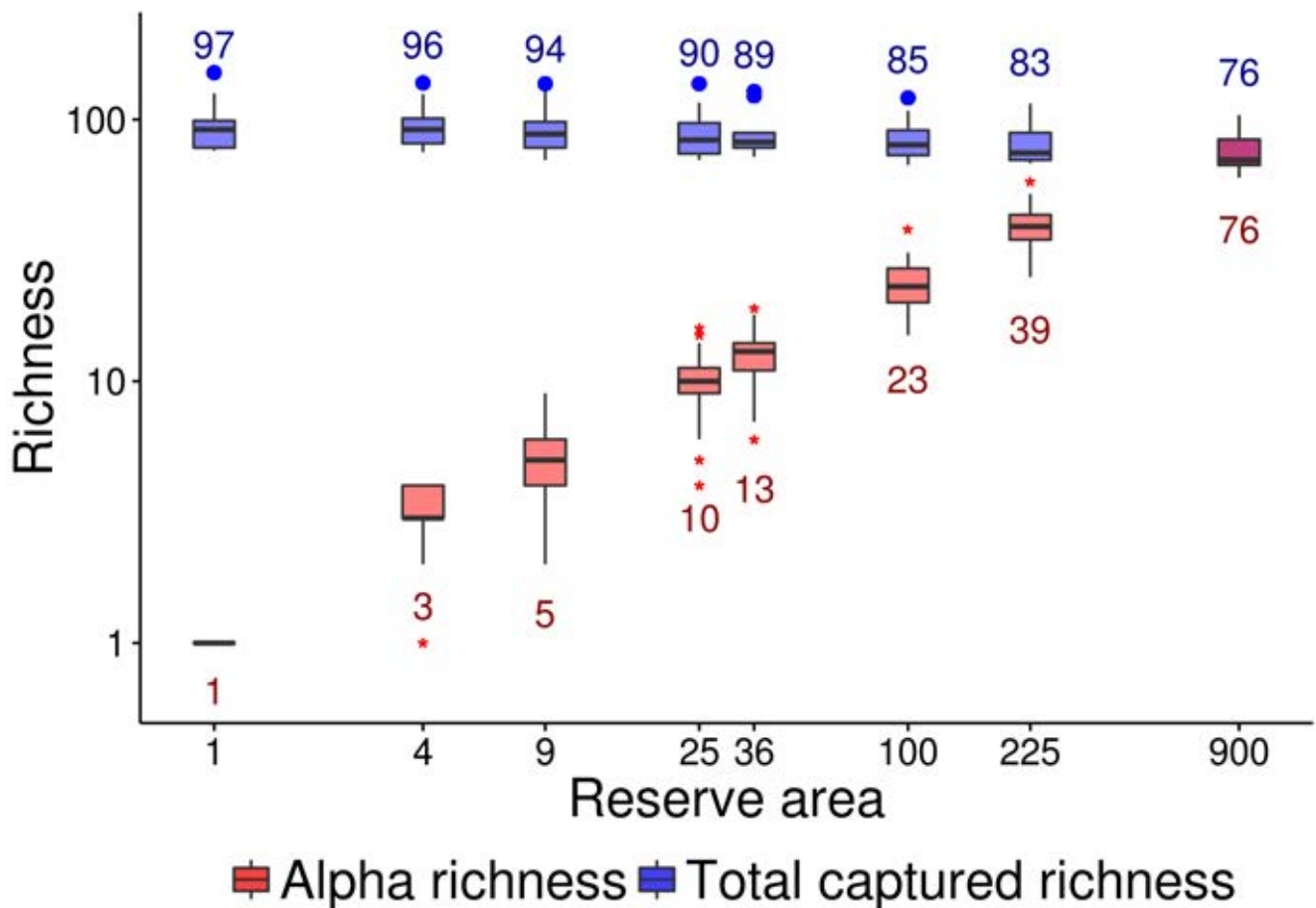
Figure 2: Richness captured within and across reserves at the start of the experiment. Red boxes show the number of phenotypes captured within each reserve across all runs. Blue boxes show the total number of phenotypes captured across all reserves. Note that, despite the clear positive relationship between the size of a reserve and its phenotypic richness, the highest total richness across reserves is achieved in configurations with small reserves. Both axes are on a log scale and numeric labels indicate means of each box.

ond half, increasing the log of the number of reserves by one increased the log of the final number of phenotypes by approximately .0956 +- .00259 (Linear mixed model, Chi-squared = 1172.6, p <.0001). This effect was substantially weaker among runs where mutations were disallowed, although still significantly different from 0 (Linear mixed model, Chi-squared = 25.298, p <.0001, slope=0.0084 +- 0.0017 log(phenotype count) per additional log(reserve count) units). This appears to be the result of a combination of the three mechanisms described in the Methods section.

More/smaller reserves capture more phenotypes

Configurations with a greater number of consequently smaller reserves captured a greater number of phenotypes in reserves, likely due to the substantial clumping of closely related organisms (high spatial autocorrelation). In the ten

initial populations, the relationship between phenotype richness within a reserve and the size of that reserve followed the pattern of a standard species-area relationship (see Figure 2), with a positive linear relationship between the logarithms of reserve size and reserve richness (Connor and McCoy, 1979). Despite this positive relationship, the total phenotypic richness summed across all reserves within an initial environment was negatively correlated with the area of each of those reserves. This negative relationship strengthened when populations were allowed to evolve for 100,000 updates in the reserve design, while the slope of the positive relationship between within-reserve richness and reserve area decreased slightly (see Figure 3). These effects both weakened dramatically when mutations were disallowed, but remained significantly different from zero. This ability for many small reserves to sample across multiple
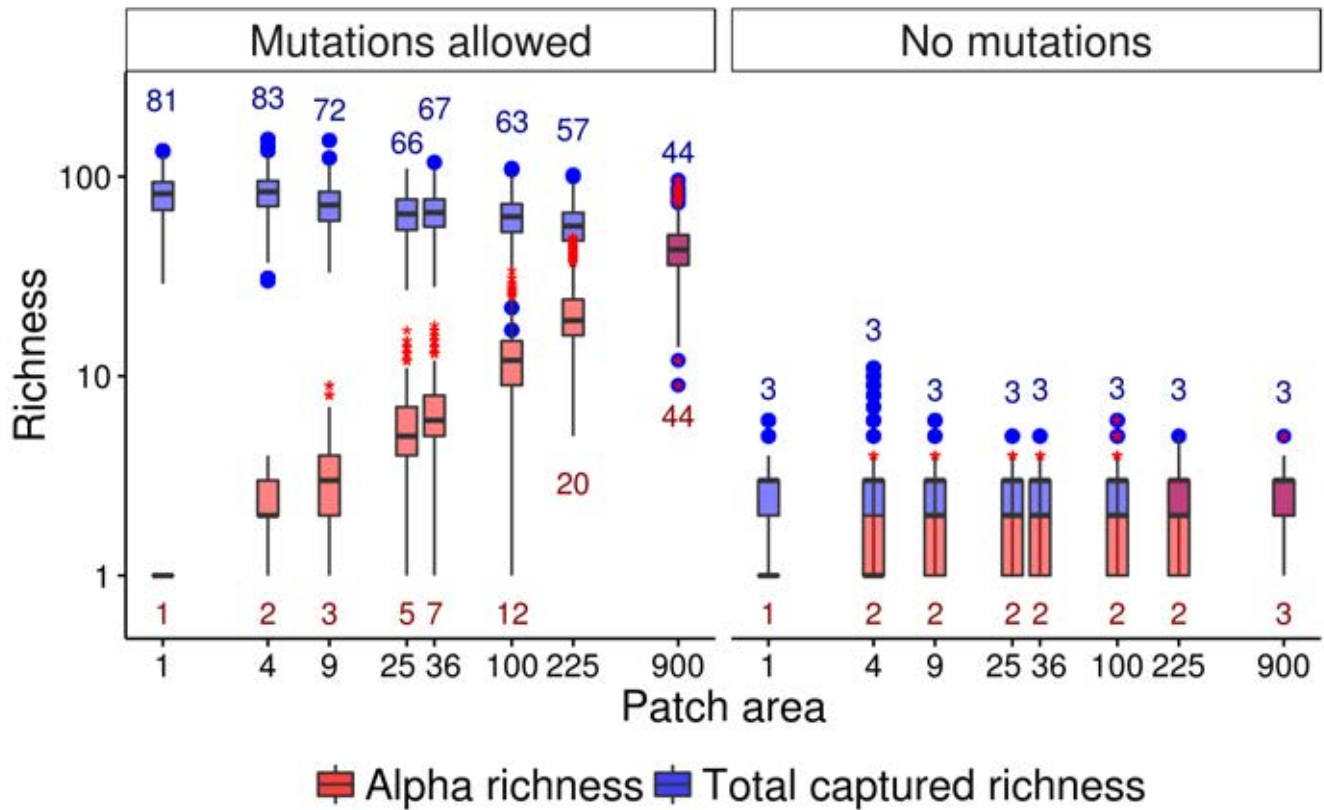
437

Figure 3: Richness within and across reserves at the end of the experiment for both conditions. Note that the negative relationship between reserve area and total captured richness has intensified since the beginning of the experiment for runs in which mutations are allowed. Both axes are on a log scale and numeric labels indicate means of each box.

species pools is precisely the scenario that Simberloff and Abele initially brought up as a counterexample to the argument that a smaller number of larger reserves was always preferable (Simberloff and Abele, 1976). Because organisms disperse locally, similar phenotypes are likely to be clumped together in space, effectively creating multiple species pools.

More/smaller reserves promote diversity maintenance

We measured the number of phenotypes that were present in the initial population but no longer present in the final population, i.e. lost phenotypes (see Figure 4). When mutations were allowed, there was a strong positive relationship between phenotype loss and reserve sizes - environments with larger reserves resulted in greater phenotype extinction by the end of the experiment (two-way ANOVA, $F(1,3980) = 1680.26$, p $<.0001$, $\eta_p^2=.296$). Among the runs where mutations were disallowed, however, there was no significant effect of reserve configuration on phenotype loss (two-way ANOVA, $F(1, 3980) = 1.17$, p $= .41$). This discrepancy is

an example of the vast impact that allowing for evolutionary dynamics can have.

There are a number of potential evolutionary drivers behind this effect, mostly related to the dynamics of selective sweeps in the population. In reproducing populations, some individuals will have more offspring than others. When one genotype has a substantial selective advantage over its competitors, selection will favor a rapid increase in that genotype's relative frequency, driving competitors in the population toward extinction. Such a process is referred to as a selective sweep, as selection sweeps less-fit variants out of the population (McVean, 2007). Selective sweeps typically dramatically reduce diversity within the affected population. Not only will the region of the genome under selection go to fixation in the population, but other variant sites on the genetic background where the beneficial trait first arose may also fix. Selective sweeps may be incomplete if, for example, a lineage encounters a competitor that is too similar in fitness. An incomplete selective sweep may also occur if a competitor exhibits negative frequency dependence and thus reaches an equilibrium. Even incomplete selective sweeps
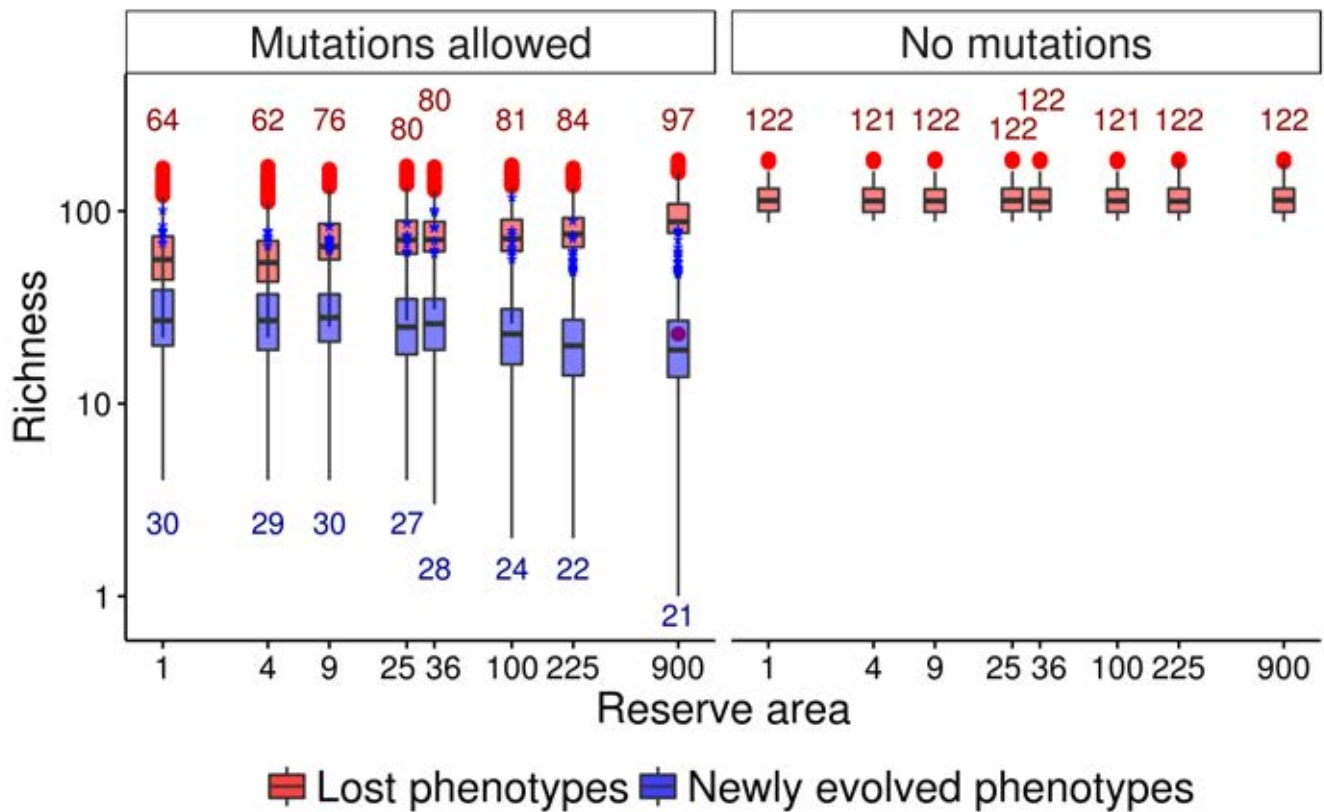
438

Figure 4: Diversity maintenance and generation across conditions. Red boxes show the number of phenotypes that were initially captured in reserves but were not present at the end of the experiment. Blue boxes show the number of phenotypes that were not initially captured in reserves but were present at the end of the experiment. Note the positive correlation between phenotype loss and reserve size when mutations are allowed. Both axes are on a log scale and numeric labels indicate means of each box.

can substantially reduce diversity, by making a large fraction of the population identical, and driving competing variants extinct (Biswas and Akey, 2006).

Selective sweeps occur faster in populations with higher strength of selection and spatial connectivity (Cantu-Paz, 2001). Both of these factors are impacted by reserve placement; large unprotected regions that fall between reserves decrease connectivity between those reserves (see Figure 5), effectively breaking them into multiple subpopulations and thereby decreasing strength of selection (Gavrilets and Vose, 2005). In this experiment, inter-reserve connectivity and reserve size are inversely correlated because we are holding the size of the entire world constant. As a result, it is hard to determine the impact of each of these variables on the dynamics of selective sweeps in this experiment. Based on the empirical results, however, the effect of reduced subpopulation size would appear to be stronger than the effect of increased connectivity.

The initial population had a significant effect on diversity maintenance, both when mutations were allowed (two-way

ANOVA, $F_{(9, 3980)} = 1602.45$, p $<.0001$) and disallowed (two-way ANOVA, $F_{(9, 3980)} = 204019.41$, p $<.0001$). When mutations were allowed, there was also a significant interaction between reserve configuration and initial population (two-way ANOVA, $F_{(9, 3980)} = 7.99$, p $<.0001$). The interaction term is likely significant because some reserve configurations happen to capture more of the initial diversity of any given initial population than others. There may also be an effect of some initial populations having higher starting diversity, stability against selective sweeps, or evolutionary potential than others.

More/smaller reserves promote diversity generation

We measured the number of phenotypes that were not present in the initial population but were present in the final population, i.e. newly evolved phenotypes (see Figure 4). Among the runs where mutations were allowed, the count of newly evolved phenotypes had a negative relationship with reserve size - environments with larger reserves resulted in
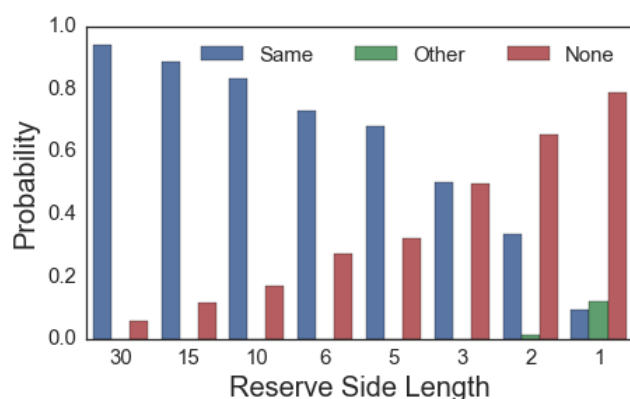
Figure 5: Reserve connectivity increases as patch size decreases. Bars show percentage of offspring from an arbitrary focal reserve that end up in the same reserve (blue), a different reserve (green), and no reserve (red). Note that reproduction events where a parent from one reserve has offspring in another reserve are incredibly rare in all reserve configurations except the two in which reserves are smallest.

fewer newly evolved phenotypes in total (two-way ANOVA, $F(1, 3980) = 278.47$, $p < .0001$, $\eta_p^2 = .065$). This relationship suggests that smaller reserves do a better job of generating new diversity. While the effect is much weaker than the effect of smaller reserves on diversity maintenance, it is still substantial (Lakens, 2013). As with diversity maintenance, it is likely that this effect is driven by dynamics related to selective sweeps. When organisms are competing against fewer other organisms for space, as is the case in a smaller reserve (but see Figure 5), the strength of selection is weakened. This is generally believed to allow time for evolutionary innovation and increased diversification into new niches. A configuration with many small reserves can be thought of as roughly equivalent to an evolutionary algorithm that maintains multiple sub-populations and allows occasional migration between them. Such algorithms are generally quite efficient, perhaps due to their improved ability to maintain and generate diversity (Tomassini, 2005).

Conclusions

We have laid groundwork for an integration of evolutionary dynamics into reserve design. Our results suggest that evolution fundamentally changes the way that reserve placement affects diversity. In the presence of evolution, configurations with a greater number of consequently smaller reserves substantially improved diversity maintenance. However, in the absence of evolution, there was no effect of reserve configuration on maintenance of the captured initial diversity. Configurations with a greater number of smaller reserves also promoted the evolution of a larger number of novel phenotypes, a dynamic that is only possible when mutations are allowed. These results extend prior research on island

model genetic algorithms to address a multi-niche ecosystem, which provides a better analog to biological ecosystems where conservation applications are relevant.

While our results have potentially important implications for conservation management decisions, it is important to recognize that a number of our simplifying assumptions will bias our results against a smaller number of larger reserves. None of the organisms in our experiment are, for instance, larger or at a higher trophic level than any other organisms. In nature, some organisms require vastly more space than others in order to have access to sufficient energy sources. Additionally, as all of our organisms are sessile, they are not at risk of wandering out of small reserves, as mobile organisms would be. Similarly, as the organisms considered here are asexual, factors such as inbreeding depression and Allee effects are not accounted for. This research represents the first step in an understanding of how evolution interacts with reserve placement. Other topics for future research include the effect of corridors, distance between reserves, placement of reserves in relation to spatial resources, interactions with motile organisms, and the impact of sexual recombination and gene flow on diversity in these systems.

Acknowledgements

References

Bates, D., Machler, M., Bolker, B., and Walker, S. (2015). Fitting Linear Mixed-Effects Models Using lme4. *Journal of Statistical Software*, 67(1):1–48.

Biswas, S. and Akey, J. M. (2006). Genomic insights into positive selection. *Trends in Genetics*, 22(8):437–446.

Cantu-Paz, E. (2001). Migration Policies, Selection Pressure, and Parallel Evolutionary Algorithms. *Journal of Heuristics*, 7(4):311.

Chow, S. S., Wilke, C. O., Ofria, C., Lenski, R. E., and Adami, C. (2004). Adaptive Radiation from Resource Competition in Digital Organisms. *Science*, 305(5680):84–86.

Connor, E. F. and McCoy, E. D. (1979). The Statistics and Biology of the Species-Area Relationship. *The American Naturalist*, 113(6):791–833.

Cowling, R. M. and Pressey, R. L. (2001). Rapid plant diversification: Planning for an evolutionary future. *Proceedings of the National Academy of Sciences*, 98(10):5452–5457.

Diamond, J. M. (1975). The island dilemma: Lessons of modern biogeographic studies for the design of natural reserves. *Biological Conservation*, 7(2):129–146.

Ferrire, R., Dieckmann, U., and Couvet, D. (2004). *Evolutionary conservation biology*, volume 4. Cambridge University Press.

Gaston, K. J. and Spicer, J. I. (2004). *Biodiversity: an introduction*. Blackwell Publishing, Malden, Mass.

Gavrilets, S. and Vose, A. (2005). Dynamic patterns of adaptive radiation. *Proceedings of the National Academy of Sciences of the United States of America*, 102(50):18040–18045.

Hassan, R., Scholes, R., and Ash, N. (2005). Ecosystems and human well-being: current state and trends: findings of the Condition and Trends Working Group. *Millennium Ecosystem Assessment*.

Kadmon, R. and Allouche, O. (2007). Integrating the effects of area, isolation, and habitat heterogeneity on species diversity: a unification of island biogeography and niche theory. *The American Naturalist*, 170(3):443–454.

Lakens, D. (2013). Calculating and reporting effect sizes to facilitate cumulative science: a practical primer for t-tests and ANOVAs. *Frontiers in Psychology*, 4:863.

Loreau, M., Naeem, S., Inchausti, P., Bengtsson, J., Grime, J. P., Hector, A., Hooper, D. U., Huston, M. A., Raffaelli, D., Schmid, B., and others (2001). Biodiversity and ecosystem functioning: current knowledge and future challenges. *science*, 294(5543):804–808.

MacArthur, R. H. and Wilson, E. O. (1967). *The Theory of Island Biogeography*. Princeton University Press.

Mace, G. M. and Purvis, A. (2008). Evolutionary biology and practical conservation: bridging a widening gap. *Molecular Ecology*, 17(1):9–19.

McVean, G. (2007). The Structure of Linkage Disequilibrium Around a Selective Sweep. *Genetics*, 175(3):1395–1406.

Montoya, D., Rogers, L., and Memmott, J. (2012). Emerging perspectives in the restoration of biodiversity-based ecosystem services. *Trends in ecology & evolution*, 27(12):666–672.

Ofria, C. and Wilke, C. O. (2004). Avida: A Software Platform for Research in Computational Evolutionary Biology. *Artificial Life*, 10(2):191–229.

Saunders, D. A., Hobbs, R. J., and Margules, C. R. (1991). Biological Consequences of Ecosystem Fragmentation: A Review. *Conservation Biology*, 5(1):18–32.

Sgro, C. M., Lowe, A. J., and Hoffmann, A. A. (2011). Building evolutionary resilience for conserving biodiversity under climate change. *Evolutionary Applications*, 4(2):326–337.

Simberloff, D. S. and Abele, L. G. (1976). Island biogeography theory and conservation practice. *Science*, 191(4224):285–286.

Smith, T. B., Kinnison, M. T., Strauss, S. Y., Fuller, T. L., and Carroll, S. P. (2014). Prescriptive Evolution to Conserve and Manage Biodiversity. *Annual Review of Ecology, Evolution, and Systematics*, 45(1):1–22.

Stockwell, C. A., Hendry, A. P., and Kinnison, M. T. (2003). Contemporary evolution meets conservation biology. *Trends in Ecology & Evolution*, 18(2):94–101.

Team, R. C. (2013). *R: A Language and Environment for Statistical Computing*. Vienna, Austria.

Tjorve, E. (2010). How to resolve the SLOSS debate: Lessons from species-diversity models. *Journal of Theoretical Biology*, 264(2):604–612.

Tomassini, M. (2005). *Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time*. Natural Computing Series. Springer Berlin Heidelberg, Berlin, Heidelberg.

Walker, B. L. and Ofria, C. (2012). Evolutionary Potential is Maximized at Intermediate Diversity Levels. pages 116–120. MIT Press.

Whitley, D., Rana, S., and Heckendorn, R. B. (1998). The Island Model Genetic Algorithm: On Separability, Population Size and Convergence. *Journal of Computing and Information Technology*, 7:33–47.

Modeling the Evolution of Mimicry

Mohiul Islam[1] and Peter Grogono[1]

[1]Concordia University
moh_i@encs.concordia.ca
grogono@cse.concordia.ca

Abstract

A novel agent based, artificial life model, for the evolution of mimicry is presented. This model is a predator-prey co-evolution scenario where pattern representation phenotype is simulated with Cellular Automata (CA), while behaviors of pattern recognition is configured with Hopfield Network. A visual three dimensional toroidal cube is used to construct a universe in which agents have complete freedom of mobility, genetic representation of behavior and reproduction capability to evolve new behaviors in successive generations. These agents are classified into categories of predator and prey species. Genome of prey species control their mobility and palatability, while 2D CA is used to represent a pattern, where the rule to generate the CA is also genetically represented. Through evolution, successive generations of prey species develop new patterns to represent them both visually and to the predators. Predators are agents with the primary purpose of providing selection pressure for the evolution of mimicry. They are equipped with Hopfield Network memory to recognize new CA pattern and make intelligent decisions to consume the prey based on their level of palatability. Using the above construction of ideas, successful emulation of the natural process of mimicry is achieved. Also complex behavior pattern of Batesian and Mullerian mimicry is simulated and studied.

Introduction

Mimicry is a process of deception. It is an evolutionary process with the help of which organisms survive by deceiving its predator. But this deception happens only if the environment contains similar appearing noxious organisms which the predators find unpalatable. Palatable organisms mimic the unpalatable ones through the process of evolution for survival of its species. The objective of this paper is to present an agent based artificial life model for simulating this natural process of the evolution of mimicry.

According to Langton, Artificial Life is '*Life made by Man rather than by Nature*'. Taylor also defines it as a tool for biological inquiry (Taylor and Jefferson, 1993). While providing a brief survey over different AL models he talks about *Wetware systems* which work at the molecular level, *Software systems* which work at the cellular level and *Hardware systems* which works at the organism level.

The initial contribution of *software systems* in artificial life was from John von Neumann when he designed the first artificial-life model (without referring to it as such), the famous self-reproducing, computation-universal cellular automata (Von Neumann, 1966). He tried to understand the fundamental properties of living systems, especially self-reproduction and the evolution of complex adaptive structures, by constructing simple formal systems that exhibit those properties.

Being a special case of complex systems, Complex Adaptive System (CAS) are diverse and are made up of multiple interconnected elements, and adaptive as they have the capacity to change and learn from experience. Echo (Hraber et al., 1997) is a class of simulation model of CAS, providing a population of evolving, reproducing agents distributed over a geography, with different inputs of renewable resources at various sites. Each agent has simple capabilities: offense, defense, trading and mate selection, defined by a set of chromosomes. Even though these capabilities are defined simply, they provide a rich set of variations illustrating the four kernel properties of CAS described by Holland (Holland, 1996).

The Inspiration: Mimicry

Henry W. Bates first published in 1862 his findings about the similarities and dissimilarities between Heliconiinae and Ithomiinae butterflies, after 10 years of research in the Brazilian rain forest. Bates collected ninety-four pieces of butterfly. He grouped them according to their similar appearance. He found butterflies having similar appearance, exhibiting morphological features which point to completely different species even families. Out of the ninety four species, sixty seven are now classified as Ithomiinae, while twenty seven of them are Heliconiinae.

Batesian Mimicry

Even though Heliconiids are conspicuously colored, they are extremely abundant. They are also slow in mobility. Still predators in the surrounding area, mostly insectivorous birds do not prey on them, because of their inedible and unpalat-

able nature. Also because of this phenomenon other edible and palatable species such as ithomiinae and pieridae, pretend to be heliconiids and thus enjoy protection.

Repulsive animals, such as heliconiids are very conspicuously colored. Having this noticeable property, they are easily recalled by predators. Their wing pattern works as a warning to predators. Once a predator has the knowledge of their inedible and unpalatable property, they would probably never attempt to try it again. As this is true, if any organism within close family and species, but being edible and having a deceptive resemblance to those conspicuously colored species will be avoided by the predators.

In general, the animal which is avoided by predator for unpalatable behavior is called the **model** and the imitating animal is called the **mimic**.

Mullerian Mimicry

Bates was not able to explain some phenomena of mimicry. Occasionally two inedible unrelated butterfly species are amazingly similar in appearance. An explanation for this was provided by Fritz Muller in 1878. When there are multiple inedible species it is hard for predators to recognize each of them to know which one to consume and which one to avoid. Because of predator's limited memory, all these species still lose their number even after being inedible. So to save this loss, and to prevent more sacrifice of their own kind, inedible species from different family also tend to evolve to have similar appearance. This phenomena is referred to as Mullerian Mimicry in the name of Fritz Muller.

Evolutionary Dynamics of Mimicry

The dynamics of mimicry has been investigated by Turner (Turner, 1988), where he states that the evolution of mimicry can be explained best by the process of punctuated equilibrium instead of phyletic gradualism. He came up with a synthetic theory (Turner, 1988), which was originated by Poulton (Poulton, 1912) and Nicholson (Nicholson, 1927), termed as the **two stage model**. This theory states that mimicry normally arises in two steps. A comparative large mutation achieves a good approximate resemblance to the model; it is followed by gradual evolutionary changes that refine the resemblance, in many cases to a high degree of perfection (Sheppard, 1972) (Ford, 1964). This two-stage theory has been applied for the explanation of Mullerian mimicry as well.

Mimicry Ring Any theory of Mullerian mimicry has to take into account the phenomenon of the coexistence of multiple mimicry rings. If we examine the local butterfly fauna in any area of the world, we will find that between all the aposomatic (warningly colored and defended) species present there are normally only a limited number of different patterns, normally far smaller than the number of species.

Each cluster of species, all sharing a common pattern, is termed as Mullerian mimicry ring. Thus, in the rain forest of South and Central America, most of the long-winged butterflies (ithomiids, danaids, and heliconids) belong to one of only five different rings.

Like Batesian mimicry, Mullerian mimicry can evolve in two stages: the mutational, one way convergence stage followed by the gradual, mutual convergence stage. It is worth mentioning that in the first stage only the less protected species can adopt the pattern of the better protected species; mutations in the other direction is not favored.

The Model: Evolution of Mimicry

Our model initializes with three kinds of agents. These agents have properties and behavior similar to the **model**, the **mimic** and the **predator**. We represent evolution of pattern for the model and the mimic with the help of Cellular Automata (CA) (Wolfram, 2002). CA can be easily represented by simple rules, which can be expressed as a binary string. The predator will be equipped with a Hopfield network (Hopfield, 1982), to have pattern recognition capability. The process of evolution will be occurring at the genetic level.

The choice of Hopfield Network memory for a predator can be considered appropriate as the number of patterns which can be recognized by this network is inversely proportional to the accuracy of recall. As more patterns are memorized, Hopfield network tends to make more errors. This behavior will be appropriate for the simulation of Mullerian mimicry. Mullerian mimicry happens because of limited memory of the predators. Because of this limited memory, multiple inedible butterflies seems to converge to a single ring.

Similar to the *Laws and Life* project by Peter Grogono (Grogono et al., 2003) the environment is designed as three dimensional, while the space will be of toroidal nature.

Past Work

Various models of mimicry has been simulated and explored. The model by Turner (Turner and Speed, 1996) and the mathematical model of Huheey (Huheey, 1988) tend to focus on the selective pressure on prey brought about by the particular learning abilities of the predator, and employ simple Monte Carlo or mathematical approaches.

Sherratt (Sherratt, 2002) provides an innovative perspective on the evolution of warning signals by considering coevolving predator and prey populations. The model's predators are deterministic, in that they have a fixed behavioral strategy over their lifetime, and cannot learn from experience. For both cryptic and conspicuous prey, each predator has fixed policy of either attacking or avoiding.

Models by Franks and Noble The latest work on modeling evolution of warning signals and mimicry with individ-

ual based simulation is done by Franks and Noble. Their initial work (Franks and Noble, 2002) seems to focus on putting some conditions of mimetic evolution in an individual based model with multiple species preyed upon by a single abstract predator, where the appearance of each prey species can evolve but their palatability is fixed.

On 2003 (Franks and Noble, 2003) another model for the origin of mimicry ring has been proposed which is based on two working hypothesis:

1. *All of the Mullerian mimics in a given ecosystem should eventually converge into one large ring in order to gain maximum protection.*

2. *If the Mullerian mimics do not converge into one large ring, then the presence of Batesian mimics could entice them to do so, by influencing the rings to converge.*

Although there are many mathematical and stochastic models of mimicry in the biological literature, this model gives attention to the evolution of mimicry ring phenomenon from an artificial life perspective.

FormAL Framework

The *"FormAL framework"* is a collection of concepts taken from Peter Grogono's Formal Artificial Life (FormAL) project (Grogono et al., 2003) and are used to build a framework for this model. In FormAL, an **Agent** is a simulated organism. It is designed simply, but with capabilities of reproduction using genetic information and modification of genome between generations. There is also interaction between agents while being able to survive and reproduce in a challenging environment.

The framework consists of a three dimensional world where agents get complete freedom of movement defined from their genetic representation. This toroidal **space** is a 3D lattice of discrete points, divided in multiple cells, which can be visualized. A **cell** is a three dimensional cubical section of the hyperspace. The purpose of the cells is to avoid expensive distance calculations. As two agents are considered *"close"* to interact when they are in the same cell and *"distant"* otherwise. **Time**, being an integer ($t \geq 0$), advances in discrete steps in the simulation, where at each step the agents update themselves.

Mobility An agent's position is calculated once during each step of update in time. The agents $position$, $force$, $acceleration$ and $velocity$ are all vector components. The $force$ component is calculated from agent's mobility gene, based on which some agents are faster/slower than others, and it is used to compute agent's $acceleration$. If the $force$ and $velocity$ are both zero, then the agent has no effect in motion. Otherwise, Newton's law is used to obtain the $acceleration$, which is integrated to obtain the new $velocity$ and new $position$.

The Prey: Models and Mimics

For this simulation the preys are heliconius butterfly and the representation of their wing pattern is with the help of cellular automata (CA). Every prey organism contain a binary genetic representation of CA which generates a fully developed pattern of size 16 by 16 bits from its initial state. With this pattern the predator will identify the prey and store its level of palatability in memory. We choose CA as it can be easily represented with the help of a binary genome and evolutionary operations on the genomic representation, such as mutation and crossover can easily be applied. This 8 bit genome has a decimal range between 0 to 255. Each of this value is associated to a unique CA pattern. In generating the pattern of figure 1, the genetic representation would be the 'New state of center cell'. To store in Hopfield memory we take a linear representation of this 2-D pattern and to find similarity between two patterns we calculate hamming distance between their linear representations.

Figure 1: Cellular Automata Rule 30

Current Pattern	111	110	101	100	011	010	001	000
New state of center cell	0	0	0	1	1	1	1	0

Table 1: Cellular Automata rule

Species diversity Using CA based pattern representation, population of prey species with a specific pattern can be grouped as one single species. Also by restricting inter species reproduction we can control the diversity of patterns. But mutation is applied when similar species mate with each other, so new species born out of generations of existing species. That is why we have two separate mutation rate for reproduction of prey species. One being the "Pattern Mutation Rate" (default values are mentioned in table 2) with which we control mutation of the first 8 bits of the genome while the "Genome Mutation Rate" is used to control mutation of rest of the 9 bit genome. Similar efforts of multiple mutation rate at varying location has been used in developing Echo (Hraber et al., 1997).

Genome The Genome of prey species consists of 17 bits. The first eight bits represent the rule, which is used to generate CA pattern. Next two bits are used to represent palatability of the organism. Following six bits are the magnitude of force with which mobility of the organism is calculated. The 17th bit is used to evaluate reproduction capability of the organism.

Reflection of punctuated equilibrium Punctuated equilibrium is more inclined to cladogenesis instead of gradualism. Also Turner (Turner, 1988) emphasizes on punctuated equilibrium to describe the evolution of mimicry instead of phyletic gradualism. The design of the model under discussion also follows Turner's explanation in terms of evolving mimicry. As it can be observed, new CA patterns evolve from existing ones in prey population just by a single mutation in the pattern gene. Mimics do not follow a gradual process of evolution to look close to models but rather the change happens randomly through a single step mutation. The mutations that are favored, helps the mimics to survive while the unfavored ones fail to persist. It can be observed later in table 5 how CA patterns of prey species can have vastly different configuration for a unit change in their representative gene, thus following the evolutionary process of punctuated equilibrium.

Palatability gene The palatability of each prey species is fixed and has been represented with 2 bits (index 8 to 9) of the genome giving it a range of 0 to 3 with four levels of palatability. For the combinations of 00 and 01 palatability is true, while for 10 and 11 it is false.

Interaction The prey have been defined to have many conglomerate behavior in the environment. Prey interaction with other prey species and with predators make the evolution of mimicry possible. Mobility of prey species and their reproduction capability are two important behaviors which result from interaction.

Mobility The mobility genes of the prey consist of 6 bits. These six bits are used to calculate the force with which each prey try to move towards any neighborhod cell. The algorithm sorts all neighboring cell descending to the number of prey species. Then it selects the cell which contains the highest number of prey with zero predator. If all the neighboring cells contain predators, then the algorithm sorts the neighboring cells descending on the number of predators and chooses the one which contains the least. This implementation is to have a conglomerate behavior of all prey species, while running away from predators.

Reproduction Every prey species starts reproducing when it reaches the *"Reproductive age limit"*. If it is capable of reproducing, which is decided based on its 17th bit gene, the prey will randomly select another prey species with similar pattern and palatability from the same cell and mate with it, given the other prey is also capable of reproduction. A prey is created from the existing genome of the two prey by applying single point crossover operation. Mutation is performed separately on the pattern gene and the rest of the genome, with two different rates to control them using the

values in table 2. So there is two point mutation for the genome.

Parameter	Value
Prey Size in the 3D FormAL environment	2 to 5
Reproduction age limit	100
Reproduction interval	1000
Pattern Mutation Rate	0.05
Genome Mutation Rate	0.5
Demise Age	2000

Table 2: Parameters to control prey population and visibility.

Predator

Predators in the system are designed to provide selection pressure to *models* and *mimics* for the evolution of mimicry. Similar to prey species, they are agents in the FormAL environment capable of mobility and reproduction. In addition, these agents are equipped with Hopfield Network Memory to be able to learn and recognize patterns of the prey species. Their mobility and reproduction capability are controlled at the genetic level, while their memory is not genetically controlled, as we could not find a suitable encoding for the genetic representation of Hopfield Network. Every new predator is born with zero memory and with no inheritance from parents. A set of parameters are defined to control predators' population and learning ability in the environment (table 3).

Learning The objective of a predator's interaction with prey is always to consume it. But based on the prey's pattern and palatability, the predator will either be able to consume it or throw it back to the environment. At this event the predator needs to learn the pattern with which the prey has been represented. The pattern represents palatability of the prey species, at least to the predator. Every time a new interaction is made by the predator its memory is initialized with all the existing pattern that has already been encountered and the new one. The learning procedure used for this memory is Hebbian Learning (Hebb, 1949), which represents a purely feed-forward, unsupervised learning. Initially the weights of the Hopfield Network are all set to zero. Using Hebbian rule, the outer product of the input - output vector pairs are calculated for each pattern. The outer vector matrix of all the patterns are summed to come up with the final weight matrix.

Input to memory Each prey contains an evolving cellular automata which is represented by a binary genome. This two dimensional pattern is serialized to be available as a one dimensional binary array, which is taken as input for any predator organism trying to interact with the prey. This binary representation of the pattern gets converted to a bipolar representation. Each input pattern consists of $m \times n = mn$ components, each component representing one pixel of the

pattern (*m* and *n* representing each dimension). The *m* by *n* pattern configuration is serialized by putting all row vectors in one single row sequentially.

Predator attack algorithm As soon as a predator reaches its attack age it selects random prey species around its vicinity and starts attacking them. This attack process also involves recognition of prey pattern. Two parameters have been defined to limit predator memorization and recognition process as both of these processes are computationally expensive. The "Hopfield Minimum Memory Size" is the number of memory a predator needs to store before making intelligent decisions about attacking a new prey species. When a predator is born, it starts attacking prey without any caution. But after every attack the predator will store its pattern and palatability level inside its memory. As soon as it reaches the minimum memory size, it will start making intelligent decision about attacking the next prey species. It will try to recognize the pattern and if found palatable, the prey will be consumed. Otherwise prey will be thrown back into the environment. If the pattern is not recognized predator will try to consume it and in the process will store its palatability and pattern into memory. In this way the predator memory is limited to *"Hopfield Maximum Memory Size"*. After reaching this memory predator will not store any more new pattern but will try to associate with the existing ones it has already stored.

Genome Each predator has a 5 bit genome. The first 4 bits are for mobility, while the last bit controls reproduction capability of each species.

Mobility Movement behavior of a predator is calculated from its genome. The first 4 bit converted to decimal is the magnitude of force (varying from 0-15) with which it will move towards the maximum crowd of prey present within its neighborhood. Number of bits are less than prey (6 bits) to reduce their maximum speed. If no prey is present in the neighborhood then this force is active in trying to keep predators distributed all over the cells with a constant mobile behavior. This behavior of predator has been designed to increase predator prey interaction in the simulation in terms of one agent chasing the other for survival of species.

Reproduction The fifth gene of the predator is used to represent their capability of reproduction. Depending on its binary value a predator in the simulation will or will not be able to reproduce. The reproduction process for predators is similar to prey species as the learning capability of predators do not have any genetic representation. Similar to the prey the predator also have the *"Reproduction Interval"* and the *"Reproduction Age Limit"* which is the minimum age it has to reach before starting to mate. Using these parameters

we can control the population of predators and by which we also control the rate of predation on prey species.

Parameter	Value
Minimum Memory Size	2 to 6
Maximum Memory Size	10
Hopfield Maximum Iterations	20
Attack Age	500
Attack Interval	100
Genome Mutation rate	0.3
Reproduction Age Limit	500
Reproduction Interval	1000 to 3000
Demise Age	2000 to 7000

Table 3: Parameters to control predator population and pattern recognition capability.

This model has been designed to come up with efficient results and achieve the main objective, *evolution of mimicry*. Creation and transformation of different mimicry ring and also the dynamics of it has been integrated to achieve interesting results. This model can also be considered as a complex adaptive system similar to Holland's work on Echo (Holland, 1996). The seven basics of a complex adaptive system which are: Aggregation, Tagging, Nonlinearity, Flow, Diversity, Internal Models and Building blocks (Holland, 1996) are present in this model. Individual components of this model such as the different types of agents and their properties can be considered as *building blocks*. Each prey species are *tagged* with individual pattern and palatability with which predators recognize them. We are providing different properties, behaviors and goals to the agents but setting them free in the environment to observe their *aggregate* behavior, resulting in *non-linear* or unpredictable outcome. The model has its *flow* as it progresses in time. Also there is *diversity* of prey species in the environment.

The Results

Data and analysis in this simulation has been concentrated on evaluating whether evolution of mimicry has taken place. This evaluation can be made with the number of different rings that has been created and the size of each of those rings along with the population of palatable and unpalatable species. Also it can be established whether Batesian Mimicry and Mullerian Mimicry have taken effect by analyzing the data set of these populations.

Mimicry Ring Reports

The mimicry ring reports consist entirely of the population of prey species categorized according to pattern and palatability. Data is stored at time interval of 10 iterations. As the number of rings that get generated reaches as many as 50 or more, and all the population of ring do not last for the entire simulation, so while storing data we have taken the most populous of the surviving 8 rings to plot. Mimicry Ring
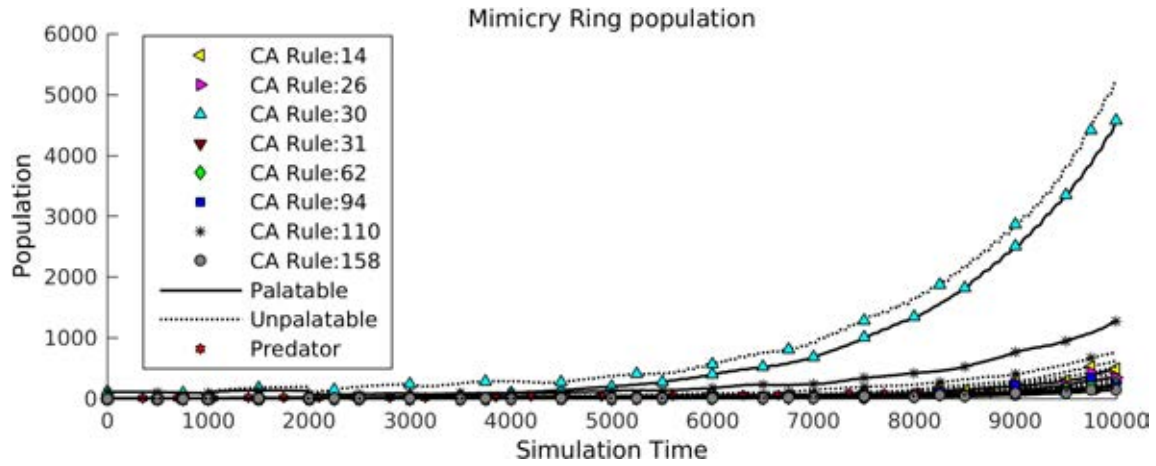
Figure 2: Population distribution of mimicry rings, initialized with 2 prey species, 10k iterations

Hamming distance between patterns is 10 % of the pattern size.

Initial configuration with two prey species

	Prey configuration			Predator configuration
Population	Rule110 (Palatable)		108	10
	Rule30 (Unpalatable)		108	
Reproduction	Age Limit	100		500
	Interval	1000		1200
Mutation Rate	Pattern	0.05		0.3
	Genome	0.5		
Demise Age	2000			2500
Minimum Attack Age				500
Memory Configuration		Minimum	2	
		Maximum	10	

Table 4: Agent configuration of 2 prey species

The set of parameters in table 4 were carefully selected to be the initial condition for this run of the simulation. This test has been done with two sets of prey species with very different Cellular Automata pattern and with opposite palatability and equal population. To control reproduction of the prey species their age limit has been set to 100 iterations into the time the species were alive. And the reproduction interval was set to 1000 iterations.

Pattern mutation rate has been set to a minimal level of 0.05 as by increasing this variable it is possible to increase the size of the number of mimicry rings present in the simulation. The genome mutation rate controls the rate at which genome of the child prey species will deviate from their parents.

Prey demise age has been kept to 2000 iterations while predator demise age is set to 2500. Predators in this simula-

tion generate selection pressure for the evolution of mimicry. So the longer a predator is present in the simulation it will be making intelligent decisions. Using this rate of demise for predator we were able to create successful mimetic population of prey species.

Initial population of predator species has been set to 10 which is in accordance with the prey population in the simulation. The reason for such low number of predator is, unlike prey species which are consumed by predators, there is no cause for the predator species to die except their natural cause of death, that is to reach their demise age. So predator population can explode very easily. That is why their population is controlled in a restrictive manner with the help of high reproduction age limit and reproduction age interval.

The plot in Figure 2 is simulation time verses prey population after running it for 10000 iterations. With the initial configuration in the above table we can observe that multiple rings of prey population have been created. Two prey species are considered to be in a ring if their CA pattern have Hamming distance within 10 bits. Population of palatable species have been represented with line curve while population of unpalatable species have been presented with dotted curve. Different signs of squares, triangles and diamonds have been used to distinguish between species of prey population. The simulation was initiated with two prey species having CA rule of 110 and 30, being palatable and unpalatable respectively. Rule 110 and 30 has been used as their phenotype is distinctly different from each other and Hopfield Network will easily distinguish them. Over time the population of CA Rule 30 dominates the population (Figure 2) as most predators recognize it as unpalatable. Similarly a palatable population of CA Rule 30 or within the same ring of palatable species starts rising, while at one point overlaps the population of CA Rule 110 (Time: 4000 approx.). CA Rule 110 was initialized as a set of palatable species.

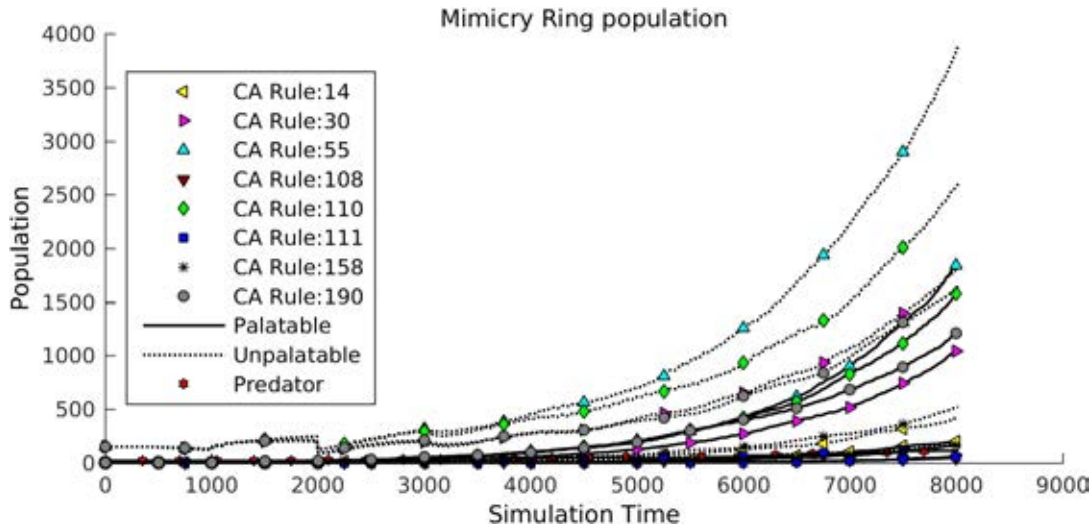We can observe from the above result that the evolution of

Figure 3: Population distribution of mimicry rings, initialized with 4 prey species all unpalatable.

mimicry has taken effect. A population of mimics were successfully able to exceed the population of other prey species, and the reason being, avoidance by predators of prey pattern similar to unpalatable ones. We can conclude that Batesian mimicry has taken effect in the simulation.

The number of rings in this simulation makes a slow increase from 2 at the initial configuration to 27 rings at the end of 10000 iterations. A small change in CA genetic representation can have a very large effect in terms of the phenotype of the pattern with which the prey is represented. For example if we take a look at the set of almost similar pattern genotype with vastly different phenotype in table 5.

CA Rule	$60 \equiv 00111100$	$61 \equiv 00111101$	$62 \equiv 00111110$
Pattern			

Table 5: Difference in prey pattern genotype and phenotype

All the patterns in table 5 have a genetic bit difference of 1. So by a single mutation there can be three different set of phenotype for a child organism from its parent. This is largely the reason for the increased number of mimicry rings created in the simulation. Only the 8 most populous rings are presented in the graphs with population verses simulation time.

To evaluate the simulation at a more complex level we increased the prey population to 900, consisting of 6 different species with very different pattern configuration. To boost predator-prey interaction we also increased the number of predator population to 30. This resulted in an enormous diversity of species where the total number of mimicry rings reached nearly 50. Details of this result can be found in (Islam, 2011).

Initial configuration with only unpalatable species

To further observe the effects of mimicry ring we initialize the simulation with all four unpalatable prey species. As explained earlier the minimum memory configuration is also set to four in accordance to the initial number of prey species. Rest of the parameters remain quite unchanged.

The results according to figure 3 are much expected. The population of unpalatable species have prevailed. After nearly 8000 iterations we can see unpalatable species of CA rule 55, 110, 30 and 190 have prevailed. All of their palatable counter parts are also increasing their population deceiving the predators.

This experiment is an ideal scenario for observing Mullerian mimicry. Mullerian mimicry occurs between multiple species of unpalatable prey population. From Franks and Noble (Franks and Noble, 2003), we note that multiple Mullerian mimicry rings are expected to converge into one large ring through the evolutionary process of punctuated equilibrium. But in this experiment as the predator's 'Minimum Memory Configuration' is set to four, all predators have the capability to recognize four prey patterns before starting to make intelligent decision of consuming them. By setting 'Minimum Memory Configuration' to one, also increasing 'Predator Demise Age' to 7000 and decreasing predator's 'Reproduction Age Interval' to 1500, we run the simulation for 6000 iterations, and there was no sign for all prey population to converge into one large ring. All four unpalatable prey population have a very dominant presence in the simulation. Even after reducing predator minimum memory to one pattern, different population of predators become familiar with different prey patterns, which resulted in the existence of multiple Mullerian mimicry ring instead of a single one.

In contrast when the simulation was initiated with only

palatable species all population of prey were consumed by predators at nearly 7000 iterations, details of which can be found in (Islam, 2011).

Analysis

For all possible initial conditions, Batesian mimicry has taken effect. It can be observed that for every ring of unpalatable species there is an existence of the palatable ring racing to reach the population count of its unpalatable counterpart. Effects of Mullerian mimicry can also be observed best for the experiment initialized with only unpalatable prey species. We initialized the model with 4 rings of unpalatable species with no palatable ones and after nearly 10K iterations, all of the initial unpalatable rings have survived with dominance. The cause of this behavior can be explained by the minimum number of patterns that each predator can store in memory, which was set to four. So this parameter was reduced to one to observe whether it is possible to converge all different unpalatable rings into one large ring, when predators are capable of memorizing only a single pattern. But as it turned out, the phenomena of "a single large ring" does not occur because different predators recognize different patterns resulting in multiple divergent Mullerian mimicry rings. It can be concluded that our results are consistent with those of Franks and Noble (Franks and Noble, 2003), that multiple Mullerian mimics do not converge into one large ring. These claims can only be made within the limits of this simulation.

Conclusion

Analysis of the results tell us that we have successfully been able to simulate the evolution of mimicry. In addition to that, this model provides a more accurate simulation of the fascinating natural process of mimicry rings and their shift in population. This model also verifies the theory of Turner in explaining the evolution of mimicry with punctuated equilibrium (Turner, 1988).

References

Ford, E. B. (1964). *Ecological genetics*. Methuen, London.

Franks, D. W. and Noble, J. (2002). Conditions for the evolution of mimicry. In *ICSAB: Proceedings of the seventh international conference on simulation of adaptive behavior on From animals to animats*, pages 353–354, Cambridge, MA, USA. MIT Press.

Franks, D. W. and Noble, J. (2003). The origins of mimicry rings. In *ICAL 2003: Proceedings of the eighth international conference on Artificial life*, pages 186–191, Cambridge, MA, USA. MIT Press.

Grogono, P., Chen, G., Song, J., Yang, T., and Zhao, L. (2003). Laws and life. In *ASC 2003: Proceedings of the 7th IASTED Conference on Artificial Intelligence and Soft Computing*, pages 158–163. International Association of Science and Technology for Development.

Hebb, D. (1949). *The Organization of Behavior*. Wiley, New York.

Holland, J. H. (1996). *Hidden order: how adaptation builds complexity*. Basic Books.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8):2554–2558.

Hraber, P. T., Jones, T., and Forrest, S. (1997). The ecology of echo. *Artificial Life*, 3:165–190.

Huheey, J. E. (1988). Supplement: Mimicry and the evolutionary process. *The American Naturalist*, 131:S22–S41.

Islam, M. (2011). Modeling the Evolution of Mimicry. Master's thesis, Concordia University, Montreal, Canada.

Nicholson, A. J. (1927). *A new theory of mimicry in insects*. Royal Zoological Society of New South Wales.

Poulton, E. B. (1912). *Darwin and Bergson on the interpretation of evolution*. Bedrock.

Sheppard, P. M. (1972). Some aspects of the geography, genetics, and taxonomy of a butterfly. *Taxonomy and geography*, pages 135–152.

Sherratt, T. N. (2002). The coevolution of warning signals. In *Proceedings of The Royal Society on Biological Sciences*, pages 741–746.

Taylor, C. and Jefferson, D. (1993). Artificial life as a tool for biological inquiry. *Artif. Life*, 1:1–13.

Turner, J. R. G. (1988). The evolution of mimicry: A solution to the problem of punctuated equilibrium. *The American Naturalist*, 131:S42–S66.

Turner, J. R. G. and Speed, M. P. (1996). Learning and memory in mimicry. i. simulations of laboratory experiments. *Philosophical Transactions: Biological Sciences*, 351(1344):1157–1170.

Von Neumann, J. (1966). *Theory of Self-Reproducing Automata*. University of Illinois Press, Champaign, IL, USA.

Wolfram, S. (2002). *A New Kind of Science*. Wolfram Media.

Artificial Societies

Population-Based Simulation of Gender Inequality Issues

John A. Bullinaria

School of Computer Science, University of Birmingham
Edgbaston, Birmingham, B15 2TT, UK
j.a.bullinaria@cs.bham.ac.uk

Abstract

A population-based simulation framework is presented that allows a principled approach for exploring gender inequalities in professional hierarchies such as universities or businesses, and how they might emerge, evolve and be rectified. Results from a representative range of cases involving gender-based discrimination and intrinsic gender-based ability differences are presented to demonstrate the power of the approach. Such artificial life simulations will hopefully inspire and facilitate better approaches for dealing with these issues in real life.

Introduction

There has been much discussion in recent years about gender imbalance in certain professions, such as university computer science departments (e.g., Camp, 1997; Altonji and Blank, 1999; Handelsman et al., 2005; Moss-Racusin et al., 2012), and how one might go about rectifying such situations, for example by better advertising or positive discrimination. However, it is often difficult to identify the best solutions when it is not clear what the main causes of the imbalance are (Halpern et al., 2007), and applying solutions based on incorrect assumptions could easily make matters worse.

One obvious potential cause of imbalance is simple discrimination against a particular gender (e.g., Davison and Burke, 2000; Moss-Racusin et al., 2012), and if that cannot be prevented, some form of positive discrimination might be an appropriate remedy. Another possible cause is that one gender might have evolved to be intrinsically less able (either on average or in the tails of the distribution) in a particular area (e.g., Geary, 1998; Browne, 2002; Baron-Cohen, 2004; Halpern et al., 2007; Halpern, 2012), and that results in less success in that area, and hence a tendency for that gender to avoid entering related professions in future. It is not obvious what interventions here would be most beneficial, or whether any intervention at all would be a good strategy. Another possibility is that, despite having intrinsically equal ability in the chosen area, one gender is disadvantaged by other factors, such as delays in career progression caused by child rearing and maternity leave (e.g., Ceci and Williams, 2011), and these cases may require different forms of intervention.

The idea of using computer simulations to model such situations and explore the best strategies for intervention in complex processes such as these is not new (e.g., Martell, Lane and Emrich, 1996; Robison-Cox, Martell and Emrich, 2007; Helbing, 2010), but what might not be so widely appreciated is that population-based simulations with ability-based selection of the type commonly used in computational intelligence (e.g., Engelbrecht, 2007) and artificial life (e.g., Bullinaria, 2009, 2010) can be effective for exploring the key causes, effects and solutions here. They can also model the evolution of such factors by natural selection. Moreover, the known methodological pitfalls that commonly arise with agent-based approaches to social and economic simulation (Richiardi, Leombruni, Saam and Sonnessa, 2006) are well understood in the field of artificial life and can thereby more easily be avoided. This paper presents a general framework for performing such simulations, and provides a selection of results that illustrate the power of this approach.

The remainder of this paper is organized as follows: The next section describes the proposed simulation framework and its associated simplifications and assumptions. Then results from some preparatory baseline simulations are presented to establish appropriate values for the various free parameters. The next two sections show how those results differ in the cases of gender-based ability differences and discrimination. Finally, the effect of interventions, and how the evolution of individual preferences affect the results, are explored. The paper ends with some conclusions and discussion.

Simulation Framework

This study begins by setting a few basic principles, and then explores what is possible within that general framework. The idea is to have an evolving population of individuals, with a range of intrinsic (innate) abilities, who can progress during their lifetimes to improve their position within their chosen professions. To draw reliable conclusions, the simulations need to be kept as clear and unbiased as possible (Bullinaria, 2009, 2010). Therefore, for the purposes of this initial study, a number of simplifying assumptions are made that help avoid any unnecessary confounding factors and also reduce the computational costs of the simulations to feasible levels:

1. There are two distinct genders, which are chosen randomly at birth with equal probability, and overall are equally able.

2. The distributions of innate individual abilities are the result of the evolutionary past, but are fixed for the duration of each simulation.

3. There are two distinct professions, which overall are equally valuable.

4. The initial individual abilities for each profession are determined randomly at birth and follow a normal (Gaussian) distribution. The means and/or the standard deviations of those distributions may depend on gender.

5. If one gender has higher mean ability in one profession, the other gender will have an equally higher mean ability in the other. The effect of the magnitude of such differences is one of the key factors to be explored.

6. Each individual can choose their profession randomly, or according to their abilities, or could have an intrinsic gender-based preference (i.e., probability) for choosing one profession over the other. Such preferences might emerge during the course of the simulations.

7. Individuals grow older, potentially improve their abilities through experience in their profession, and eventually retire and leave the working population.

8. Professional development involves a series of stages, and promotion between them is (by default) determined purely according to the best abilities currently available at each level (Rosenbaum, 1979). Discrimination or intervention in that process are other key factors to be explored.

9. If an individual does not get promoted within a set number of simulated years, they are likely to give up and leave the working population. It is also possible that varying percentages of individuals leave the working population for other reasons. Such details will need to be explored.

10. Individuals leaving the population are replaced by new individuals, and profession preferences may be based on the more successful individuals of previous generations.

There clearly remains much scope for variations within this general framework, and what emerges will depend on the relative magnitudes of the various parameters involved. There is also scope for variations designed to investigate the consequences of these initial simplifications.

The simulations follow common Artificial Life procedures. For each new individual in the population, a record is created and initialized with their innate gender, intrinsic abilities for the two professions, and any preferences for the professions. Thereafter it will be regularly updated with their age, chosen profession, stage in their profession, and number of years since reaching that stage. After updating for a number of simulated years, the population averages will settle down into a steady state, and the relevant results can be computed.

In principle, the above general framework can be used to simulate "professions" in any species. For example, food provision versus offspring protection in wild dogs. However, this paper will concentrate on abstract human professions, and therefore adopt human-like lifetimes and other parameters. It will assume, for simplicity, that all individuals enter their chosen profession at age 20 and retire at age 70, and that each profession has 7 stages, so 6 promotions are required to reach the top stage. Again for simplicity, it will be assumed that there is just one employer for each profession, so there is no need to simulate transfers between employers, or sub-groups of eligible individuals being considered at each promotion stage, as that is already known to bias the results (Lyness and Judiesch, 1999). A population size of 10,000 provides a sufficient number of individuals per profession per stage per gender for a reasonable level of competition at each stage,

even when the distributions become skewed. The ability scale is measured in arbitrary units, and that will be set (without loss of generality) by taking the standard deviation of the initial Gaussian distributions to be 1.0, and measuring all other ability differences relative to that.

A workable grain size for the simulations is one round of updates per simulated year, and 10,000 simulated years is plenty for all populations to settle down into a stable final state. Updating the individual ages, applying any ability increments, and replacing retired and removed individuals is straightforward. Dealing with the promotions between the stages of each profession requires further specification. One approach is to maintain pre-chosen numbers at each stage to correspond to typical companies (e.g., Robison-Cox, Martell and Emrich, 2007). An alternative approach, adopted here, is to promote a fixed fraction x of eligible individuals at each stage in each profession each year. Varying the promotion criteria and x, and requiring a certain number of years at a given stage before becoming eligible, are factors that will need to be explored empirically. Finally, an important aspect of this study is to incorporate into the standard setup a whole range of parameterized ability differences, discriminations and interventions that might be considered relevant.

The output of each simulation will usually be the final population of individuals, each with a gender, age, preference for profession, profession, ability in their chosen profession, and profession stage. Typically, the main factors of interest will be the various differences in population means between genders, such as how the profession preferences and numbers at each profession stage depend on gender (e.g., Robison-Cox, Martell and Emrich, 2007). Sometimes the evolution of the key parameter values throughout the simulation will also be of interest. To obtain reliable results, means and standard deviations over thirty runs of each simulation are computed, and unpaired t tests are used to determine the statistical significances of any differences found.

Baseline Simulation Results

The approach adopted is to first present the results from the simplest possible simulation set-up, and then systematically investigate how the potential variations affect those baseline results. Such a sequential approach also facilitates the setting of the various parameter values at each stage.

The baseline case simply has the most able individuals promoted at each stage, with an equal promotion fraction x at each stage varied from 0.01 to 0.05. The resulting numbers at each profession stage are shown in Figure 1. For very small x values (0.01), no individuals reach the upper stages. For high values (0.04 and above), more are in the highest stage than in some of the lower stages. Values around 0.02 to 0.03 probably come closest to realistic situations in academia or industry where there is a pyramid structure with fewer individuals as one moves up the hierarchy (Rosenbaum, 1979). Having different promotion fractions x_s for each stage s might be required to model realistic scenarios, and that can easily be done, but, for simplicity, the following will continue with a single value x across all stages.

The first variation requires individuals to wait at each stage for a certain number of years w before they become eligible for promotion. Now, a higher proportion of eligible
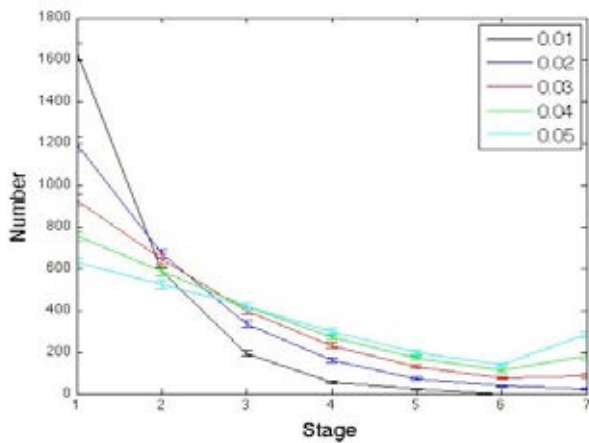
Figure 1: The initial baseline results showing the number of individuals at each profession stage and how that varies across a range of different promotion fractions *x*.
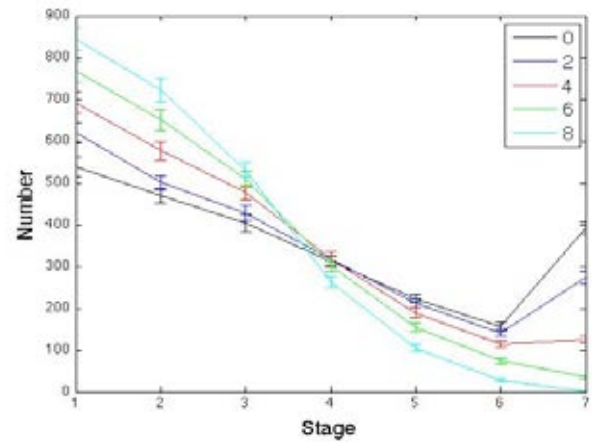

Figure 2: Number of individuals at each stage for promotion fractions *x* = 0.06 for different numbers of years *w* required at each stage before becoming eligible for promotion.
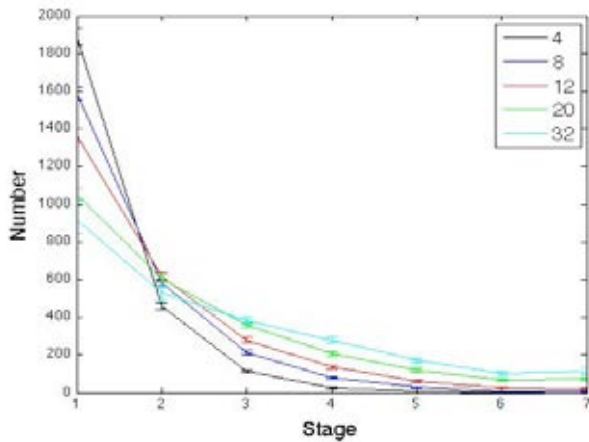

Figure 3: Number of individuals at each stage for promotion fractions *x* = 0.06 and wait *w* = 4 for different numbers of years *g* without a promotion before giving up and leaving.
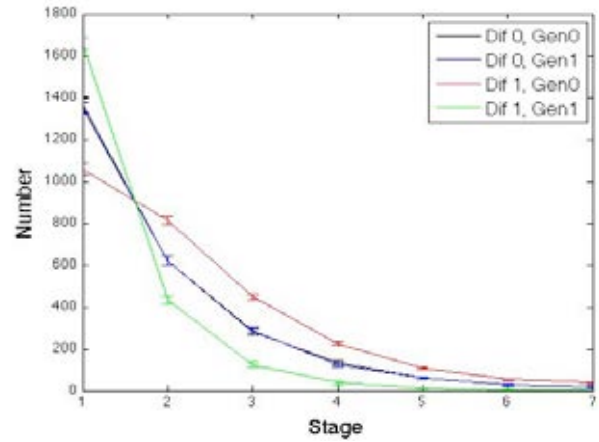

Figure 4: Number of individuals at each stage for *x* = 0.06, *w* = 4 and *g* = 12 for each gender (Gen0, Gen1) with no ability difference (Dif 0) and one std. dev. difference (Dif 1).

individuals need to be promoted each year to fill the higher stages. For a promotion fraction of 0.06, the effect of varying the required number of years *w* from 0 to 8 is shown in Figure 2. For 0 years, the promotion fraction is too high, as seen in Figure 1. Waits *w* of around 6 years lead to a reasonable distribution of individuals across the stages.

The next variation explores the effect of giving up and leaving the profession if promotion is not achieved within a certain number of years *g* after becoming eligible. Now, a slightly shorter wait *w* is needed so sufficient numbers are eligible for promotion at each stage. For a promotion fraction of 0.06 and a 4 years wait for eligibility, the effect of varying the number of years *g* before giving up from 4 to 32 is shown in Figure 3. For *g* = 32 years, there is little difference from never giving up. For fewer years, the numbers at later stages fall more sharply, and since the total number of individuals is fixed, there are more at the initial stage.

For situations that have learning or experience increase the individuals' abilities in line with the number of years in their chosen profession, or at each stage in that profession, it will

be interesting to investigate the different age distributions that emerge for each gender at each stage, and to go on to explore the effect of factors such as maternity leave.

Exploring Gender-based Ability Differences

Having seen how the three promotion parameters (*x*, *w*, *g*) affect the distribution across stages, explorations of the effect of gender differences can begin. The baseline simulations suggest that a promotion fraction of 0.06, a 4 years wait for eligibility, and giving up after 12 years, provides a reasonably realistic basis for the forthcoming simulations. Varying those values by small factors will inevitably change the results, but is unlikely to affect the general emergent patterns.

To begin, separate distributions for the two genders (Gen0 and Gen1) can be plotted for the case when each individual randomly chooses one of the two professions. Figure 4 shows the results for one particular profession when the gender difference is zero as above (Dif 0), and when the mean
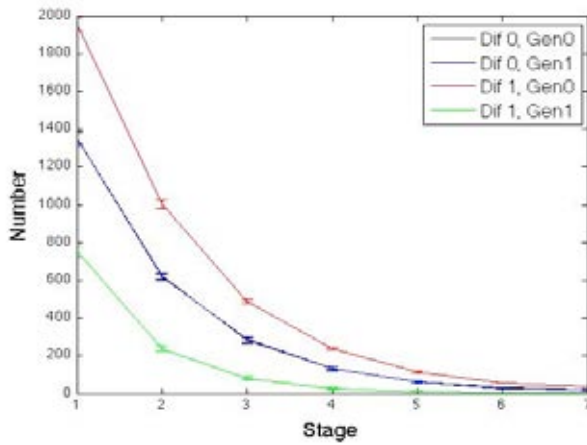
454

Figure 5: Number of individuals at each stage for same set-up as Figure 4, but with each individual pursuing the profession they are best at, rather than choosing one at random.
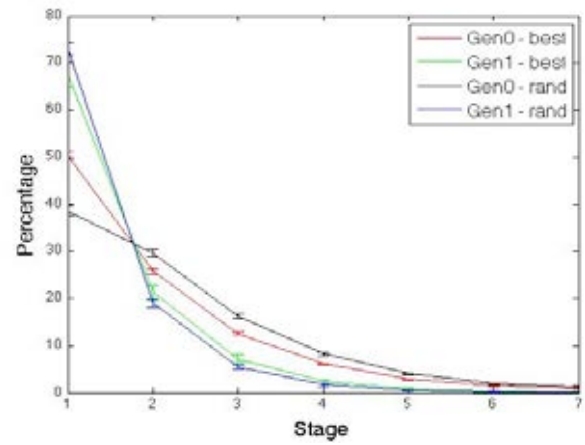

Figure 6: The Dif 1 results of Figures 4 and 5 as percentages of the whole population of each gender, for random choice of profession (rand) and choice of best profession (best).
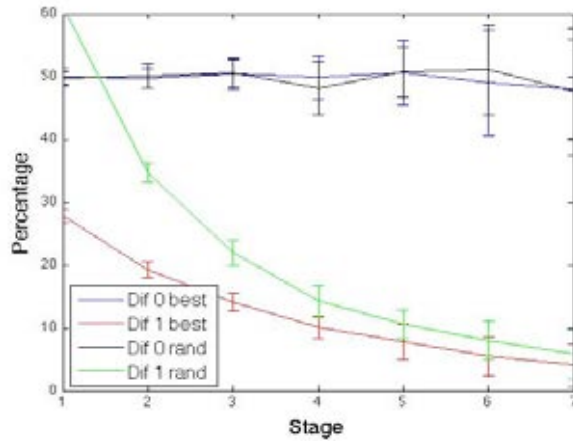

Figure 7: The percentage of Gen1 at each stage, with and without ability differences (Dif 1, Dif 0), for random choice of profession (rand) and choice of best profession (best).
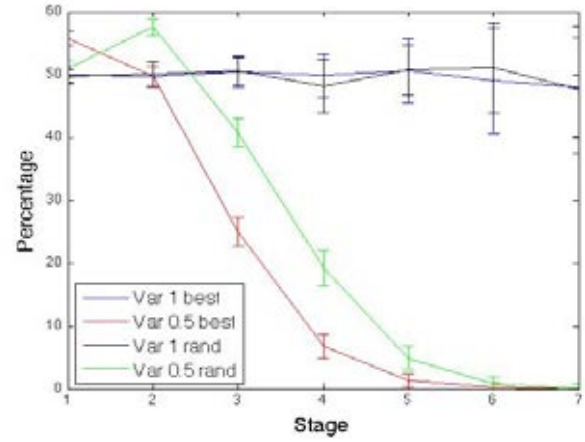

Figure 8: The percentage of Gen1 at each stage, for equal (Var 1) and half (Var 0.5) ability variance, for random choice of profession (rand) and choice of best profession (best).

abilities differ by one standard deviation (Dif 1). Obviously, there is no significant gender difference in the zero difference case. For a unit difference, the more able gender (Gen0) for the given profession has more individuals at the higher stages, and fewer stuck at the initial stage. The effect of dropouts also affects the total number of each gender participating in the profession. For the Dif 1 case there are 2746 (std. dev. 39) of Gen0 and only 2276 (std. dev. 40) of Gen1, which is a significant difference (*t test*, $p < 0.01$). For Dif 0, there are 2511 (std. dev. 54) of Gen0 and 2500 (std. dev. 46) of Gen1, with no significant difference (*t test*, $p > 0.05$).

If, rather than choosing their profession randomly, each individual were to choose the profession for which they have the best ability, the outcome is rather different as shown in Figure 5. Again there is no significant gender difference for the Dif 0 case, but for Dif 1 there is a massive statistically significant (*t test*, $p < 0.01$) reduction in the number of Gen1 individuals choosing the profession, 1114 (std. dev. 44) compared to 3893 (std. dev. 52) for Gen0. Figure 6 shows the Dif 1 results for random choice of profession (rand) and

choice of best profession (best), as percentages of the whole population of each gender at each stage in the profession. The ability-based choice of profession brings the gender distributions a little closer together, but Gen1 still has much reduced numbers at the higher stages compared to Gen0.

Another way of representing the data is as the percentages of each gender at each stage. Since these always total 100%, it is sufficient to present the results only for Gen1. These are shown in Figure 7 for best and random profession choice. For no gender difference (Dif 0), the percentage of Gen1 at each stage is not significantly different to 50%. For Dif 1 and random profession choice, the proportion at stage 1 is slightly over 50% (due to weaker individuals waiting for a promotion that never comes) and then falls for later stages. When the best profession is chosen, there is a lower starting point, and a slower fall off at later stages. A gender-based ability difference leads to stage distribution percentage differentials even when self selection leads to reduced participation of the less able gender. This kind of pattern, known as a shrinking pipeline, is found in real populations,

(Camp, 1997), though not necessarily for the same reason.

Interestingly, similar shrinking pipelines can arise even when there is no difference between genders in their mean abilities. If the variance in abilities for Gen1 is less than that of Gen0, as apparently happens with some human skills (e.g., Humphreys, 1988), that can give Gen1 a disadvantage at later stages of promotion, even if the means are the same for each gender. Figure 8 shows the effect of a factor of two in ability variances for random and best choices of profession. Similar patterns also arise when there are combinations of mean and variance differences. It is clear that there are many possible types of gender differences in ability that can account for the unequal gender distributions observed in real professions. From the simulation point of view, one can add further realism by replacing the simple Gaussian distributions used here with something more appropriate, but determining what those distributions should be might not be so easy (e.g., Benbow, 1988). Of course, the observed differences may also occur when there are no ability differences at all, and that is what will be investigated next.

Exploring Gender-based Discrimination

Perhaps the question of most practical importance is: how do the above patterns of gender differences vary when, rather than any intrinsic ability difference, there is discrimination against a particular gender? Given the range of possibilities, it is not feasible to study all types of discrimination here, nor the potential reasons for them. However, to demonstrate the power of the simulation framework, it is sufficient consider a simple abstract case. In particular, suppose an individual of one gender had to be vastly superior to a rival of the other gender before being promoted before them. That might, for example, arise due to different perceived prior probabilities of the abilities for the two genders (that are not necessarily correct) being used in conjunction with the actual evidence submitted with the promotion application. It could also be indirect, rather than direct, discrimination, for example due to one gender being less likely to be awarded prestigious invited talks or prizes (e.g., Gürer and Camp, 2001), or due to the promotion criteria being skewed in favour of one gender (e.g., Schneider, 1998; Ginther and Hayes, 2003; Mixon and Trevino, 2005). To be specific, the above simulations were re-run with a Gen1 individual only being promoted in preference to a Gen0 individual in the profession of interest when their ability is at least one unit higher. The symmetry was maintained by having discrimination in the opposite direction for the other profession. That leads to exactly the same stage distribution as in Figure 4 for the case when the Gen0 ability distribution really was one unit lower.

What is different between the discrimination and ability difference cases is the average abilities at each stage A similar pattern emerges for both random profession choice and ability-based choice, though choosing according to ability not surprisingly leads to better ability levels throughout. In the ability difference case (Dif 1), the ability of the weaker Gen1 is lower than Gen0 at the entry stage 1, but the ability-based promotions lead to much closer ability levels at later stages. In the discrimination case (Disc 1), the Gen0 abilities are reduced to the same degree as the Dif 1 case, because there is less competition for promotion, but the Gen1 abilities
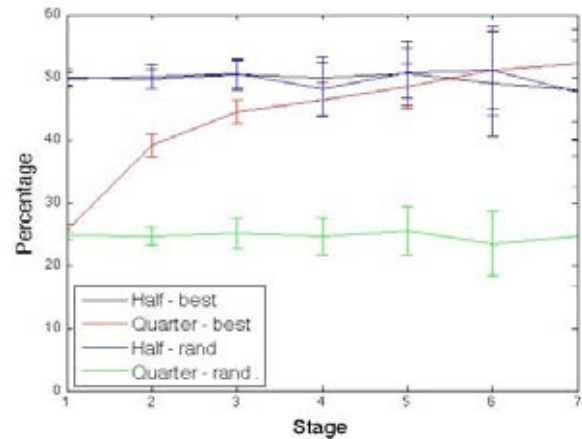

Figure 9: Percentage of Gen1 at each stage, with no ability or discrimination differences, starting at half or quarter total, for profession chosen randomly (rand) and by ability (best).

are much higher, due to the extra ability required to achieve promotion. Similar, though smaller (half a std. dev.) gender differences in ability have been observed in real corporations, suggesting that the presence of gender discrimination there (Lyness and Heilman, 2006). That, of course, does not mean there is necessarily a gender discrimination based glass ceiling in all cases, but there is certainly evidence consistent with that existing elsewhere too (e.g., Sabatier, 2010).

It is often suggested that indirect forms of discrimination are discouraging young women from entering particular professions, such as computer science, in the first place (e.g., Gürer and Camp, 2001). This effect can be modelled too, with the above simulations run in the same way whatever factor is reducing the numbers entering the given profession. The simplest possible case has no other promotion-based discrimination and no intrinsic ability differences, and the results in Figure 9 show for two rather different starting fractions that, as long as the profession is chosen randomly, those fractions persists throughout the stages. However, if the individuals that choose the profession are doing so according to their best abilities, they are likely to be at the higher end of the ability distribution, and fair promotions will allow them to rise quickly through the stages so that the gender proportions become equalized at the highest stages.

Figures 7, 8 and 9 demonstrate how rather different pipeline patterns emerge depending on the situation simulated. These are the "pure" cases. In practice, there is likely to be more than one form of ability or discrimination difference present, and untangling the various factors will be a challenge. This is where the simulation approach proposed here will prove most useful, as it enables all the possible combinations and variations to be simulated relatively easily and reliably, with the inevitable interactions accommodated automatically.

Exploring Intervention Policies

The shrinking pipelines and gender differences in the numbers entering some professions are often argued to be important issues that need addressing. For example, the lack of women in certain higher stages of academia might discourage women from studying those subjects and that may lead to critical
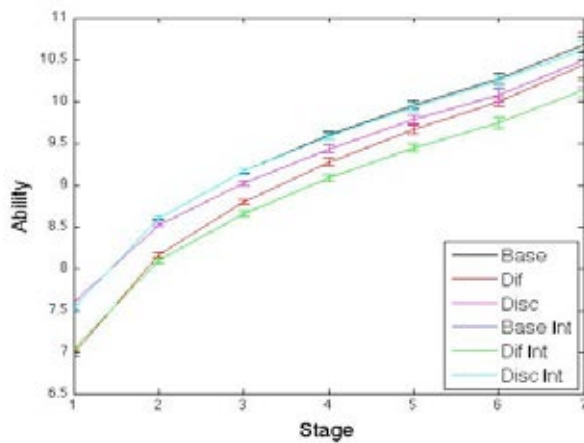
Figure 10: The average overall abilities at each stage for the standard and intervention populations for the three situations. The Base, Base Int and Disc Int results are identical.
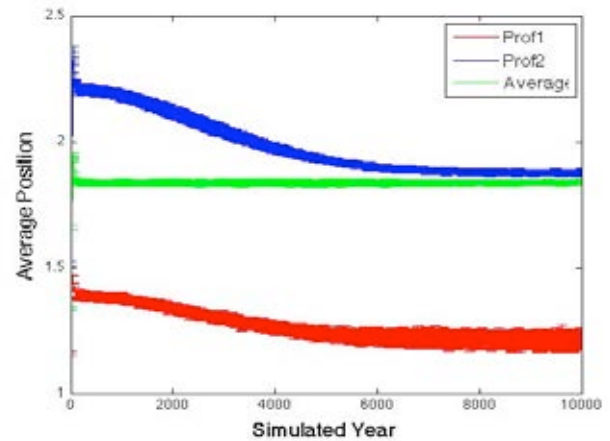

Figure 11: Average positions (profession stages) achieved by one particular gender while strong profession preferences emerge. The speed of change is parameter dependent.

skilled-worker shortages some areas (e.g., Camp, 1997).

The classic intervention would be to simply make sure that the numbers of each gender at each stage of each profession are as equal as possible. That can be implemented easily by ranking the eligible individuals of each gender separately, and promoting equal numbers of each gender to the next stage to give the required total number of promotions overall. The consequence of doing that in the above simulation framework does then lead to no significant number differences between the baseline, gender-based ability difference, and gender-based discrimination cases.

This intervention results in the expected differences in the corresponding abilities. All groups have the same average ability at each stage, except the less able Dif 1 Gen1 case which is one unit below at all stages, because equal numbers of promotions are taking place despite the lower ability levels of that gender. That leads to the important practical question: what is the average ability of the individuals at each stage, irrespective of their gender. That is shown in Figure 10. The baseline (Base) and baseline with intervention (Base Int) results are identical, since there is no gender imbalance for the intervention to correct, and these exhibit the best average abilities overall. The discrimination (Disc) case is slightly worse, since it unfairly allows weaker Gen0 individuals into the upper stages, rather than more able Gen1 individuals. The discrimination with intervention (Disc Int) case is no different to the base case, since the intervention successfully corrects the discrimination-based imbalance, and once again allows the best individuals at each stage to be promoted. The innate ability difference (Dif) case is overall worse than the base and discrimination cases, because that corresponds to Gen1 individuals having lower abilities than the base case, and that inevitably brings the population averages down. Obviously, if the gender difference corresponded to improved abilities for Gen0 over the baseline, rather than reduced abilities for Gen1, that would lead to improved population averages over the base case. The important question is: what will the consequences of intervention be in this case? As Figure 10 clearly shows, this makes the overall population performance (Dif Int) worse, particularly at the higher stages, since it forces the promotion of weaker Gen1 individuals over

better Gen0 individuals. This highlights the importance of understanding the problem before trying to correct it.

Evolving Preferences for Professions

So far, the simulations have been run for many generations to allow enough time for the various population distributions to stabilize, but none of the innate properties or preferences have been allowed to evolve or change from one generation to the next. However, the steady-state evolutionary computation approach underlying the general framework proposed here can automatically allow any inherent parameters (such as gender-based abilities, or preferences for particular professions) to evolve by natural selection if required (Bullinaria, 2009). In some species, such factors may be encoded genetically, but for current human professions it is more likely that such information will be passed on mimetically, in the form of social learning or mimicry (Bullinaria, 2010). For example, low take-ups of particular professions could emerge as a sensible reaction to poor progress in that profession by members of their gender in previous generations. There are clearly many such factors that could usefully be explored in the proposed simulation framework, and many ways they could be implemented, but one simple example should suffice to demonstrate the power of the simulation approach.

Suppose individuals choose their professions stochastically, rather than according to their abilities, but with particular intrinsic probabilities. The initial population would have equal preferences for the two professions, but individuals in later generations will have preferences that vary according to the success of recently replaced individuals. There are many ways that can be implemented, but one approach is enough to illustrate what typically emerges: each new individual copies the preference probabilities of the more successful of the last two replaced individuals (i.e. the one with the highest final position) but with a random "mutation" added from the range [-0.02, 0.02]. Those small mutations are sufficient to allow the preferences to drift away from the symmetric 0.5 values if a final position advantage emerges from doing so.

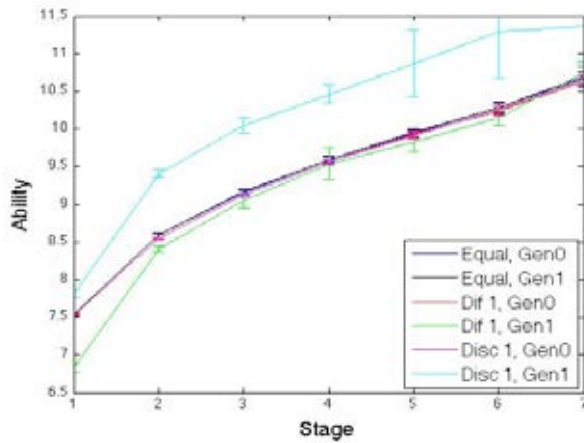There are eighteen distinct combinations to consider: three

Figure 12: Ability levels for each gender (Gen0, Gen1) in the baseline (Equal), ability difference (Dif 1) and discrimination (Disc 1) cases, for preference-based profession choice.
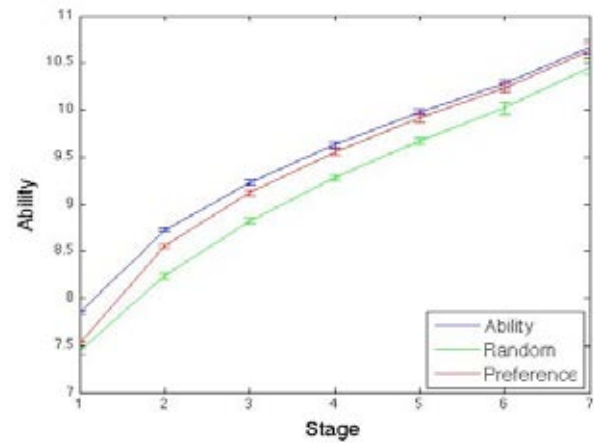

Figure 13: Ability levels for each gender in their appropriate profession for the ability difference (Dif 1) or discrimination (Disc 1) cases, for the three profession choice approaches.

basic conditions (baseline, gender-based ability difference, and gender-based discrimination), three profession choice approaches (by best ability, purely random, and random with intrinsic preferences), and each case can either involve, or not involve, intervention to equalize the numbers of each gender. Clearly, all the baseline cases, all the ability-based and pure random profession choice cases, and all the intervention cases, lead to the average preferences remaining at 0.5 because there is nothing to drive natural selection away from that symmetric case. However, those cases were still run to provide a check that no unexpected biases exist in the simulations. If there is no intervention, both the ability difference and discrimination case preferences shift towards the profession where the greatest success is most likely, while the preferences for the baseline case remain near 0.5 as expected. This is probably the simplest explanation of many of the observed gender differences in the numbers entering certain professions.

Since the professions and genders are set up symmetrically, and there is a fixed promotion rate at each stage, the overall average position at any given time for each gender must be independent of all the other factors, including any changing profession preferences, so it is not obvious what the changes really are optimizing. Figure 11 shows the average positions achieved by one gender as the preferences emerge when either an ability difference or discrimination favors Prof2 advances. The average positions achieved in *both* professions decrease as a result of the preferences changing away from being equal. Both genders gravitate towards the profession they do best at, increasing the competition there, and reducing the average position there for their gender. Those individuals remaining in the other profession face a larger pool of competitors of the better performing gender, and they are worse off on average too. It is the higher numbers in the best profession for each gender that keeps the average position constant throughout.

Often the most important issue for the businesses concerned is the average abilities at each stage in the two professions, irrespective of the genders involved. Figure 12 shows the ability levels at each stage after strong preferences emerge. Each profession employs individuals almost exclusively of just one gender, so the effects of gender-based discrimination

or ability differences are very small, and all the ability levels converge, except for the very few individuals who persist in the profession that discriminates against them.

A final question of interest here is how does the emergence of profession preferences affect the society as a whole, given that they have been driven purely by individuals wanting to reach higher positions in their chosen profession. Figure 13 compares the ability levels for each gender in their most appropriate profession. The best abilities overall come from ability-based profession choice, and the worst abilities come from a random profession choice. That is to be expected, given that random choice means many individuals will not end up performing according to their best potential. The not-so-obvious result is that emergent profession preferences are able to bring the ability levels close to the ability-based levels, particularly for the higher stages. This might be important for the population as a whole if the abilities are difficult to assess before the profession choice needs to be made.

Conclusions and Discussion

A general population-based framework has been proposed that enables the simulations of gender-based differences in various professions that involve ability-determined promotions up some form of hierarchy. The representative results presented were primarily chosen to demonstrate that the models do lead to reliable results in key simplified scenarios, though the approach can also be used to generate novel results for known real-world scenarios. The simulations presented have served to show how the principal factors can be studied effectively within the framework, and illustrated how distinct causes can lead to indistinguishable consequences, how preferences are able to emerge by natural selection, and how inappropriate interventions can make matters worse rather than better.

To simplify the presentation, all the simulations reported in this paper have only involved two professions, and all the gender-based differences have been symmetric across those professions. In reality, of course, there are many more than two professions, and a distinct lack of symmetries, but the proposed simulation framework is general enough to cope

easily with such complications. The results presented in this paper will then serve as the baseline against which those more realistic simulations can be compared.

There are clearly many other factors that could be built into the simulations, so hopefully this modeling approach will become more widely used in the future. A key issue is that there are too many potential gender-based effects to simulate all the possible combinations, but there are numerous specific hypotheses that could be tested empirically with the approach. One concerns the effect of gender differences in risk taking leading to differences in the variance of abilities (e.g., Schubert, 2006; Robison-Cox, Martell and Emrich, 2007). Another relates to distinct career paths to the top levels of some professions, with different gender effects on each (e.g., Robison-Cox, Martell and Emrich, 2007). It is also easy to fix the number of individuals at each level, rather than let it emerge by promoting a fixed fraction of eligible individuals each year, and that would lead to simulations more like those of the corporate management study of Robison-Cox, Martell and Emrich (2007) than the merit-based promotions more typical in academia. The consequences of other suggestions could also be explored, such as that women are less aggressive about seeking promotion, or are quicker to give up waiting for promotion, or more likely to leave or take time out for other reasons such as maternity leave, etc. All these ideas could be tested explicitly within the presented framework.

There are a number of further computational complexities that could relatively easily be incorporated into the simulation approach presented in this paper, that have previously been tested in simulations of Life History Evolution (Bullinaria, 2009, 2010), such as allowing abilities and preferences that change with time, or having parameter value distributions rather than parameters fixed at particular values. Hopefully, however, this short paper has been sufficient to demonstrate that the general framework proposed will allow all manner of additional factors to be explored in a more systematic manner than previously, in which the assumptions and simplifications are explicit, and the effects quantifiable. It is inevitable that some readers will disagree with the particular assumptions and simplifications employed in the simulations presented here. Hopefully progress can be made by other researchers using the approach to test the consequences of varying those assumptions and simplifications, and performing simulations more carefully matched to their own data and beliefs.

References

Altonji, J.G. and Blank, R.M. (1999). Race and gender in labor market. *Handbook of Labor Economics*, **3C**, 3143-3259.

Baron-Cohen, S. (2004). *The Essential Difference*. London, UK: Penguin.

Benbow, C.P. (1988). Sex differences in mathematical reasoning ability in intellectually talented preadolescents: Their nature, effects, and possible causes. *Behavioral And Brain Sciences*, **11**, 169-183.

Browne, K.R. (2002). *Biology at Work: Rethinking Sexual Equality*. New Brunswick, NJ: Rutgers University Press.

Bullinaria, J.A. (2009). Lifetime learning as a factor in Life History Evolution. *Artificial Life*, **15**, 389-409.

Bullinaria, J.A. (2010). Memes in artificial life simulations of Life History Evolution. In: *Proceedings of Artificial Life XII*

Conference (Alife XII), 823-830. Cambridge, MA: MIT Press.

Camp, T. (1997). The incredible shrinking pipeline. *Communications of the ACM*, **40**, 103-110.

Ceci, S.J. and Williams, W.M. (2011). Understanding current causes of women's underrepresentation in science. *Proceedings of the National Academy of Sciences*, **108**, 3157-3162.

Davison, H.K. and Burke, M.J. (2000). Sex discrimination in simulated employment contexts: A meta-analytic investigation. *Journal of Vocational Behavior*, **56**, 225-248.

Engelbrecht, A.P. (2007). *Computational Intelligence: An Introduction*. Sussex, UK: Wiley.

Geary, D.C. (1998). *Male, female: The evolution of human sex differences*. Washington, DC: American Psychological Association.

Ginther, D. and Hayes, K. (2003). Gender differences in salary and promotion for faculty in the humanities 1977–1995. *Journal of Human Resources*, **38**, 34-73.

Gürer, D. and Camp, T. (2001). Investigating the incredible shrinking pipeline for women in computer science. Final Report NSF 9812016.

Halpern, D.F. (2012). *Sex Differences in Cognitive Abilities (4th Edition)*. New York, NY: Psychology Press.

Halpern D.F. et al. (2007). The science of sex differences in science and mathematics. *Psychological Science*, **8**, 1-51.

Handelsman J. et al. (2005). More women in science. *Science*, **309**, 1190-1191.

Helbing, D. (2010). *Quantitative Sociodynamics*. Berlin, Germany: Springer.

Humphreys, L.G. (1988). Sex-differences in variability may be more important than sex differences in means. *Behavioral and Brain Sciences*, **11**, 195-196.

Lyness, K.S. and Heilman, M.E. (2006). When fit is fundamental: Performance evaluations and promotions of upper-level female and male managers. *Journal of Applied Psychology*, **91**, 777-785.

Lyness, K.S. and Judiesch, M.K. (1999). Are women more likely to be hired or promoted into management positions? *Journal of Vocational Behavior*, **54**, 158-173.

Martell, R.F., Lane, D.M. and Emrich C. (1996). Male-female differences: A computer simulation. *American Psychologist*, **51**, 157-158.

Mixon, F. and Trevino, L. (2005). Is there gender discrimination in named professorships? An econometric analysis of economics departments in the US South. *Applied Economics*, **37**, 849-854.

Moss-Racusin, C.A., Dovidio, J.F., Brescoll, V.L., Graham, M.J. and Handelsman, J. (2012). Science faculty's subtle gender biases favor male students. *Proceedings of the National Academy of Sciences*, **109**, 16474-16479.

Richiardi, M., Leombruni, R., Saam, N. and Sonnessa, M. (2006). A common protocol for agent-based social simulation. *Journal of Artificial Societies and Social Simulation,* **9**, 1-15.

Robison-Cox, J.F., Martell, R.F. and Emrich, C.G. (2007). Simulating gender stratification. *Journal of Artificial Societies and Social Simulation,* **10**, 1-8.

Rosenbaum, J.E. (1979). Tournament mobility: Career patterns in a corporation. *Administrative Science Quarterly*, **24**, 220-241.

Sabatier, M. (2010). Do female researchers face a glass ceiling in France? A hazard model of promotions. *Applied Economics*, **42**, 2053-2062.

Schneider, A. (1998). Why don't women publish as much as men? *Chronicle of Higher Education*, **45**, 14-16.

Schubert, R. (2006). Analyzing and managing risks – On the importance of gender differences in risk attitudes. *Managerial Finance*, **32**, 706-715.

Job Insecurity in Academic Research Employment: An Agent-Based Model

Eric Silverman[1], Nic Geard[2] and Ian Wood[1]

[1]Teesside University
[2]University of Melbourne
e.silverman@tees.ac.uk

Abstract

This paper presents an agent-based model of fixed-term academic employment in a competitive research funding environment based on UK academia. The goal of the model is to investigate the effects of job insecurity on research productivity. Agents may be either established academics who may apply for grants, or postdoctoral researchers who are unable to apply for grants and experience hardship when reaching the end of their fixed-term contracts. Model results show that in general adding fixed-term postdocs to the system produces less total research output than adding half as many permanent academics. An in-depth sensitivity analysis is performed across postdoc scenarios, and indicates that promoting more postdocs into permanent positions produces significant increases in research output.

Introduction

In recent decades the career landscape for academics has changed markedly. Upon graduating with a PhD, many aspiring academics enter a series of fixed-term postdoctoral research fellowships. Permanent positions are increasingly difficult to come by – in the UK, for example, only 3.5% of PhD graduates will succeed in getting an academic position (Powell, 2015). Intense competition for academic posts coupled with ever-increasing numbers of PhD graduates means that the academic workforce in the UK has shifted substantially toward fixed-term contracts – currently 68% of researchers in the UK are on fixed-term contracts (University and College Union, 2015).

Much has been written about the impact insecure academic working conditions can have on the individual. According to the University and College Union report *Making Ends Meet – The Human Costs of Casualisation in Higher Education* some 21% of UK academics on fixed-term or zero-hour contracts have trouble putting food on their dinner tables, despite many of these individuals having higher degrees and substantial experience. In the United States, adjunct professor positions – casualised positions in which teaching staff are paid per course or by the hour at very low rates, often without health insurance – now make up the overwhelming majority of academic positions on offer.

Some 75% of US academics are now 'contingent teaching faculty', or adjuncts, a ten-fold increase since 1975 (Hoeller, 2014).

In the case of funding, evidence suggests that the current trajectory of academia – toward larger grants targeted at 'research leaders', which tend to bring more fixed-term postdocs with them – is not necessarily a productive one. A study of the projects funded by the National Sciences and Engineering Council of Canada found that scientific impact was a decreasing function of funding – bigger projects produced less impact per dollar than smaller ones (Fortin and Currie, 2013). Similarly, a recent study of 398 project PIs in the UK found that while productivity – number of publications – was positively correlated with funding, the relationship with impact factor and citation number was far weaker, and diminishing returns set in as funding levels rise (Cook et al., 2015).

While there have been modelling studies examining competitive research funding systems and illuminating some of these shortcomings (Geard and Noble, 2010), currently we are unaware of any attempts to model the structure of academic careers. This seems a significant oversight given the prevalence of stress and job dissatisfaction reported by postdocs worldwide (Van der Weijden et al., 2015). In such circumstances, could the stress and uncertainty of postdoc employment lead to substantial impact on research productivity in academia?

Given that the majority of postdocs are hired with grant funding, we suggest that understanding the impact of the trend towards fixed-term contracts will require an examination of both competitive research funding structures and insecure postdoctoral employment. This paper presents a first attempt at modelling a simple academic system which incorporates both of these key elements. We propose that modelling techniques drawn from complex systems science are highly appropriate for this kind of meta-research, as we need to explicitly represent the complex and inter-related nature of sector-wide constructs like research funding councils and academic career paths.

Utilising previous work on modelling research funding

(Geard and Noble, 2010), we have constructed an agent-based model in which established academics compete for grants while postdocs compete for tenure. By examining the impact of these interrelated systems on overall research productivity we seek a deeper understanding of how the trend toward fixed-term contracts in academic employment may have affected the academic community. Performing a detailed sensitivity analysis allows us to examine the impact of key model parameters on overall research output.

The Simulation Model

The simulation model used here is based substantially on previous work by Geard and Noble (2010). The same code base was used as a starting point, and the postdoc employment mechanisms were added on top of this core functionality. Model parameters and postdoc behaviour were inspired by the characteristics of UK academia. A brief summary of the funding model will be provided here; for further details, please see the original paper.

Core Research Funding Model

The model represents research funding as a competitive bidding process, in which academic agents submit proposals each semester in the hope of obtaining a grant. Here we assume that 30% of research proposals are funded, so agents attempt to get their research funded by investing time into bid preparation. The grant evaluation process in the model evaluates proposals based on the research quality of the submitting agent and the amount of time spent preparing the bid – we also assume diminishing returns on time spent.

Agents each have an individual research quality, which is a figure ranging between 0 and 1. Each semester, agents produce research output based on their research quality and modified according to their time allocation strategy for bid preparation. The final output numbers are conceptualised as 'units of research', i.e., scientific publications.

Agents who are funded benefit from an increase to their research quality, which is intended to represent the material benefits of research funding: increased resources, time bought out from teaching obligations, and so forth. However, getting proposals funded requires bid preparation time, which in turn reduces research productivity. Agents make these decisions by looking at their history of successful applications in the recent past and altering their time allocations to attempt to optimise their success rates – the 'Memory A' strategy in Geard and Noble (2010). Here we defined the recent past as the previous ten semesters, as shorter memories produced more chaotic application behaviour due to the volatility introduced by the regular influxes of new agents in the baseline and postdoc scenarios.

Baseline Model: Simple Growing Population

The core funding model begins with an initial population of 100 established academics which stays static as the simulation runs for 100 semesters. For this paper we first developed a basic extension of the model which assumes that some portion of the disbursed grant funding is used to fund additional permanent academic positions. This allows us to provide a simple baseline case to compare with the postdoc scenarios.

Every semester, a number of additional academics are added to the system equal to half the number of disbursed grants, rounded up. These academics are given a random level of individual research quality, which is reduced slightly for the first two semesters to represent their acclimatisation process as they join the ranks of tenured faculty. Otherwise these new academics behave identically to the established academics.

Postdoc Model

Postdocs were added to the simulation through the implementation of some additional mechanisms. While the core funding model remains the same, as does the behaviour of the established academics, in the postdoc case additional agents with unique properties are added to the simulation each semester.

When a grant is successfully funded, the agent who submitted the grant receives the same beneficial increase to their research quality as in the baseline case. In addition, every grant funds a new postdoc who is then added to the simulation. Postdocs differ from established academics in several key aspects:

1. Postdocs spend 100% of their time doing research

2. Postdocs are unable to apply for grants

3. Postdocs are on fixed-term contracts, ranging from 4 to 10 semesters in length

4. New postdocs experience the same small reduction in research quality as new academics in the baseline scenario

5. At the end of their contract postdocs experience a reduction in research quality for their final two semesters

Note that the inability of postdocs to apply for grants is not true everywhere – postdocs in the UK are unable to apply for grants, but can do so in Australia, for example. In some countries this varies between funding agencies, as in the United States.

At the end of a contract, some postdocs will be fortunate enough to get promoted into a permanent position. Promoted postdocs are converted into established academics, will become able to apply for grants, and must devote some portion of their research time to bid preparation. The likelihood of being promoted is determined by a parameter, the *promotion chance*, which will be examined in detail later in this paper. The default promotion chance is 15%, a rate which was thought to be reasonable after comparing the
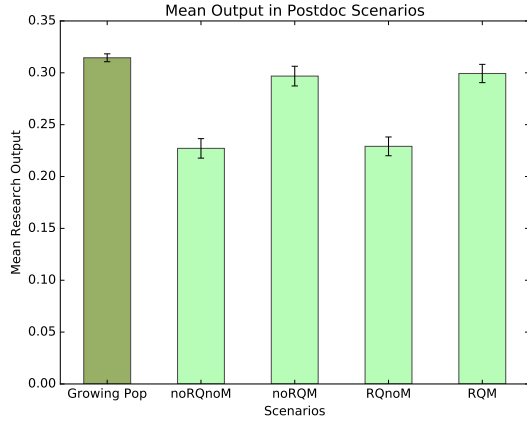
Figure 1: Mean research output per academic across five different scenarios. The Growing Population scenario does not include postdocs.

Figure 2: Total research output for the entire system across five different scenarios. The Growing Population scenario does not include postdocs.

widely variable statistics between different UK higher education organisations.

In the real world, promotion is not always a direct result of performance – timing, luck, geographical location, connections, and even nepotism can play a role. In order to examine the role of merit-based promotion in this model, we allow the promotion process to be set to either take into account an agent's research quality, or to promote a random selection of agents. In the former case, agents are ranked by research quality and a percentage of the top-ranked agents corresponding to the promotion chance will be advanced into permanent positions. In the latter case, a random sample of equivalent size is selected to be promoted. These two cases are referred to as *RQ* and *noRQ* scenarios in the Results section. Agents who are not promoted are removed from the system and may no longer contribute to research output.

In order to model the notion that postdocs provide useful experience and thus increase the quality of new permanent academics, we included an option for a *mentoring bonus* for newly-promoted agents. In the mentoring scenario, newly-promoted agents gain a significant bonus to research quality to represent the proposed benefit of this intensive research experience. The mentoring and no-mentoring scenarios are referred to later as *M* and *noM*, respectively. In the default scenario, the mentoring bonus adds an additional 0.5 to an agent's research quality.

Results

In order to examine the impact of fixed-term contracts on academic research productivity, we analysed the research output of a number of different scenarios. The first set of scenarios compares a variety of postdoc settings with the baseline growing population case. The second and third sets of scenarios investigate the postdoc model more deeply,
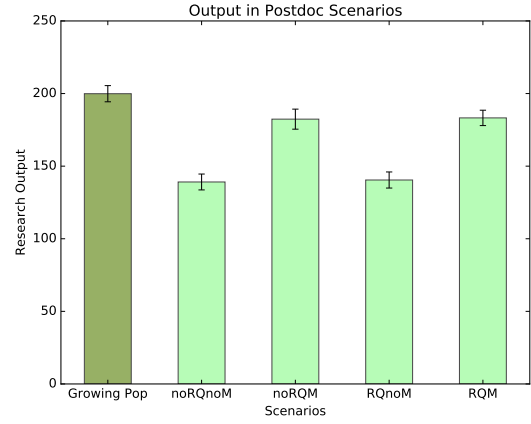
looking at the impact of varying the promotion chances of postdocs and the levels of job-hunting stress they experience at the end of a contract, respectively. All three of these sets of scenarios used default settings of the key parameters as follows: postdoc promotion chance at 15%; mentoring bonus to research quality at 0.5; new postdoc stress and end-of-contract stress at 0.3. The mentoring bonus was turned off in certain scenarios.

Following these analyses, we recorded the results of 8,000 runs of the simulation across a comprehensive range of parameter settings and then performed a detailed sensitivity analysis. These techniques will be described in more detail in the Sensitivity Analysis subsection below.

Scenario Set 1: Postdocs vs Permanent Academics

The first set of scenarios compares four different postdoc scenarios to the baseline scenario in which the population consists entirely of permanent academics. These four scenarios correspond to the four possible combinations of the *RQ* and *M* parameter settings described above:

Table 1: Postdoc Scenarios – Set 1

Scenario Name	Settings
noRQnoM	Random promotions, no mentoring
noRQM	Random promotions, mentoring
RQnoM	Non-random promotions, no mentoring
RQM	Non-random promotions, mentoring

Figure 1 provides a comparison of mean research output per academic across these five scenarios. The results are averaged across fifty runs of the simulation for each scenario, and the error bars indicate the standard deviation. The basic growing population scenario outperforms all four postdoc
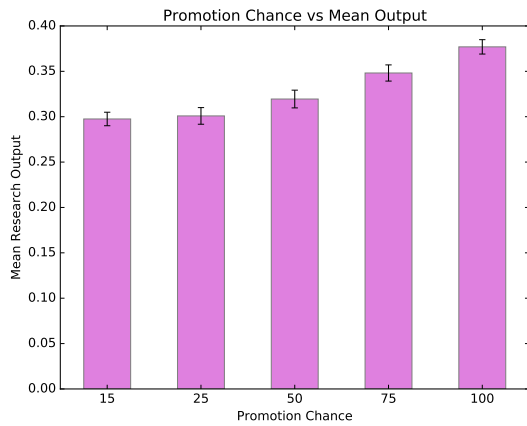
Figure 3: Mean research output per academic for five different values of the promotion chance parameter.

Figure 4: Total research output across the system for five different values of the promotion chance parameter.

scenarios on this measure; in the postdoc scenarios, mentoring appears to be the most important driver of increased research output. Perhaps surprisingly, promoting postdocs randomly or non-randomly seems to make little difference to the final outcome. Results suggest that the drop in research output in the postdoc scenarios may derive from the instability introduced by a constant influx of postdocs with unpredictable research quality; the fact that agents require a longer memory in the postdoc scenario in order to settle on stable time allocation strategies supports this interpretation. Postdocs are also ineligible for grant-related research bonuses, which negatively affects research output levels.

Figure 2 shows another comparison between the five scenarios, this time for total research output across the whole academic system. The figures displayed here are the mean final research outputs averaged across fifty simulation runs for each scenario. Again we see that the basic growing population case outperforms every postdoc scenario, even though only half as many permanent academics are hired in that scenario. Mentoring again takes precedence in the postdoc scenarios, as the mechanics of promotion seem to make little difference to the final outcome.

Scenario Set 2: Promotion Chance

In this second set of scenarios we compared the final outputs of simulation runs using five different values of the promotion chance parameter. This parameter sets the likelihood of a given postdoc being promoted into a permanent position. In this set of scenarios we used parameter settings for the *RQM* scenarios from Set 1, as these seemed to provide the most favourable results among the possible postdoc scenarios. All other parameters were kept at the default values indicated in the model description above.

Figure 3 shows a comparison of the mean research output per academic at the end of the simulation. Again each
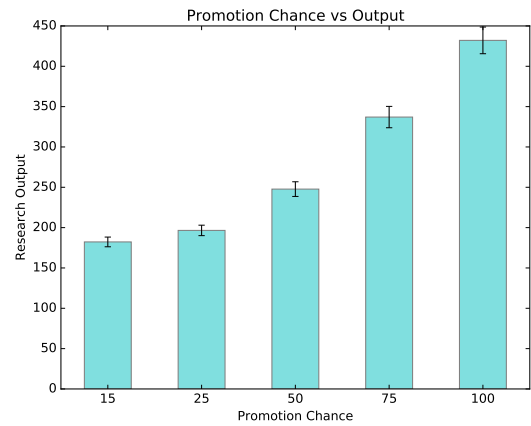
scenario was run fifty times and the error bars indicate the standard deviation. In this set of scenarios the 100% promotion scenario was the most successful; these runs gave us the highest values for mean research output out of any postdoc scenarios we ran.

In Figure 4 we provide a comparison of total research output across the entire system using the same five values for promotion chance. Here we see a marked increase in overall research productivity when 100% of postdocs are promoted – the mean total output is significantly more than double what we see at 15%. However, this clearly would be the most expensive option in a postdoc-employing academic world – in a later subsection we will examine the cost issue in more detail.

Scenario Set 3: Job-Hunting and Stress

In the third set of scenarios, we investigate the impact of job insecurity on research output. As described in the introduction, a number of studies of postdocs and fixed-term academics have revealed the difficult consequences of insecure and low-paid academic work on individuals. In order to represent the potential negative impact of these stressors, we have implemented a small research quality penalty, set to 0.3 by default, which represents the impact of postdocs needing to spend time searching for academic jobs, many of which require lengthy and detailed application processes to be completed, and the stress caused caused by impending redundancy.

For these analyses we again collected data from sets of 50 simulation runs for five different values of the key parameter, in this case the job-hunting/stress penalty applied to postdocs reaching the end of their contracts. We decided to set the upper bound for job stress at 0.7, as we felt it reasonable to assume that most postdoc positions, while potentially stressful, would likely not take up more than 70% of
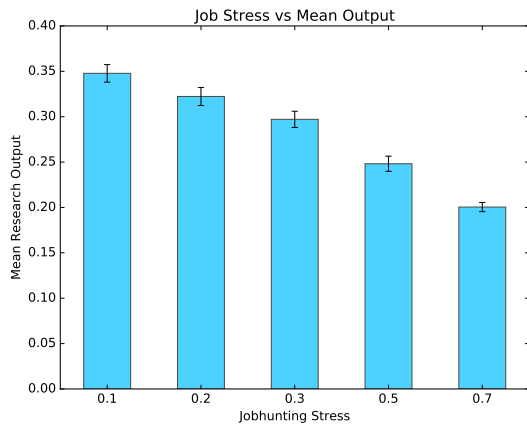
Figure 5: Mean research output per academic for five different values of the job-hunting stress parameter.

Figure 6: Total research output across the system for five different values of the job-hunting stress parameter.

researchers' time due to that stress.

Figure 5 shows a clear downward trend in mean research output for individual academics as the level of job stress increases, and Figure 6 shows a near-identical result for total research output across the entire system. We note that in all of these scenarios postdocs only represent approximately 10-15% of the total academic population at the end of a normal simulation run, and yet the impact of this stress parameter is very significant.

This result indicates the prominent role that even this small population of postdocs has in the research landscape. Established academics must divide their time between research and bid preparation, and failed bids often lead researchers to put enormous amounts of time into the next round of preparations. As a consequence, established academics tend to enter periods of 'feast or famine' in which they either spend all their time writing bids and failing to produce any research, or they succeed with several applications in a row and feel safe in reducing their bid preparation time in order to increase their research output – which is then further increased by the research bonus added by the grant itself.

In contrast, postdocs devote 100% of their available time to research, and since they cannot apply for grants they are not distracted from their work by the grant-funding lottery. During the average simulation run postdocs frequently average nearly double the research output of established academics, with only top-achieving grant-holders able to exceed their productivity. Postdocs thus take up the slack in the research community while everyone else fights to win grants. As a result, postdocs account for a significant fraction of the overall research output, and thus reductions to their productivity have a strong impact on the overall research output of the academic population.
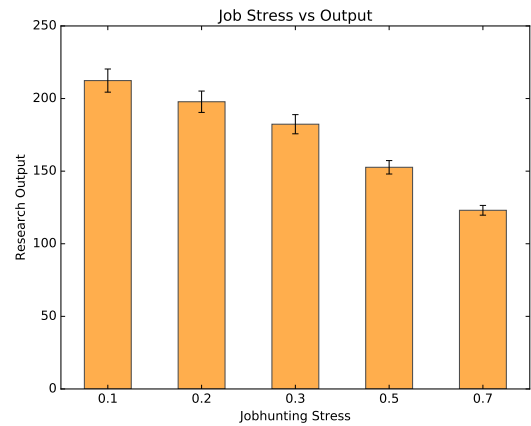
Sensitivity Analysis

Despite the relative simplicity of the agent behaviours in this model, the system does incorporate a number of elements which may interact in unexpected ways. In order to further understand the dynamics of the model we looked to uncertainty quantification methods, which can allow us to delve deeper into the effects of each model parameter on research outcomes.

Our chosen method was inspired by a previous UK research project known as Managing Uncertainty in Complex Models, or MUCM (`http://www.mucm.ac.uk/`). The MUCM team developed some specialised software specifically for use in the analysis of complex computational models. One of these pieces of software, GEM-SA, implements a Gaussian process emulator, which allows us to perform an in-depth sensitivity analysis of complex computational models with multiple input parameters (O'Hagan, 2006), including agent-based models (Silverman et al., 2013).

Detailing the construction of Gaussian process emulators is beyond the scope of the current paper, so we recommend reading Kennedy and O'Hagan (2001) for further details. To summarise briefly, Gaussian process emulators provide a measure of the influence of each individual input parameter on the total output variance of the simulation. The emulator works on the assumption that the single output variable specified – total research output at the end of the simulation, in this case – can be understood as a composition of a series of main effects driven by the input parameters, interaction effects for all combinations of those parameters, and a constant term. In the current implementation, additional uncertainty introduced by the computer code itself is also taken into account. In essence, the emulator builds a statistical model of the computer model based on an input training set.

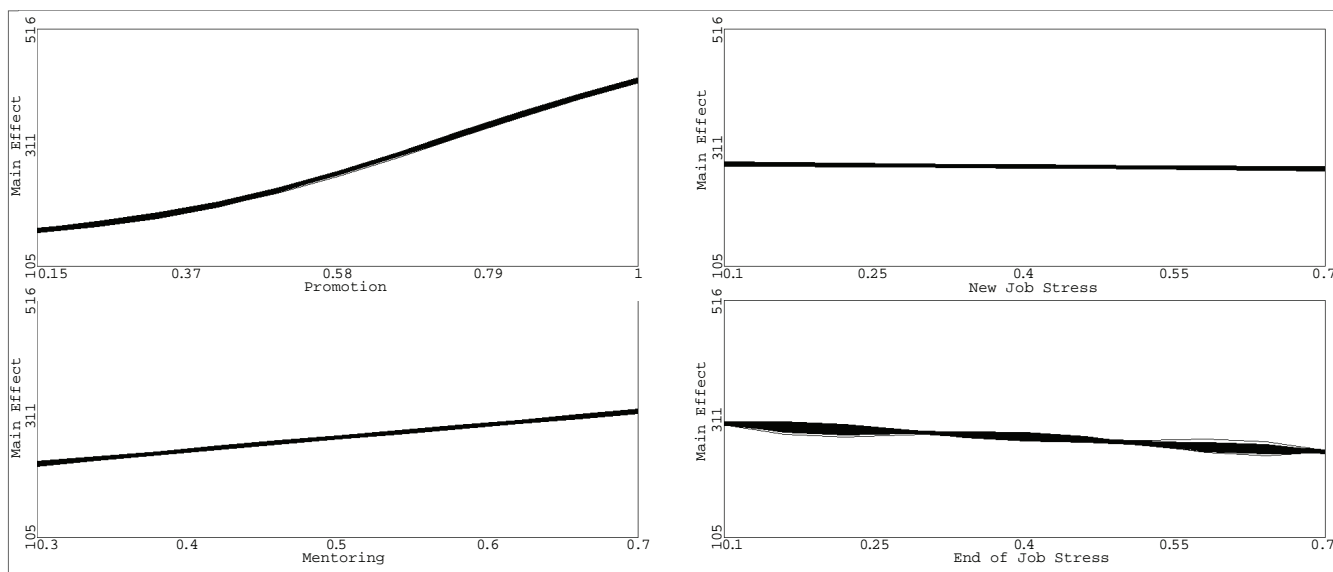For this sensitivity analysis we chose four key input pa-

Figure 7: Results of Gaussian Process Emulator demonstrating the impact of four input parameters on final research output values for the whole system. The emulator was run with 400 different parameter combinations; each combination was run 20 times and the outputs averaged. Source: GEM-SA software (own calculations).

rameters – postdoc promotion chance, mentoring bonus for just-promoted postdocs, and the stress caused by entering a new position and by leaving a position. The final output of interest was the total research output across the system at the end state of the simulation.

GEM-SA requires a large training set in order to produce good results, so we generated a set of 400 possible parameter combinations for these four inputs – the maximum allowable in the GEM-SA software. Promotion chance values ranged between 0.15 – 1.0, mentoring bonus between 0.3 – 0.7, and job stress for both entering and leaving positions between 0.1 – 0.7. We then ran each one of those 400 settings 20 times, resulting in 8,000 total simulation runs, and took the mean of the total research output for each setting, then passed those results to GEM-SA.

Table 2 provides a summary of the GEM-SA output after 41,000 runs of the emulator. We can clearly see that the single largest driver of research output in these postdoc scenarios is the likelihood of postdoc promotion, which accounts for 86.43% of the final output variance. The mentoring bonus provided to newly-minted academics is the second largest contributor, accounting for 8.87% of the output variance. Job-hunting stress at the end of a contract plays a small role in the final results, but interestingly stress due to entering a new position is largely inconsequential – this could be due to the tendency for new academics to struggle to achieve consistent outcomes regardless of their stress levels, at least until they settle into a more stable pattern of bid preparation.

In Figure 7 we provide the graphs generated by the GEM-

Table 2: Effect on output variance from input parameters

Parameter	Variance (%)
Promotion Chance	86.43
Mentoring Bonus	8.87
New Postdoc Stress	0.08
Job-Hunting Stress	2.57
Promotion x Mentoring	1.31
Promotion x New Stress	0.01
Promotion x Job-Hunt Stress	0.69
Other Interactions	0.02

SA software, which show the effects of each input parameter on total research output. The graph demonstrates that as the postdoc promotion chance increases, the total research output increases as well – and once again the effect on the final output is significant despite the relatively small size of the postdoc population. Similarly, the size of the mentoring bonus applied to newly-promoted postdocs has a positive impact on total research output, although significantly less pronounced than the effect of promotion chance. Both starting a new job and reaching the end of a contract appear to impact negatively on research output, though the influence from the latter is somewhat variable. This makes intuitive sense, given that the amount of postdocs eligible for redundancy in each simulation will vary significantly depending on random factors in the simulation, in contrast to new-job stress which every postdoc is guaranteed to experience.

Return on Investment

While the results to this point reinforce the interpretation that higher rates of postdoc promotion lead to greater research output, in the real world this would have substantial cost implications. Postdocs are welcomed by universities as employees given that their salaries are paid for by external funding in most cases – taking those employees on as permanent academics requires a significant investment from the university's point of view.

In order to better judge the cost-effectiveness of these scenarios, we implemented a very simple return on investment (ROI) calculation as a rough indicator of relative performance between scenarios. The simulation compares its total research output in a given time step to a funding-free scenario in which we calculate the research output of the current agent population if they were able to spend 100% of their time doing research only. Research outputs linked to funding – grant-related research quality bonuses and all postdoc research output – are removed. The ROI is then defined as the difference between the funded research output and the funding-free output, divided by the amount of funding disbursed. This gives us a measure of the amount of additional research purchased with each unit of funding.

Figure 8 shows a comparison of ROI for five promotion chance scenarios. Note that all results are in the negative – in other tests we also found that postdoc scenarios produced less research despite the increase in investment compared to the base case. Perhaps surprisingly, ROI becomes less poor in the higher promotion chance scenarios – so despite the additional cost, in a world with postdocs promoting more of them seems to produce dividends in terms of increased research output for the money spent.
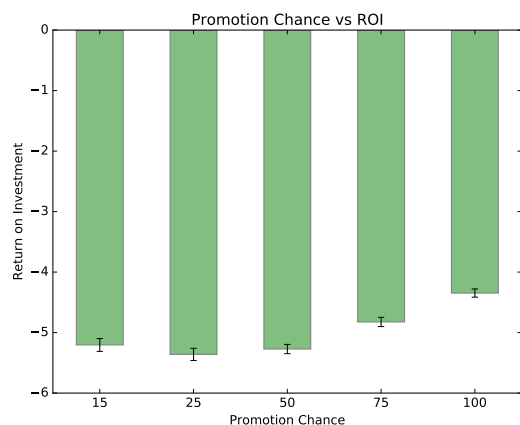

Figure 8: Results of ROI calculations for five postdoc promotion scenarios.

Discussion

While the core functionality of this simulation is relatively simple, understanding the complex agent behaviour and its consequences requires in-depth analysis. The multiple sets of scenarios presented here are intended to provide a relatively complete picture of the simulation outcomes across a range of parameter settings, and to give a comparison between the postdoc and non-postdoc scenarios.

In Scenario Set 1 we compared the postdoc scenarios with a growing population in which half as many permanent academics were hired. Notably, in every case the non-postdoc scenarios produced higher individual research productivity and higher total productivity. In Scenario Set 2 and 3 we examined two unique properties of the postdoc agents: their chance for promotion, and the job-hunting stress they feel toward the end of their contracts. We found that higher promotion chances lead to significantly higher research output, both individually and for the whole population. Unsurprisingly we found that higher stress leads to lower output – but the effects were surprisingly strong given the small size of the postdoc populations. The sensitivity analysis reinforces the results of Scenario Set 2, showing that postdoc promotion chance is driving the majority of the output variance in the postdoc scenarios.

These results lead us to conclude that in this simple model of postdoc careers in a competitive funding environment, the career path of postdocs has a significant impact on research productivity across the academic system. Postdocs end up accounting for a large fraction of the overall research output in the simulation while established academics get caught up in competing for grant funding, so the impact of job-related stress and poor mentoring is also felt across the population.

Unfortunately this does not bode well for real-world academia, as studies repeatedly confirm the poor mentoring and career development offered to postdocs around the world (Felisberti and Sear, 2014; Åkerlind, 2005). Postdocs regularly report significant anxiety about their career prospects, problems making ends meet financially, and a lack of career guidance and institutional support. The simulation shows that leaving postdocs to shoulder these burdens unsupported may have unexpectedly severe impact on our research productivity.

There are indications that this careers guidance aspect is being taken more seriously. In the UK institutions have signed up in large numbers to the Concordat to Support the Development of Researchers (Vitae, 2008), an agreement which calls on institutions and funders to develop strong frameworks for researchers' career development. However, this simulation suggests that offering supportive work environments and career advice may not be sufficient – maximising the sector's research potential would involve a more substantive rethink of the current state of academic careers and funding.

This simulation is only an abstract representation of the

funding and careers situation in academia, and should not be taken as a recipe for policy at this early stage. However, these results do give us a sense of the dynamics at work between competitive funding systems and postdoctoral researchers. In a system which highly incentivises senior academics to spend significant time on grant applications, postdocs are intended to fill the research gaps – but when those same postdocs' careers are tied to insecure short-term funding, research funders end up actually getting fewer outputs for their money.

Future work will need to examine these systems in more detail. At the moment, grants are represented very simply: one grant is much like another; and any single grant only attracts one postdoc. Further differentiation between types of research funding may help us develop some new methods of research funding disbursement that could alleviate some of these issues. Future versions of the model would also benefit from additional detail on the postdoc life course – postdocs in this simulation only have one contract and one attempt to achieve promotion, whereas in the real world postdocs often work on a succession of fixed-term contracts.

Similarly, the treatment and experience of postdocs varies significantly between countries, disciplines, and even between individual institutions – this model is based on the postdoc situation at research-intensive universities in the UK. In the real world postdocs may face a wide variety of obstacles depending on where they may be employed, which could substantially change their coping strategies. Understanding the lived experience of postdocs through quantitative and qualitative studies and incorporating this data into a more sophisticated decision-making model would allow for a more detailed representation of the varied postdoc career landscape.

While these results look dire for the postdoc scenarios, we do not believe this is due to unjust assumptions on our part. This model is relatively optimistic: more postdocs get promoted in the simulation than in many real-world academic systems; ROI calculations do not include costs like redundancy payments or training costs for new postdocs; and 30% of all grants are funded regardless of the number of applicants. Even in this relatively positive environment, postdoc scenarios still underperform compared to non-postdoc scenarios, and our return on investment is quite poor. While we reiterate that the model is too early to serve as a driver for substantive policy, we suggest that it provides food for thought when the academic community wishes to evaluate its performance, both as researchers and as employers.

Acknowledgements

References

Åkerlind, G. S. (2005). Postdoctoral researchers: roles, functions and career prospects. *Higher Education Research and Development*, 24(1):21–40.

Cook, I., Grange, S., and Eyre-Walker, A. (2015). Research groups: how big should they be? *PeerJ*, 3:e989.

Felisberti, F. M. and Sear, R. (2014). Postdoctoral researchers in the UK: A snapshot at factors affecting their research output. *PLoS ONE*, 9(4):e93890.

Fortin, J.-M. and Currie, D. J. (2013). Big science vs. little science: How scientific impact scales with funding. *PLoS ONE*, 8(6):e65263.

Geard, N. and Noble, J. (2010). Modelling academic research funding as a resource allocation problem. In *3rd World Congress on Social Simulation*. Event Dates: 6-9 September 2010.

Hoeller, K. (2014). *Equality for Contingent Faculty: Overcoming the Two-Tier System*. Vanderbilt University Press, Nashville, USA.

Kennedy, M. and O'Hagan, T. (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society, Series B*, 63(3):425–464.

O'Hagan, A. (2006). Bayesian analysis of computer code outputs: a tutorial. *Reliability Engineering and System Safety*, 91(10-11):1290–1300.

Powell, K. (2015). The future of the postdoc. *Nature*, 520:144–147.

Silverman, E., Bijak, J., Hilton, J., Cao, V. D., and Noble, J. (2013). When demography met social simulation: A tale of two modelling approaches. *Journal of Artificial Societies and Social Simulation*, 16(4):9.

University and College Union (2015). Making ends meet: the human cost of casualisation in post-secondary education. https://www.ucu.org.uk/stampout. Accessed 15/02/2016.

Van der Weijden, I., Teelken, C., De Boer, M., and Drost, M. (2015). Career satisfaction of postdoctoral researchers in relation to their expectations for the future. *Higher Education*, pages 1–16.

Vitae (2008). Concordat to support the career development of researchers. https://www.vitae.ac.uk/policy/vitae-concordat-vitae-2011.pdf. Accessed 17/02/2016.

Cultural wave front expansion explains multiple stages of diversity during the Neolithic Transition in Europe

Cornelis J. Drost[1] and Marc Vander Linden[1]

[1]Institute of Archaeology, University College London
n.drost@ucl.ac.uk

Abstract

The Axelrod model of cultural dissemination is a convenient analogue to the description of archaeological cultures based on a series of material features, such as styles of pottery, agriculture, domestication, etc. Allowing a population to spread into uninhabited, or sparsely inhabited, territory, while undergoing cultural interaction, generates a 'wave front' containing larger homogeneous cultures, with a backwater of diversity. A very similar process is observed in the neolithic transition - the arrival of the first farming technology at the end of the Mesolithic - in south-eastern Europe (c. 8000-6000 cBC), where the first observable neolithic cultures are large and homogeneous, and these are succeeded by greater diversity. The model presented here demonstrates how the dynamics of a spreading wave can explain the observed progression from large, spreading cultures to smaller, more diverse cultures.

Introduction

Archaeological cultures are commonly identified based on a set of material features common to sites belonging to that culture. Such features can include the types or styles of pottery, stone tools, agriculture, or animal husbandry used, among others (Clarke, 1968). For sites related to the neolithic transition in South-Eastern Europe, analysis of this material culture shows a progression from large, homogeneous and distinct cultures present initially (Figure 1a), to smaller, more diverse and inter-related cultures 3-500 years later (Figure 1b).

Data such as that for the latter sites can be difficult to interpret, as it does not allow cultures to be easily distinguished based on any one, or even several, material features. There is both a greater overlap between the features of spatially distant sites, and a greater variety in the features of nearby sites, when compared with data from the earlier sites.

As demonstrated by Bocquet-Appel et al. (2009), the neolithic transition is accompanied by a 'wave front' of increased population, that spreads gradually into Europe. Modelling this spread of a population into a previously unoccupied, or sparsely occupied, region suggests that the apparent progression from homogeneous to diverse culture is a natural consequence of the spreading process, rather than being an artefact of noisier data.

Theory

Assuming a culturally heterogeneous initial population that is bordered, in some or all directions, by unoccupied area, an expanding front will form along any boundary which is not enclosed. Whereas cultures in a saturated area may spread by interaction with neighbouring cultures, those on the front have access to unoccupied neighbouring sites, and are able to replicate themselves into that area without modification, allowing them to expand much more quickly than if they were enclosed.

Over a longer time period, the greater diversity behind the front also spreads, through the action of interaction rather than replication, and may erode the original front cultures. These diverse cultures both compete and mix with each other, forming an intermixed assortment of cultures: some growing larger; some being eliminated; and others dividing into partly related subcultures.

If the rate of spreading/replication is sufficiently high compared to the rate of cultural interaction, then large areas of homogeneous culture can form in the spreading front, faster than they are eroded by cultural interaction with dissimilar cultures behind them.

Unlike replication, this cultural interaction is not directed by the availability of space, and spreads much more slowly into the area occupied by the front. When this diversity does reach an area previously outside the initial area, possibly long after the front population, it will consist of several small-medium sized cultures, larger the further they've had to travel from the initial area.

The arrival of this chaos is visible in the CA results (Figure 1b) as both a greater diversity within closely spaced sites, and less distinction between distantly related sites - which may previously have been located in different culturally homogeneous streams.

Model

The cultural dissemination model of Axelrod (1997) provides a good analogue to the description of archaeological cultures based on assemblages of material features. This is adapted to the scenario of a spreading population by includ-
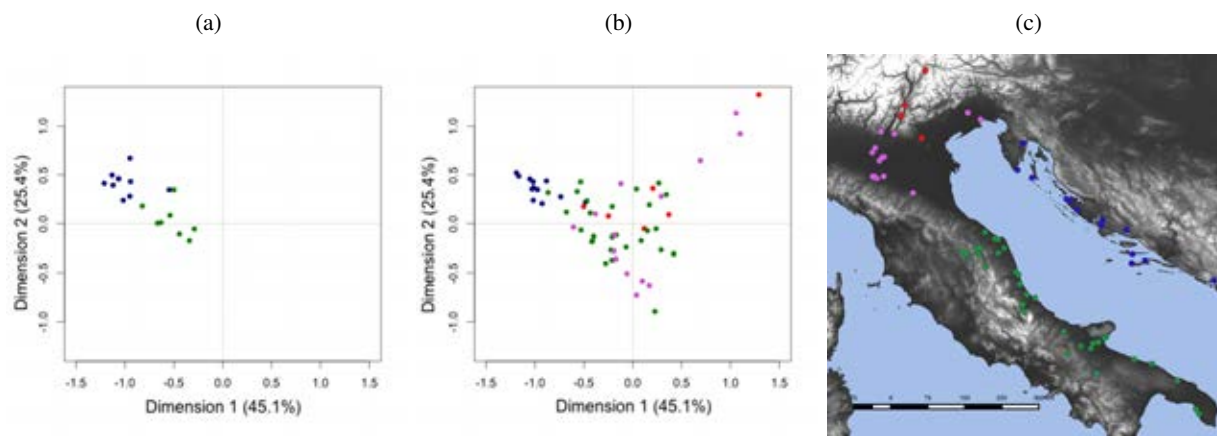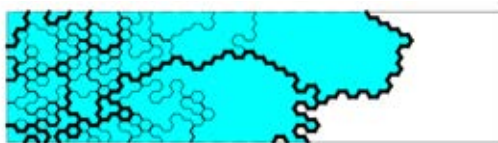
(a) (b) (c)

Figure 1: Correspondence analysis of animal species' distribution in zooarchaeological assemblages for (a) early (c. 6000-5400 BC) and (b) late (c. 5400-4500 BC) Neolithic. Points are colored by geographic region (c), not all sites are present in both plots (a) and (a).

(a) Large homogenous culture on wave front, after 90,000 steps

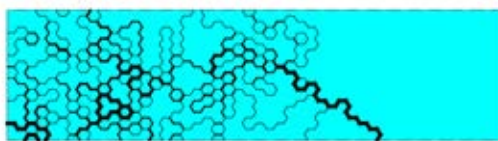(b) Diverse cultures erode territory of original front, after 240,000 steps

Figure 2: One realisation of the modified Axelrod model showing the spread of two large homogeneous wave front cultures (a), followed by their erosion by more diverse cultures (b). Line thickness indicates level of cultural dissimilarity, unshaded areas are unoccupied.

ing initially unpopulated loci, into which a culture can clone itself without modification, in a demographic step which occurs once for every N typical interaction steps.

A cultural interaction consists of comparing two randomly selected neighbouring sites, and with probability equal to their cultural similarity - the proportion of features for which they are equal - setting one of the dissimilar features in one site to the value in the other. A demographic step consists of randomly selecting two neighbouring sites, and if one is occupied and the other is not, cloning the culture of the occupied site into the unoccupied site. This simple addition results in the formation of the homogeneous front described above, along with a more diverse 'backwater', which gradually encroaches on the area first occupied by the front. Simultaneously, the cultures in this backwater are mixed, resulting in some cultures growing geographically to the exclusion of others.

Importantly, this model simulates the behaviour of culture, not people. Sites are either occupied or not, and need to be occupied to have culture. The spread of one culture into another does not imply any change in the population there, only a change in their culture.

The model is also neutral - no trait or feature is selected over any other - all interactions happen at random, as in the original Axelrod model. The only fitness differential arises from the ability of sites with unoccupied neighbours to clone themselves.

Conclusion

The ability for cultures on a spreading front to replicate themselves without mixing or modification, followed by the erosion of this front by more diverse culture over time, may offer the first process based explanation of the observed tendency from initial homogeneity to later diversity, common to many spreading phenomenon in archaeology.

References

Axelrod, R. (1997). The Dissemination of Culture A Model with Local Convergence and Global Polarization. *Journal of Conflict Resolution*, 41(2):203–226.

Bocquet-Appel, J.-P., Naji, S., Vander Linden, M., and Kozlowski, J. K. (2009). Detection of diffusion and contact zones of early farming in Europe from the space-time distribution of 14c dates. *Journal of Archaeological Science*, 36(3):807–820.

Clarke, D. L. (1968). *Analytical archaeology*. Routledge, London.

Cooperation and Reputation in Primitive Societies

Fernando P. Santos[1,2], Francisco C. Santos[1,2] and Jorge M. Pacheco[1,3,4]

[1]INESC-ID and Instituto Superior Técnico, Universidade de Lisboa, 2744-016 Porto Salvo, Portugal
[2]ATP-Group, Lisboa, Portugal
[3]Centro de Biologia Molecular e Ambiental da Universidade do Minho, 4710-057 Braga, Portugal
[4]Departamento de Matemática e Aplicações da Universidade do Minho, 4710-057 Braga, Portugal
fernando.pedro@tecnico.ulisboa.pt

Indirect Reciprocity (IR) is possibly the most elaborated and cognitively demanding mechanism of cooperation discovered so far. It involves status and reputations and has been heralded as providing the biological basis of our morality (Nowak and Sigmund, 2005). Whereas under direct reciprocity one expects to receive help from someone we have helped before, under IR one expects a return, not from someone we helped, but from someone else; in this sense, helping the "right" individuals may contribute to a reputation uplift that increases the chance of being helped, by someone else, at a later stage. This reputation shift depends on the socially adopted norm that defines what actions (and under which contexts) are reckoned as good or bad. Most theoretical models employed to date have studied how IR can lead to the emergence and sustainability of cooperation in infinite populations (Ohtsuki and Iwasa, 2006; Nowak and Sigmund, 2005). However, it is known that cooperation, norms, reciprocity and the art of managing reputations, are features that date back to primitive, small-scale societies While different features characterize a primitive society, here we take into consideration three of the most important: The evidence that interactions occur within small tribes, the central role played by reputations and the ease with which reputation information spreads within the tribe. In small populations, stochastic finite size effects are not only important, but may even render infinite populations analyses misleading (Imhof et al., 2005). Thus, it remains an open question which norms prevail in small-scale societies and their influence in the evolutionary dynamics of IR.

With the current extended abstract, we would like to summarize a new analysis of this problem. In Santos et al. (2016) we show that population size strongly influences the merits of each social norm, while proposing a new formal tool to assess the evolutionary dynamics of reputation-based systems in finite populations. We investigate to which extent norms found to promote cooperation in large populations will remain effective in small societies, and also to which extent the capacity of a social norm to foster cooperation depends on the community size. We consider a population of individuals who randomly interact in pairs through a donation game,

where one player is a potential provider of help (donor) to the other (recipient). The donor may cooperate and help the recipient at a cost c to herself/himself, conferring a benefit b to the recipient (with $b > c$); otherwise no one pays any costs nor distributes any benefits. Reputations are public and attributed by a bystander who witnesses a pairwise interaction. We adopt a world of binary reputations, Good (G) and Bad (B), which in our case are mere labels with no a-priori meaning. Their significance emerges in association with individual behavior in connection with the donation game. This binary reputation scheme, despite its formal simplicity, allows to consider a plethora of moral rules with variable complexity and it is specially amenable to a systematic mathematical treatment, in the framework of population dynamics. To perform an evaluation, the bystander uses a social norm, that is, a rule that converts the combined information stemming from the action of the donor and the reputation of the recipient into a new reputation for the donor. Social norms encoding this type of information are classified as 2nd-order norms (Ohtsuki and Iwasa, 2006). Four of these social norms have been given special attention (see matrices on Fig. 1): Stern-judging (SJ, also known as Kandori), which assigns a good reputation to a donor that helps a good recipient or refuses help to a bad one, assigning a bad reputation in the other cases (Pacheco et al., 2006); Simple-Standing (SS, similar to SJ, but more "benevolent" by assigning a good reputation to any donor that cooperates; Shunning (SH), similar to SJ but less "benevolent", by assigning a bad reputation to any donor that defects; and Image Score (IS, actually a first order norm) where all that matters is the action of the donor, who acquires a good reputation if playing C and a bad reputation if playing D (Nowak and Sigmund, 2005). In the space of 2nd-order norms that we consider, a duple p suffices to unambiguously define a strategy, by specifying the action directed at a G or B recipient. This leads to the following 4 possible strategies: unconditional Defection (AllD, $p = (D, D)$), unconditional Cooperation (AllC, $p = (C, C)$), Discriminator strategy (Disc, $p = (C, D)$), that is, cooperate with those in good reputation, and defect otherwise), and paradoxical Discriminator strategy (pDisc,
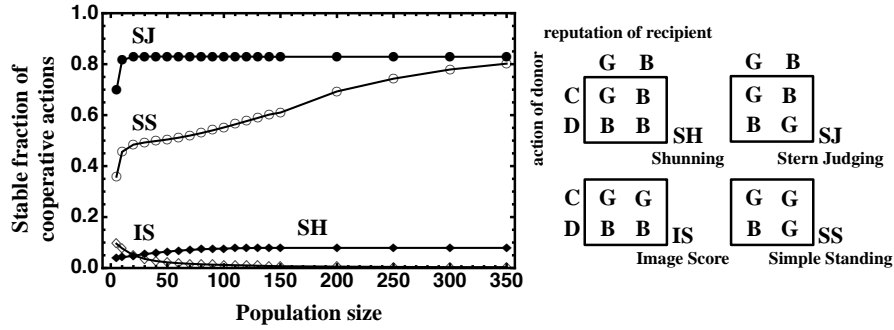
Figure 1: Stern-judging (SJ) is able to foster the highest rates of cooperation, independently of the (finite and small) population size; Differently, the efficiency of SS in fostering cooperation strongly depends on the population size and on the error rate committed by individuals (Santos et al., 2016). SH harms cooperation by being too strict compared to SJ due to the abusive widespread assignment of bad labels. See main text and Santos et al. (2016) for details on the strategic dynamics induced by each social norm. The matrices illustrate the 4 dominant social norms in terms of the new reputation (B/G, inside each square) attributed to a donor given its action (C/D, rows) and the reputation (B/G, columns) of the recipient.

$p = (D, C)$, the opposite of Disc). Unlike previous studies, in Santos et al. (2016) we investigate the evolutionary dynamics of these 4 strategies within finite populations by means of a stochastic birth-death process, both analytically and through large-scale computer simulations (for details, please see Santos et al. (2016)). As detailed in Fig. 1, we show that SJ clearly stands out for small population sizes, dominating with SS for large population sizes. Indeed, it can be shown that only SJ and SS are able to combine a high prevalence of an ALL-Disc configuration with the incidence of Good reputations in this configuration, efficiently fostering high levels of cooperation (Fig. 1). Yet, in small-scale societies, SJ significantly promotes more cooperation than SS, as the latter fails to prevent the invasion of unconditional defectors (AllD) in small populations. On the other hand, SJ fosters an ideal coordination between strategy and prevailing reputations, assuring the stability of configurations where individuals cooperate in the donation game. Indeed, the high degree of symmetry of SJ allows the promotion of cooperation, irrespectively of the emerging meaning of G and B labels, also allowing "paradoxical" strategies to prevail and promote cooperation. These results remain valid for a wide interval of reputation assignment time-scales, errors of execution and reputation-assignment inaccuracies.

To conclude, a single social norm (SJ) emerges as the leading norm in small-scale societies. That simple norm dictates that only whoever cooperates with good individuals, and defects against bad ones, deserves a good reputation. Remarkably this pattern is consistent with recent empirical results (Hamlin et al., 2011) showing that toddlers positively evaluate i) those who treat others prosocially, ii) those who behave negatively towards those who have acted antisocially, and iii) puppets that harm antisocial puppets. This said, behavioral experiments in this context remain a

vast open territory and very active area of research. Finally, our modeling framework has the advantage of being naturally extendable to social norms of higher order, enlarging the complexity of the norms studied to date. Work along these lines is in progress, with promising preliminary results of interest within the area of evolution of biological complexity, and the ALife community in general.

Acknowledgements

References

Hamlin, J. K., Wynn, K., Bloom, P., and Mahajan, N. (2011). How infants and toddlers react to antisocial others. *Proc Natl Acad Sci USA*, 108(50):19931–19936.

Imhof, L. A., Fudenberg, D., and Nowak, M. A. (2005). Evolutionary cycles of cooperation and defection. *Proc Natl Acad Sci USA*, 102(31):10797–10800.

Nowak, M. A. and Sigmund, K. (2005). Evolution of indirect reciprocity. *Nature*, 437(7063):1291–1298.

Ohtsuki, H. and Iwasa, Y. (2006). The leading eight: social norms that can maintain cooperation by indirect reciprocity. *J Theor Biol*, 239(4):435–444.

Pacheco, J. M., Santos, F. C., and Chalub, F. (2006). Stern-judging: A simple, successful norm which promotes cooperation under indirect reciprocity. *PLoS Comput. Biol*, 2(12):e178.

Santos, F. P., Santos, F. C., and Pacheco, J. M. (2016). Social norms of cooperation in small-scale societies. *PLoS Comput. Biol*, 12(1):e1004709.

Nobility-targeting raids among the Classic Maya: Cooperation in scale-free networks persists under tournament attack when population size fluctuates

Roberto Ulloa[1], and Tom Froese[2,3]

[1] Department of Modern Languages and Literatures, University of Western Ontario, London, Canada
[2] Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas, Universidad Nacional Autónoma de México, Mexico
[3] Centro de Ciencias de la Complejidad, Universidad Nacional Autónoma de México, Mexico
roberto.ur@protonmail.com

Abstract

Cooperation in scale-free networks has proven to be very robust against removal of randomly selected nodes (*error*) but highly sensitive to removal of the most connected nodes (*attack*). In this paper we analyze two comparable types of node removal in which the removal selection is based on tournaments where the fittest (*raids*) or the least fit (*battles*) nodes are chosen. We associate the two removals to two types of Maya warfare offences during the Classic period. During this period of at least 500 years, political leaders were able to sustain social order in spite of *attack*-like offences to their social networks. We present a computational model with a population fluctuation mechanism that operates under an evolutionary game theoretic approach using the Prisoner's Dilemma as a metaphor of cooperation. We find that paradoxically *battles* are able to uphold cooperation under moderate levels of *raids*, although *raids* do have a strong impact on the network structure. We infer that cooperation does not depend as much on the structure as it does on the underlying mechanism that allows the network to readjust. We relate the results to the Maya Classic period, concluding that Mayan warfare by itself cannot entirely explain the Maya political collapse without appealing to other factors that increased the pressures against cooperation.

Introduction

An intriguing peculiarity of the Mayan warfare during the Classic period was the corporeal involvement of the elites, especially because their most relevant members, i.e. *nobility*, often became direct targets of offences in the form of *raids* (Webster, 2000). This raises the question what impact nobility losses would have had on government, and how the political class of the elite organized itself in order to keep the necessary cooperation required to sustain social order. The types of elite casualties rendered by Mayan warfare resemble scenarios that have been studied in the literature of social networks analysis, specifically when two types of node removal, called *error* and *attack*, are analyzed in scale-free (*sf*) networks (Albert et al., 2000), i.e. networks in which distribution of the number of connections of each node follows a power law; therefore they are considered highly heterogeneous networks. On one hand, the random removal of nodes (*error*) can be equated to casualties of the general population, including lower ranking elites that fought as warriors. On the other hand, removal of the most connected nodes (*attack*) can be likened to specific nobility-targeting raids, in which the most central members of the social hierarchy were the main victims. Previous studies revealed that although the *sf* structure and cooperation are very robust to *error*, they are highly vulnerable to *attack* (Callaway et al., 2000; Cohen et al., 2000, 2001; Perc, 2009). At the same time, some research has suggested that there is a strong relation between network structure and cooperation, and, in particular, that the heterogeneity of a network drives cooperation (Ichinose et al., 2013; Santos et al., 2012).

In this context, we find the Classic Maya period puzzling because it constitutes a counterexample: a warfare involving *attack*-like victims such as city rulers (Martin and Grube, 2008), and yet exhibited a relatively stable social and political organization that maintained complex levels of social order for at least 500 years until widespread political collapse around 800AD (Webster, 2002). We also know that there was a drift towards a less hierarchical political structure (i.e. less heterogeneous) during the Classic period (Jackson, 2013), and this contradicts previous findings which suggest that the loss of network heterogeneity should have brought about a decline in cooperation, but didn't, at least for the indicated time span.

We attempt to clarify these contradictions by adapting a computational model from (Miller and Knowles, 2015a, 2015b) and including similar mechanisms to *error* and *attack*; these mechanisms (called *battles* and *raids* respectively) differ in that they involve tournaments of a randomly selected group of nodes. In this tournament, a loser *(for* *battles)* or a winner (for *raids*) is selected based on its fitness score that is calculated according to the performance of that node playing the Prisoner's Dilemma (Nowak and May, 1992). Thus, Miller and Knowles' model allows for fluctuation of network size (i.e. growing by addition of nodes up to a maximum number and then shrinking by means of removal of a certain number of nodes via a series of tournaments), which turns out to promote cooperation.

Metaphorically, we can imagine that our model represents the network structure of the elite fraction of a given Maya polity. A random subset of these elite individuals will form bands of warriors, which engage in repeated battles to protect their city from attacks and to attack other cities. Who is most likely to perish in these battles depends to a large extent on whether the nobility is the main protagonist (and target) of these battles, or whether it is mainly a matter of survival of the fittest (i.e. death of the least fit).

Our results show that the nobility-targeting *raids* (don't immediately affect cooperation in the overall network of elite individuals, although they change the network structure towards a less hierarchical topology. This result supports the theory that Maya society could have sustained moderate levels of systematic warfare that involved the elite for an indefinite time. A collapse can, however, be explained by a significant increase of warfare, such as occurred in the Petexbatun region (Webster, 2000), or, alternatively, by a secondary factor that introduced pressure on the living conditions at the end of the Classic period, e.g. an environmental change, economic crisis or land degradation. Our results are also consistent with archeological observations of a shift from highly centralized to a more distributed form of network structure that occurred in the Classic period (Jackson, 2013). Therefore, assuming that the archeological account of Maya warfare is correct, our model suggests that nobility-targeting may be an explanation of the structural changes in the political system of the Maya.

Mayan warfare: *battles* and *raids*

Archaeological findings suggests that the inhabitants of Mayan cities and in particular their elites were involved in constant warfare (Webster, 2000). Evidence of warfare can be found in the Preclassic period (2500 – 250 BC)[1], but most of the conflict is recorded to have developed during the Late Preclassic (400 BC – AD 250) and Classic periods (AD 250 – 1000) (O'Mansky and Demarest, 2007). In the Early Classic period (AD 250–600), warfare was characterized by small, sporadic raids, with their main objective seemingly the capture and subsequent sacrifice or imprisonment of nobility (highly important members of the elites). This practice intensified across the Late Classic period (AD 600–800) (O'Mansky and Demarest, 2007), and culminated in numerous city sackings and burnings during the Terminal Classic period (AD 800–1000) (Normak, 2007).

The Maya elites were both part of the attacks (as warriors, i.e. "soldiers") and their main targets (Webster, 2000). For example, evidence from Aguateca (Petén, Guatemala) indicates that in one particular war, which was carried out to eliminate another Maya state (AD 810), members of the elite made up the majority of warriors (Aoyama, 2005). Additionally, warfare-related art and inscriptions emphasize high-ranked individuals (Stuart, 1993; Van Tuerenhout, 2001), and this has led some archeologists to argue that wars were fought between the elites exclusively (Freidel, 1986). This would imply a small numbers of warriors, a maximum of 600 to 1000 for Tikal, which was one of the largest Mayan cities (Hassig, 1992), or 500 to 600 for Copan, for which very accurate demographic estimates exist (Webster et al., 2000, 1992). Therefore some researches have argued that the war forces must have also involved commoners, however the direct involvement of the elites is not disputed (Webster, 2000).

Additionally, the nobility members were often the main targets of the attacks. For example, the capture and sacrifice of the ruler of Copán (Honduras) by Quiriguá (Guatemala) in AD 738, the capture and unknown fate of the ruler of Tikal (Petén) by Caracol (Petén) in AD 562, the capture and vassalage of the ruler of Seibal (Petén) by Dos Pilas (Petén) in AD 735, or the capture of the ruler of Naranjo (Petén) by Tikal

in AD 744 (Martin and Grube, 2008). These examples are clearly some of the most important since they were direct attacks to the main ruler, however the capture and sacrifice of enemies was a common practice as it also has been associated to status rivalry, i.e. competitive behavior exhibited by elite members to increase their status (O'Mansky and Demarest, 2007): they fought to assert their roles in society and their areas of influence (for example, in their roles in the royal courts) by means of war merits. We can safely assume that the higher the captives' rank, the higher the merit.

The dual role of individuals as warriors and elite members (or work force) holds true as no evidence of standing armies has been found (Van Tuerenhout, 2001). Beyond taking part in warfare, residents of Maya cities must have had other responsibilities including the elite political roles of the nobility; the loss of these individuals due to warfare would then imply changes in the structure of the elite social network.

We stress a distinction between two warfare scenarios: (1) *raids* with the goal to capture (and often sacrifice) nobility members, and (2) relatively large-scale *battles* between sites (although in reality they are not mutually exclusive). The two scenarios would have resulted in different outcomes: in the first scenario, no matter whether attackers or defenders emerged victorious, it would result primarily in *nobility victims* (presumably the attackers, a select group of skillful warriors, were also relevant members of the elites, considering the association of status and warfare recognition). In the second scenario, a high number of elite members probably died in combat (*casualties*). However, in this case, we argue that most of the victims were less important members of the elite since the nobility, if participating, should have enjoyed some extra protection during combat, e.g. they probably would not have been fighting in the frontlines.

With the present research, we investigate the consequences of the two described types of warfare, in particular the effects of elimination of either nobility, or of less influential members of the elite, on the structure and functioning of elite society. First, we hypothesize that the removal of nobility would have impacted both the elite's social structure, as well as the ability of the government to exert its function (as measured by the extent of cooperation), whereas the removal of less influential members would be less disruptive.

Second, we make a more specific prediction related to how the network structure is expected to change. In the case of the Maya, there was a transition towards a less hierarchical structure among the elites in which the ruler was gradually losing power during the Classic Period (Jackson, 2013), and this transition coincides with the increase of Maya warfare across the Classic Period. We therefore hypothesize that the increase in nobility-targeting warfare, in the context of normal fluctuations in population size, facilitated the emergence of less centralized political structures, as represented by the increasing importance of the Mayan royal court.

Related work: *Error* and *attack* on *sf* networks

In order to study the Mayan warfare scenario, we are modeling the interaction among elite individuals with the Prisoner's Dilemma (*PD*), a widely used representation of social dilemmas, i.e. a situation in which the individual success

---

[1] All the time periods are based on (Webster, 2000)

(expressed as reward, or fitness in evolutionary terms) calls for actions that harm collective wellbeing, and which therefore implies that the emergence of cooperation from selfish individuals is paradoxical (Axelrod, 1984). For this reason, the *PD* serves as a metaphor of the elites' capacity to take decisions that could lead their city to prosperity, as opposed to simply personal reward. Regardless if the elite's cooperation involves corruption or not, it would be impossible to keep centralized power to sustain social order if the members of the elite don't cooperate among them.

We will also investigate whether the social network structure of the elites serves as a mediator for levels of cooperation. In order to model the Maya elites' social structure, we use scale-free (*sf*) networks, i.e. networks in which the degree (i.e. *k*, the number of connections of one node to other nodes) distribution of the nodes follows a power law distribution, generated by evolutionary preferential attachment growth, where a new node attaches to an old node according to its fitness based on the outcomes of several rounds of the *PD*, one round per neighbor (Poncela et al., 2008). Due to their heterogeneous structure, *sf* networks have proven to be suitable models of other archaeo-logically inferred social networks (Brughmans, 2012). The small size of the Maya elite, estimated at 1% or 2% (Adams and Smith, 1977) implies a high concentration of power in a few nodes which is also consistent with the node's degree distribution of *sf* networks and the rich get richer nature of preferential attachment. This will serve as the starting point in our simulations, after which we will perform systematic attrition of the two different types of elite members (where (1) fittest nodes represent influential elite members, i.e. nobility, and (2) less fit nodes represent less influential members of the elites). We will then analyze the effects of this attrition on cooperation within the social network, and on its structure.

Application of the proposed methodology extends existing research on the structure tolerance of *sf* networks to *error* and *attack*, and its relation to cooperation. In terms of tolerance, the structure of *sf* networks has formally proven to be resilient against random removal of nodes, i.e. *error*; however, it was sensitive to removal of the most connected nodes, i.e. *attack* (Callaway et al., 2000; Cohen et al., 2000, 2001). In terms of cooperation, *sf* networks have been shown to promote cooperation (Santos and Pacheco, 2005) and it is robust to *error*, but it quickly decreases under *attack*, and therefore the decrement has been linked to a decline in the network heterogeneity (Perc, 2009); although the link is less strong in dynamic networks (Ichinose et al., 2013; Poncela et al., 2008).

Previous simulations of these processes are concentrated on the use of preferential attachment, where a new node attaches to an old node according to its degree only (Barabási and Albert, 1999). Diverging from this, we will instead investigate evolutionary preferential attachment (Poncela et al., 2008) as it includes the nodes' performance (fitness) playing PD to decide the attachment of new nodes. Moreover, although cooperation has shown to be more robust in dynamic networks, to our knowledge none of the previous studies on *attack* have focused on an underlying mechanism of growth and shrinkage of the network based on the node fitness, such as the fluctuating model of (Miller and Knowles, 2015a, 2015b). In their model cooperation increased under attrition of nodes that were chosen by applying a probability that favored the nodes with least fitness (and indirectly less connections).

In our application of their model, we would like to propose that the low attrition levels from Miller and Knowles (2015) can approximate the normal expected mortality rate among the elite, and that higher levels of attrition would then correspond to an increase of mortality due to *casualties* of warfare. More specifically, since attrition in Miller and Knowles (2015) was directed at the least fit, it would be representative of large-scale *battles* where mostly the less relevant members of the elite died, in contrast to *raids* conducted with the explicit aim of capturing or killing the nobility (most fit).

Therefore, we first implement the Miller and Knowles model and replicate their results, focusing on interpreting the data within the Maya context. After this, we extend the model to test the effects of the removal of the fittest nodes (i.e., *raids*) when the fluctuation system (set at different *casualty* levels) is still present. In this case, we reverse the attrition's selection probability to now address the nodes with high fitness, which also tend to be the most connected ones since a higher fitness is more probable with a high number of connections. We can assume that in both scenarios non-elite members of society were also negatively affected, but this is unlikely to have impacted the structure of government and is not modeled explicitly According to the existent theory, removing a few highly connected nodes of the social network will have a bigger impact than the removal of many of the less connected ones. However we will show that the fluctuation in network size will produce an equilibrium of network structure that allows the persistence of cooperation.

Methods

Our model is based on the fluctuation models described by Miller and Knowles (Miller and Knowles, 2015a, 2015b) which comprise alternating growing and shrinking phases (i.e. *battles*, attrition of some of the least fit members) of the population. We have additionally included *raids*, a mechanism of attrition of some of the fittest nodes based on a tournament selection that is analogous to theirs except that it selects the fittest nodes; both attritions can operate constantly but at different rates. Similar to theirs, our simulations keep a population size of around 1000 agents (1009 is the maximum) because, given that elites among the Maya are estimated to represent 1% to 5% of the population, 1000 elite members would correspond to a total population of between 20000 and 100000, which agrees with population estimates for Mayan cities.

	B: Cooperate	**B: Defect**
A: Cooperate	1 \ 1	0 \ b
A: Defect	b \ 0	0 \ 0

Table 1: Payoff matrix for the weak Prisoner's Dilemma. Column 1 shows player A's strategy, and row 1 shows player B' strategy. The payoff of the combination of A and B strategies are shown in the middle cell as A's payoff (blue) \ B's payoff (red), where *b* represents the temptation to defect.

An edge between two agents (nodes) represents that they know each other, and therefore it exists the possibility of an interaction between them: an engagement in the weak version of the Prisoner's Dilemma (*PD*) game (following Miller and Knowles' model implementation), in which each agent obtains a payoff according to its own strategy and the strategy of its

rival. Table 1 shows the payoffs that agents A and B obtain according to the two possible strategies that they can play, i.e. cooperate or defect, as formulated in (Nowak and May, 1992). We can imagine that cooperating nodes that have the largest numbers of connections to other cooperators represent high-status nobility among the elite.

The only parameter b represents the temptation to defect. In principle, the dilemma only exists when $b > 1$ because the strategy that gives the biggest payoff to one agent depends on what the other plays. Otherwise, for $b < 1$, the only rational solution to obtain the maximum possible payoff, regardless the other agent's strategy, is to cooperate. The temptation to defect (b) represents how competitive the situation is, e.g. it could represent the lack of water, in which case people would try to get as much as possible of it only for themselves before it runs out instead of sharing it with their group.

Following Miller and Knowles, all the simulations start with one of two extreme cases of 3 agents that are either all cooperators (CCC) or all defectors (DDD). This enable us to observe the response of the model under the best and worst case starting conditions. An iteration (t) of the simulation consists of five steps:

1. Play PD. In each edge of the network, PD is played between the two connected agents (neighbors of each other) representing an interaction between two elite members. This results in each agent playing against all its neighbors once, and accumulating a fitness score equivalent to the sum of the payoffs (r) obtained in all the games:

$$f_i(t) = \sum_{j=1}^{k_i(t)} r_{i,j} \qquad (1)$$

2. Update Strategies. Updating of behavioral strategies is based on imitation of the most successful elite members, and the implicit rule: cooperate with cooperators (or, defect with defectors) if they are performing better. Each node i in the network randomly selects another node j from its neighbors. If the fitness of node i (f_i) is less than the fitness of the neighbor j (f_j), then the node i will change its strategy to the neighbor's according to the following probability:

$$P_i = \frac{f_j(t) - f_i(t)}{b * max(k_i, k_j)} \qquad (2)$$

This probability is proportional to the difference between the nodes fitness scores; therefore agents that produced very low fitness compare to their selected neighbor are more likely to copy the neighbors' strategy. In order to obtain a probability, the denominator normalizes the fitness according to the maximum possible difference between the two nodes given their current degree (k).

3. Grow network. In each iteration the elite will grow including new members, newborns, kin or outstanding/skillfull commoners. 10 new nodes with a randomly selected strategy (C or D) are connected to the network by 2 edges that are created according to the evolutionary preferential attachment mechanism (Poncela et al., 2008). An existing node i will be connected through one of the two edged to the new node according to the following probability:

$$\Pi_i(t) = \frac{1 - \varepsilon + \varepsilon f_i(t)}{\sum_{j=1}^{N(t)} (1 - \varepsilon + \varepsilon f(t))} \qquad (3)$$

Here, $N(t)$ is the number of nodes available to connect, not including neither any of the 10 new nodes that are being added in this step nor any existing node already connected to the new nodes (i.e. without replacement), and $\varepsilon \in [0,1[$ is a parameter that adjusts the selection pressure, i.e. the lower the selection pressure ε the more probable is that a non-well-fit node will get a connection to a new node. In all our simulations we have set a high selection pressure of $\varepsilon = 0.99$, favoring the evolutionary preferential attachment process.

4a. Battles. We changed the name attrition (used by Miller and Knowles) to *battles* to easily distinguish it from the attrition of relevant nodes, i.e. *raids* (Step 4b). If the population reaches a size bigger than a specific value (1000 in our simulation), then the network is shrunk by $C\%$ of nodes. Each of these nodes was the loser, i.e. the member with the least fitness, of a tournament of S participants in which the payoffs were compared. The participants were a randomly chosen 1% of the population, $S = 1\% \times N(t)$. In case of ties, the loser is selected randomly from among the ones that tied. The tournament is performed as many times as necessary to have a group of losers that is equivalent to the $C\%$ of the population. Then they are removed from the network together with their edges. Any disconnected nodes resulting from this process are also removed. We note that removals caused by *battles* resemble *casualties* (C), or generally speaking mortality, during warfare in which the elite were involved, in which the least fit members were more likely to die. We also point out that when the tournament involves one participant ($S = 1$), *battles* are equivalent to *error* (random removal of nodes). Additionally, when $S > 1$, *battles* always selects among the least fit nodes (in the worst case, the S-th fittest), whereas *error* does it the majority of the time as in *sf* networks the distribution of fitness, as it is for connections, is expected to be unbalanced, i.e. very few nodes will concentrate most of the reward being less likely to be selected in a randomly uniformed process. The tournament avoids the selection of the fittest nodes as *raids* (Step 4b) will be responsible of this selection.

4b. Raids. All the previous steps (i.e. steps 1.-4a.) are equivalent to those described in Miller and Knowles (Miller and Knowles, 2015a, 2015b); but *raids* are an extension of step 4a that we are adding to study the impact of nobility-targeted *raids*. As step 4a, *raids* resemble existing game theoretic nomenclature, i.e. *attack*, except it also contains a tournament component. In contrast to the previous steps (1.-4a.) which are performed every iteration, *raids* are performed each T iterations, i.e. *frequency* of *raids* of $F = 1 / T$. This means that conflicts in which nobility are expected to die occur relatively less frequently compared to the number of deaths caused by generalized warfare. The selection mechanism for the *nobility victims* ($V\%$ of population size), is analogous to the selection mechanism in the *battles* step (Step 4a.); except that in this case instead of losers the winners are removed, a winner of a tournament is the one that has the most fitness instead of the least. As with *battles*, any disconnected nodes resulting from this process are also removed.

The main response variable of this model is the percentage of cooperators (i.e. agents that have the 'cooperate' (C) strategy) in time step t. To analyze the effects of *battles* (Step 4a) we implemented our own version of this model. After getting statistically different results - although qualitatively similar - we compared Miller and Knowles' code (provided by the

authors) with ours, and found an important difference in the way the strategies were updated. Their implementation was updating strategies asynchronously, i.e. each agent would update its strategy s_i with a copy of the neighbor's strategy s_n as soon as they met the conditions of Step 2. This produces a situation in which an agent may transmit the updated strategy s_n instead of the original s_i, which according to our criteria should be the correct one, because s_i is the strategy the agent used to obtain his current reward associated to Equation 2. Instead, our implementation updates the strategies synchronously, i.e. each agent first evaluates which should be its new strategy s_n without changing their current strategy s_i until every agent knows their new strategy s_n for the next iteration.

Maya warfare: an experimental application

Is cooperation reduced by an increment of *battles* during warfare? Miller and Knowles results indicated that *battles* able to increase cooperation compared to the absence of *battles* when (1) the simulations started with a defector founded network (DDD) and (2) *battles were* set at a low level and the simulation started with cooperator founded networks (CCC). However, in both scenarios, there was an inverse relationship between *battles* and cooperation, i.e. the higher the *casualties* (*C*) in *battles* the less cooperation was achieved. In other words, a very small *battles* levels are able to boost cooperation but higher values start to negatively impact cooperation although at a rather slow rate.

Their study explored values of *C* from 0% (no *battles*) to 50% (Miller and Knowles, 2015b). In our first experiment, we decided to expand this to values from 0% to 90% in increments of 10% while keeping the same values for the temptation *b* (1.0, 1.3, 1.6, 1.9, 2.2, 2.5, 2.8, and 3.1). We also include interesting values of 0.1%, 0.5%, 1%, 2.5% and 5% because they approximate realistic figures based on current average annual mortality in different countries, including highly violent ones (United Nations, 2013). Our first experiment will (1) validate our simulation, (2) report the new results after the correction procedure for synchronically updating strategies, (3) further confirm the inverse relation between *casualties* and cooperation, and (4) provide a comparison point for our second experiment.

How cooperation is affected by the nobility-targeting *raids* in scenarios of different *casualties* (*C*) rates? Our second experiment includes *raids* (Step 4b) as part of the iteration. For *nobility victims* (*V*), we explore the values 0.1%, 1% and 10%, whereas for its *frequency* (*F*), we explore the values 1/10, 1/20, 1/40 and 1/80. The values were selected according to exponential sequences for a broad exploration; for *V*, the sequence corresponds to $(10^{-k})_{k=1}^{3}$, and for *T* (*F*=1/*T*), to $(10 \times 2^{k})_{k=2}^{3}$. We also explored the model with different levels of *C* (*casualties*): 0 (no *battles*), 0.1, 0.25, 0.5, 0.75, 1.0, 2.5, 5, 7.5, 10, 20, 30, 40, and 50. Finally, we kept the same values of the first experiment for the temptation *b*.

Each configuration of parameters (scenario) runs for 2000 iterations and it is repeated 50 times in order to reduce random effects. The two experiments allow us to explore four different conditions for cooperation to emerge: (1) no attrition (*battles* or *raids*), (2) just *battles*, (3) *raids* without any *battles*, and (4) the combination of *raids* and *battles*.

Results

The results of our first experiment were different from those obtained by Miller and Knowles (Fig 1, 2a) due to our synchronous mechanism of updating strategies (compared to their asynchronous mechanism), however qualitatively speaking the results are very similar and their conclusions hold. Figure 1 confirms that for scenarios that started with a group of defectors (figure 1A), *battles* strongly favor cooperation but, for scenarios that started with a group of cooperators (figure 1B), only small amounts of *casualties* improves cooperation. Since the two figures (1A and 1B) are very similar for C>0.5, it seems that *battles* eliminate the influence of the starting state.
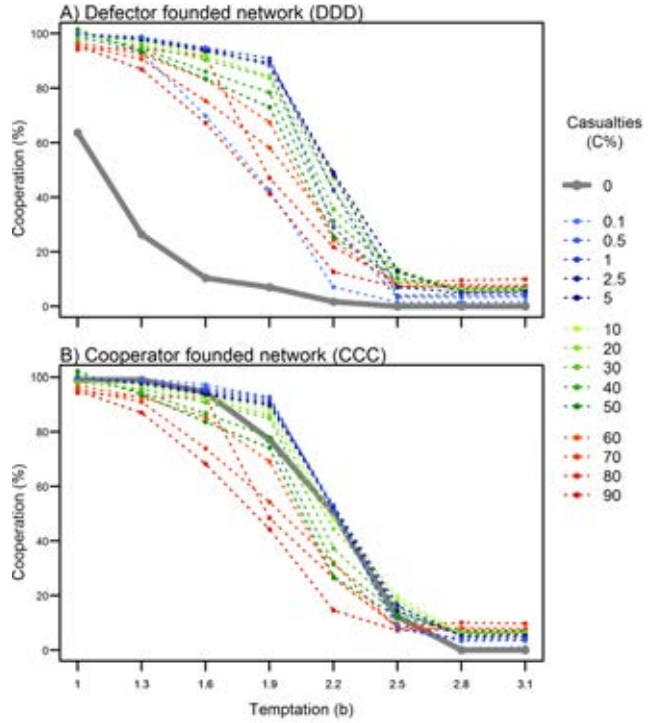
Figure 1: Average percentage of cooperators for different levels of *casualties*. X-axis displays levels of temptation to defect (b); Y axis displays the average percentage of cooperators calculated. Each line color indicate one rate of *casualties*, including 0% that serves as a baseline and it is highlighted with a ticker gray line. Data points are averages of the last 20 iterations (of 2000) for the 50 repetitions.

At the same time, we can also observe that when we further increase *casualties*, cooperation decreases; however the rate of decrement is slow; levels of *casualties* below or equal to 30% (*C < 30*) are able to hold similar cooperation compared to no *casualties* (*C = 0*) with cooperator founded networks (CCC). For defector founded network (DDD), all levels of *casualties* proved to be better than no *casualties*. We did find that the lowest level of *casualties* (0.1%) appears to be insufficient to raise cooperation to the highest levels (lightest blue dotted line in figure 1A)

The results obtained in the second experiment further proves the benefits of *battles* in terms of holding cooperation; our model is able to sustain cooperation when we systematically remove the fittest nodes of the network (*raids*). In figure 2, we

Figure 2: Cooperators for different levels of Elite Attrition. The first and second rows present graphs for scenarios without *casualties* (C = 0, top row) and with casualties (C = 2.5, bot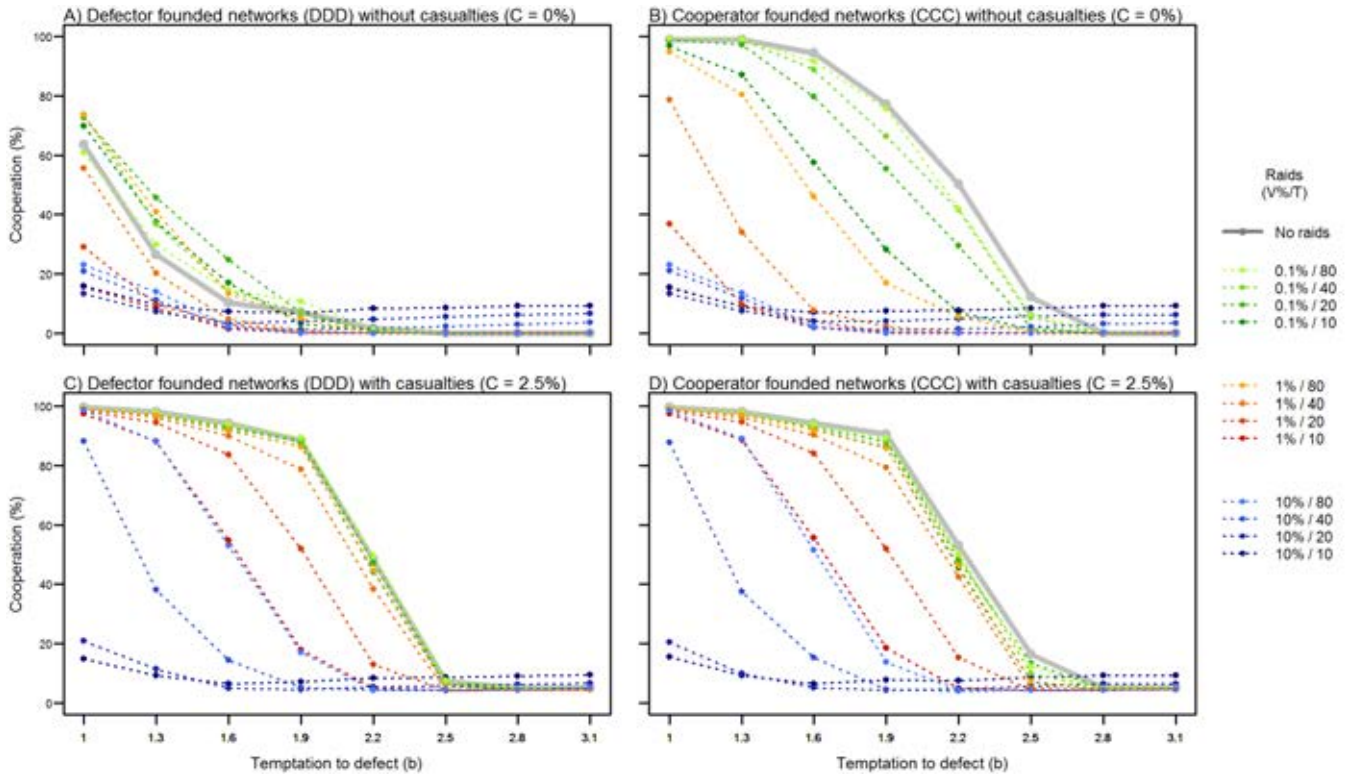tom row), for scenarios with cooperator (left column) and defector (right column) founded networks. The casualties (C) for all graphs is set to 2.5%. The legend shows the different levels of *nobility-victims* (*V*) and its *frequency* (*F*), i.e. V% / F. The baseline is the case in which there is no *raids*. Data points are averages of the last 20 iterations (of 2000) for the 50 repetitions.

show the results obtained for cooperator and defector founded networks (columns), and for scenarios without *casualties* (C = 0) and with *casualties* of 2.5% (C = 2.5). We picked this value arbitrarily because we found that any other values of *casualties* between 0.5% and 20% showed very similar results (data not shown). *Casualties'* levels below 0.5% (C < 0.5) are able to sustain cooperation but not as well as the shown in figure 2; whereas a steady decline of cooperation is observed for *casualties* above 20% (C > 20).

The benefits of *casualties* become very evident when we look into defector founded networks (figures 2A and 2C); in fact, cooperation is boosted almost as much as if the network would have been founded by cooperators (with C = 2.5%) as results in figures 2C and 2D are hardly distinguishable between each other. There are also substantial benefits of *casualties* for cooperation in cooperator founded networks. When we compare directly the different levels of *raids* (individual dotted lines) between figure 2C and 2D we observe that cooperation holds much better when *casualties* are present, e.g. even the lowest rates of *raids* (green lines) affect cooperation in the scenario without *casualties*, whereas it takes middle rates (red lines) of *raids* when *battles* are present.

In terms of network structure, we should expect some changes since we are trimming relevant (connected) nodes of the networks. In figure 3, we illustrate this changes in a qualitative approach based in examples that shows the internal behavior of the model for two interesting scenarios where the temptation to defect is at a safe value (b = 1.6), i.e. we observe
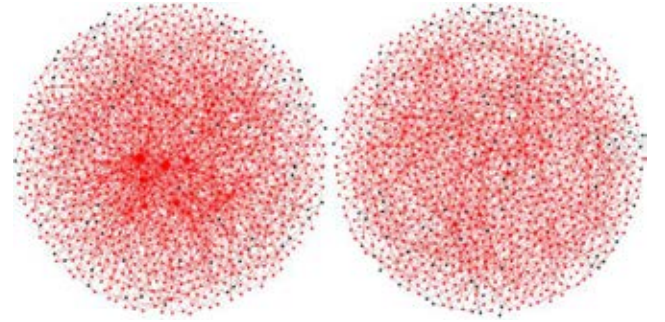
Figure 3: Network structure without *raids* and with *raids*. Two networks obtained after 2000 generations from one run of the simulation (arbitrarily the 1st run out of 50 repetitions) of two scenarios: left, without *raids,* and right, with *raids* (E = 0.1% and F = 10), both with, C = 2.5%, b = 1.6 and defector found network (DDD). The red dots represent cooperators, and the black defectors. The node size is proportional to the most connected node of both scenarios (k=89), therefore size is comparable across graphs.

a clear convergence towards cooperation. In a defector founded network (DDD), we compare the first repetition (out of 50) that was performed without *nobility victims*, V = 0%, (left), and the first repetition from the ones that were performed with V = 0.1% and T = 10, (right). We can visually notice the structural

difference between them; the one without *nobility victims* (left) is more edge-dense, and contains a few highly connected nodes (biggest nodes) which is characteristic of the *sf* networks, whereas the one with *nobility victims* (right) present less edges and with the most-connected being almost indistinguishable due to its small size which is relative to the biggest node of both graphs.

In order to confirm whether the graphs in figure 3 represent *sf* networks we used the Python powerlaw package (Alstott et al., 2014). This package is able to statistically test if a distribution follows a power law. For the left graph, we can establish a statistical difference ($p = 0.028$) against the assumption of an exponential (null hypothesis), therefore it is very likely that it is a *sf* network, and for the right graph, we are not able to find a difference ($p = 0.958$).

V	0%	0.1%				1%				10%			
b /F	0	80	40	20	10	80	40	20	10	80	40	20	10
1	85	49	21	7	7	12	16	13	8	18	1	3	88
1.3	85	49	33	7	3	9	13	5	0	5	16	7	90
1.6	80	51	33	14	1	1	6	27	50	63	7	37	91
1.9	94	87	78	71	80	75	75	73	15	27	61	85	97
2.2	88	84	86	85	80	77	76	44	64	66	89	89	93
2.5	85	87	87	85	78	79	85	87	87	86	90	95	89
2.8	96	96	94	95	89	90	86	91	95	87	93	92	82
3.1	96	99	98	96	91	93	93	89	94	82	91	85	77

Table 2: Total of sf networks produced by each scenario. The bolded cells represent parameters of each scenario including cooperator and defector founding populations when causalities (*C*) are set to 2.5%; the first column (starting at the 3rd row) shows the temptation values (*b*), first row the *nobility victims* (*V*) and second row its *frequency* (*F*), i.e. the number of iterations after which the network is pruned. Each of the non-bolded cells presents the number of networks (out of 100) that prove to be *sf* ($p < 0.05$). The degradation is applied according to the number of *sf* networks produced. The cells with towards fairer tones shows the scenarios in which less networks proved to be *sf* networks, whereas the red tones show the ones in which more networks proved to be *sf*.

In table 2 we present the amount of networks that passed this statistical test ($p < 0.05$) for each scenario in order to show that the networks presented in figure 3 are not isolated cases. In the table, we merged the results for cooperator and defector founded networks since they were similar between them. As suggested by Miller and Knowles, we confirm that their fluctuation model using evolutionary preferential attachment without *nobility victims* (second row) generally produces *sf* networks (above 80% for all temptation values). With a few exceptions, the majority of networks were unable to pass the power law test when *raids* were present and the temptation was below 1.9; *sf* networks are frequently found again for temptation $b >= 1.9$. For $b > 1.9$, the structural change could be associated to a decline of cooperation, however, for $b = 1.9$, we still have multiple cases in which cooperation still holds (for V=0.1% and V=1% / F ϵ (40, 80)) and yet the structure fits that of a *sf* network. In terms of the network size, we found that when *nobility victims (V)* was set at 10%, the average size of the final networks was always below 905 nodes. This suggests that the growth phase was not fast enough to recover the

network, but also that the *raids* completely isolated many nodes that are also removed in Step 5; in this sense, we also observed that in these cases there were generally multiple components.

Discussion

We showed that the fluctuation model presented by Miller and Knowles improves the cooperation robustness against removal of the most connected nodes (*attack*-like mechanism), in this case selected by tournament (*raids*). This kind of node removal directly targets the heterogeneity of *sf* networks, which has been argued to dominate the fate of cooperation. We numerically showed that this is not necessarily the case, and that cooperation can persist under moderate levels of *raids* if there is a mechanism that allows for the network to readjust its ties. Surprisingly, *sf* networks structures reappear again when cooperation starts declining. The main reason for this seems to be that most of the nodes have no reward (i.e. fitness) in highly-defector-composed networks, therefore some minimal reward (due to random chance) would become very advantageous to attract new nodes (see equation 3). Some of the new nodes will be cooperators (half of them aproximately) that will keep the initial advantage propagating to next generations. Conversely, when cooperation is very high, the rewards are better distributed among the nodes, and so are the possibilities of getting new connections.

Methodologically speaking, we presented a parametrized attack mechanism (*raids*) that can be set at different rates and although it does not necessarily remove the top most connected nodes, these nodes are the most likely to be removed. Given the sensibility of *sf* networks to *attack*, this is a more cautious approach to study resilience of cooperation under removal of important nodes. In this sense, *battles* has the advantage over *error* that intentionally avoids the removal of the fittest nodes (*raids*). That said, further research should explore the presented model under traditional forms of *attack* (without the tournament) or even more sophisticated forms of it (Morone and Makse, 2015). Similarly, it is also important to evaluate smaller sizes (S) of *battles* tournament, including S = 1 (equivalent to random removal without the tournament, i.e. *error*) because it is a more realistic representation of mortality in societies. The model should be extended so that the agents recognize specific individuals (e.g. by using a history of interactions with each neighbor), leading to the use of a particular strategy towards each neighbor instead of reacting uniformly depending on the fitness of a randomly chosen neighbor (Step 2).

Regarding the Maya warfare, we were able to replicate scenarios in which cooperation persists for indefinite time in spite of nobility-targeting *raids*, which explains why the Maya political collapse of the AC 800 isn't directly associated with these kind of attacks. This collapse could be explained if the *raids* would have increased leading up that time, which is consistent with evidence in the Petexbatun area (Webster, 2000), though in this particular region the large increment of *battles* could have played a role as well. For other areas where we lack evidence for elevated warfare, our model favors the hypothesis that additional factors could have entered into play at the end of the Maya Classic period that increased the temptation to defect (*b*), e.g. environmental or economic crisis, or land degradation.

The model also allows us to venture the hypothesis that the nobility-targeting *raids* might have contributed to the emergence of a less hierarchical organization among the elites during the Maya Classic, thus supporting the relation between increased warfare and a more decentralized political hierarchy pointed out in the literature. We appeal to archeologists to verify if our results and new hypotheses are consistent with and helpful to explain the events of the Maya Classic.

Acknowledgments

References

Adams, R. E. W., and Smith, W. D. (1977). Apocalyptic visions: the Maya collapse and mediaeval Europe. *Archaeology*, *30*(5), 292–301.

Albert, R., Jeong, H., and Barabasi, A.-L. (2000). Error and attack tolerance of complex networks. *Nature*, *406*(6794), 378–382.

Alstott, J., Bullmore, E., and Plenz, D. (2014). powerlaw: a Python package for analysis of heavy-tailed distributions. *PLoS ONE*, *9*(1), e85777.

Aoyama, K. (2005). Classic Maya warfare and weapons: spear, dart, and arrow points of Aguateca and Copan. *Ancient Mesoamerica*, *16*(02), 291–304.

Axelrod, R. (1984). *The Evolution of Cooperation*. New York: Basic Books.

Barabási, A.-L., and Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509–512.

Brughmans, T. (2012). Thinking through networks: a review of formal network methods in archaeology. *Journal of Archaeological Method and Theory*, *20*(4), 623–662.

Callaway, D. S., Newman, M. E. J., Strogatz, S. H., and Watts, D. J. (2000). Network robustness and fragility: percolation on random graphs. *Physical Review Letters*, *85*(25), 5468–5471.

Cohen, R., Erez, K., ben-Avraham, D., and Havlin, S. (2000). Resilience of the internet to random breakdowns. *Physical Review Letters*, *85*(21), 4626–4628.

Cohen, R., Erez, K., ben-Avraham, D., and Havlin, S. (2001). Breakdown of the internet under intentional attack. *Physical Review Letters*, *86*(16), 3682–3685.

Freidel, D. (1986). Maya warfare: an example of peer-polity interaction. In C. Renfrew and J. F. Cherry, editors, *Peer Polity Interaction and Socio-political Change*, pages 93–108. London: Cambridge University Press.

Hassig, R. (1992). *War and Society in Ancient Mesoamerica*. Berkeley: University of California Press.

Ichinose, G., Tenguishi, Y., and Tanizawa, T. (2013). Robustness of cooperation on scale-free networks under continuous topological change. *Physical Review E*, *88*(5).

Jackson, S. E. (2013). *Politics of the Maya Court: Hierarchy and Change in the Late Classic Period*. Norman: University of Oklahoma Press.

Martin, S., and Grube, N. (2008). *Chronicle of the Maya Kings and Queens: deciphering the dynasties of the ancient Maya*. London: Thames & Hudson.

Miller, S., and Knowles, J. (2015a). A minimal model for the emergence of cooperation in randomly growing networks. In *European Conference on Artificial Life 2015*, pages 114–121. The MIT Press.

Miller, S., and Knowles, J. (2015b). Population fluctuation promotes cooperation in networks. *Scientific Reports*, *5*, 11054.

Morone, F., and Makse, H. A. (2015). Influence maximization in complex networks through optimal percolation. *Nature*, *524*(7563), 65–68.

Normak, J. (2007). Lethal encounters: warfare and virtual ideologies in the Maya area. In Per Cornell and Fredrik Fahlander, editors, *Encounters, Materialities, Confrontations : Archaeologies of Social Space and Interaction*, pages 165–197. Newcastle: Cambridge Scholars Press.

Nowak, M. A., and May, R. M. (1992). Evolutionary games and spatial chaos. *Nature*, *359*(6398), 826–829.

O'Mansky, M., and Demarest, A. A. (2007). Status rivalry and warfare in the development and collapse of Classic Maya civilization. In R. J. Chacon and R. G. Mendoza, editors, *Latin American Indigenous Warfare and Ritual Violence*, pages 11–33. USA: University of Arizona Press.

Perc, M. (2009). Evolution of cooperation on scale-free networks subject to error and attack. *New Journal of Physics*, *11*(3), 033027.

Poncela, J., Gómez-Gardeñes, J., Floría, L. M., Sánchez, A., and Moreno, Y. (2008). Complex cooperative networks from evolutionary preferential attachment. *PLoS ONE*, *3*(6), e2449.

Santos, F. C., and Pacheco, J. M. (2005). Scale-free networks provide a unifying framework for the emergence of cooperation. *Physical Review Letters*, *95*(9), 098104.

Santos, F. C., Pinheiro, F. L., Lenaerts, T., and Pacheco, J. M. (2012). The role of diversity in the evolution of cooperation. *Journal of Theoretical Biology*, *299*, 88–96.

Stuart, D. (1993). Historical inscriptions and the Maya collapse. In J. A. Sabloff and J. S. Henderson, editors, *Lowland Maya Civilization in the Eighth Century A.D.*, pages 321–354. Dumbarton Oaks, Washington, D.C.

United Nations, D. of E. and S. A., Population division. (2013). *World Population Prospects: The 2012 Revision*. New York: United Nations.

Van Tuerenhout, D. (2001). Maya warfare: sources and interpretations. *Civilisations*, *50*(1/2), 129–152.

Webster, D. (2000). The not so peaceful civilization: a review of Maya war. *Journal of World Prehistory*, *14*(1), 65–119.

Webster, D. (2002). *The Fall of the Ancient Maya: Solving the Mystery of the Maya Collapse*. London ; New York: Thames & Hudson.

Webster, D., Freter, A., and Gonlin, N. (2000). *Copán: the Rise and Fall of an Ancient Maya Kingdom*. Fort Worth: Harcourt College Publishers.

Webster, D., Sanders, W. T., and van Rossum, P. (1992). A simulation of copan population history and its implications. *Ancient Mesoamerica*, *3*(01), 185–197.

Equality seekers or moderate monopolists: Social structure affects the evolution of distributive norms

Kazuaki Kojima[1], Reiji Suzuki and Takaya Arita

Graduate School of Information Science, Nagoya University, Japan
[1]k-kojima@alife.cs.is.nagoya-u.ac.jp

Introduction People in some societies tend to put a greater value on equality in distribution of resources even if they have to pay expensive court costs to achieve it, while people in some other societies tend to aim at a maximal share (the whole) but withdraw readily if any conflict occurs. Nash demand game (NDG) is a one-shot two-player game and has been widely used for modeling such bargaining situations in computational and game theoretic approaches. Each player simultaneously demands a portion of some good. If the total amount demanded by the players is less or equal than available good, each player obtains the claimed request. Otherwise, neither player gets anything. Whereas the studies using NDG can account for why people favor the equal distribution (Skyrms (1996)), it is too simple to deal with various distributive norms. We use the demand-intensity game (DIG) which adds a psychological factor to NDG while maintaining such simplicity that it can be analyzed by the concepts and tools of the game theory (e.g., Kojima and Arita (2012)).

The goal of this study is to clarify the origin and evolutionary dynamics of distributive norms using DIG. Previous studies have shown that population structures tend to promote cooperative behavior by means of cooperative clustering and assortative interactions. We perform the evolutionary simulation focusing on the effect of the population structures on the evolution of distributive norms. We show a surprising result that network structures significantly change the evolutionary scenario. A population distributed over a regular network tends to evolve a strong equality norm. However, as the random links increase in the network, the more we see the scenario in which monopolists occupy the population who ask for the whole but with a moderate or timid intensity. We also find that network structures with some intermediate randomness create an interesting scenario in which several norms emerge in a cyclic manner.

Model DIG is a one-shot game between two players. Each player has values of d and i ($0 \leq d, i \leq 1$) as a strategy $S(d,i)$: $S(d_0, i_0)$ for player 0 and $S(d_1, i_1)$ for player 1. d indicates how much portion of the resource she wants and i indicates the intensity of the demand as shown in Fig. 1.
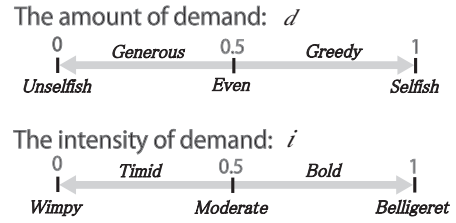

Figure 1: Representation of strategies.

For example, a strategy with $d = 1$ is described as "selfish" while a strategy with $i < 0.5$ is described as "timid."

Player 0's payoff is defined as the following equations (1)-(3). If the aggregated demand between both do not exceed the total resource 1, each player obtains her demand d as a payoff without a conflict. Otherwise, each player obtains a payoff reduced by the conflict cost defined as the mean intensity of them, from the tentative payoff tp considered as follows. The self demand d_0 is separated into two parts: the directly obtained part $(1-d_1)$ and the resting overlapped part (d_0+d_1-1). The latter will be divided at the ratio based on the difference between player's intensities ($1 + i_0 - i_1 : 1 + i_1 - i_0$).

$$payoff = \begin{cases} d_0 & (if\ d_0 + d_1 \leq 1), \\ tp \cdot (1 - cost) & (otherwise), \end{cases} \quad (1)$$

$$tp = (1 - d_1) + (d_0 + d_1 - 1)\frac{1 + i_0 - i_1}{2}, \quad (2)$$

$$cost = \frac{i_0 + i_1}{2}. \quad (3)$$

There are two typical strategies: $d = 0.5$ and $d = 1$. Although debatable, we simply associate the former with "egalitarianism" and the latter with "libertarianism." The ideal society in the sense of equality and efficiency (anyone receives 0.5 in every game) can be achieved by not only egalitarianism norm $S(0.5, *)$ but also "wimpy" libertarianism norm $S(1, 0)$.

Social networks in our model are represented as Moore neighborhood structure on a toroidal square lattice consisting of 100 x 100 nodes each of which has a player. Net-
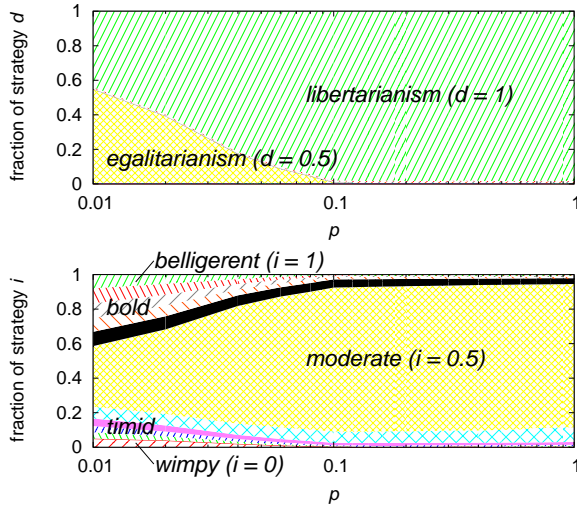
Figure 2: Distribution of the evolving strategies from 1000th to 1500th generations as a function of the rewiring probability p averaged over 100 trials ($w = 1$). Each area corresponds to each strategy value, sorted in order of increasing value.

works are constructed by rewiring each link with a probability p. In each generation of evolution, each player plays DIG with directly connected neighbors. The fitness for player j is defined as $exp[w\pi_j]$ in which w represents the intensity of selection and π_j represents the average payoff. Each player adopts a strategy selected fitness-proportionally from own and all neighborhoods' strategies as a next strategy and then changes d and i to a random value with a low probability 0.05, respectively.

Effects of social structure We conducted evolutionary simulations in which d and i have discrete values from 0 to 1 with a 0.1 resolution. Figure 2 shows the distribution of the evolving strategies from 1000th to 1500th generations ($w = 1$). Although it is not clear from this figure, basically just one strategy always occupied the population as a norm except for some region of p.

We see a clear tendency that as the random links increase egalitarianism disappears and instead libertarianism grows and then every time occupies the population. We also found that egalitarianism was coupled with various intensities, $S(0.5, *)$ and libertarianism was coupled mainly with the moderate intensity, $S(1, 0.5)$.

Furthermore, we found that in a few trials network structures with some intermediate randomness ($p \approx 0.06$) create an interesting scenario in which several norms (including timid or moderate libertarianism and bold or moderate egalitarianism) emerge in a cyclic manner as shown in Fig. 3. It is similar to cyclic dynamics emerging from voluntary interactions in the public goods game or the prisoner's dilemma
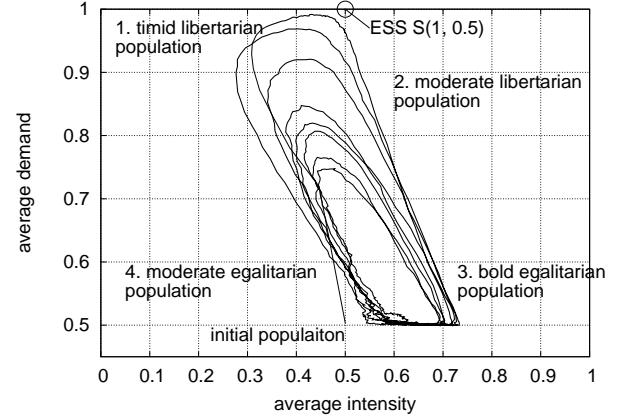

Figure 3: A typical evolutionary trajectory of strategies averaged over each population. Starting from $(0.5, 0.5)$ as mean values of a random population, it drew clockwise circles composed of (1) a growth of the number of libertarians, (2) an increment of the intensities, (3) an increase of the number of bold egalitarians and (4) a decrement of the intensities.

(Hauert et al. (2002), Suzuki et al. (2008)).

In addition, when using networks with strong spatial locality (small p), strong selection (large w) favored the coexistence of egalitarianism and timid libertarianism. Timid or wimpy libertarians, by obtaining a high payoff against any strategy, could survive under strong selection in spite of the advantageous network structure for egalitarianism.

Conclusion We demonstrated that the population structure could strongly affect the evolution of distributive norms by performing the evolutionary simulations using an extended version of the Nash demand game. Specifically, we showed that spatial locality favors equality seekers while randomness in the networks favors "moderate" or "timid" monopolists. This result might offer significant implications to us living in a world where an increasing number of people are connected to each other through social networking although our tendency to connect with similar others should be taken into consideration.

References

Hauert, C., De Monte, S., Hofbauer, J., and Sigmund, K. (2002). Volunteering as red queen mechanism for cooperation in public goods games. *Science*, 296:1129–1132.

Kojima, K. and Arita, T. (2012). How do equity norms evolve? - an evolutionary game theory approach to distributive justice. *Artificial Life and Robotics*, 17(2):287–292.

Skyrms, B. (1996). *Evolution of the Social Contract*. Cambridge University Press, Cambridge.

Suzuki, R., Kato, M., and Arita, T. (2008). Cyclic coevolution of cooperative behaviors and network structures. *Physical Review E*, 77:021911.

Language and Cultural Evolution

Evolving Artificial Language through Evolutionary Reinforcement Learning

Xun Li[1] and Risto Miikkulainen[2]

[1,2]University of Texas at Austin
xun@cs.utexas.edu

Abstract

Computational simulation of language evolution provides valuable insights into the origin of language. Simulating the evolution of language among agents in an artificial world also presents an interesting challenge in evolutionary computation and machine learning. In this paper, a "jungle world" is constructed where agents must accomplish different tasks such as hunting and mating by evolving their own language to coordinate their actions. In addition, all agents must acquire the language during their lifetime through interaction with other agents. This paper proposes the algorithm of Evolutionary Reinforcement Learning with Potentiation and Memory (ERL-POM) as a computational approach for achieving this goal. Experimental results show that ERL-POM is effective in situated simulation of language evolution, demonstrating that languages can be evolved in the artificial environment when communication is necessary for some or all of the tasks the agents perform.

Introduction

Highly efficient and low-cost computer systems have made computational simulation possible at an unprecedented scale in recent decades. In the specific field of language evolution, computational simulation provides a complementary methodology that can help researchers develop detailed hypotheses on language origins and evolution and test these hypotheses in the virtual laboratory of simulation (Cangelosi and Parisi, 2002). Furthermore, from a technical perspective, an understanding of the fundamental principles in language evolution may lead to innovative machine learning algorithms and communication methods that are applicable to interactive software agents and multi-agent systems (Wagner, Reggia, Uriagereka, and Wilkinson, 2003).

Language is a powerful tool that helps humans coordinate actions to accomplish various tasks. It is also a skill that is acquired through lifetime learning. The purpose of this paper is therefore to establish a simulation framework, i.e. an artificial world and a computational method that captures these important features into a simulation of language evolution. Such a framework should then make it possible to gain new insights into evolution of natural and artificial language.

The first part of the simulation framework: "the jungle world", is an artificial environment in which agents attempt to hunt and mate through coordinated actions. Initially, the agent population have neither any knowledge on the rules of the world nor any existing code of communication. Through

generations of evolution, the agent population must develop their own language and learn to use that language to coordinate their hunting and mating efforts. Additionally, for each agent, the language and the behavioral policy in the artificial world must be acquired through interaction with other agents and the environment during lifetime. Thus, the goal of evolution is to (1) evolve a language, (2) evolve it in service of coordinated behavior, and (3) evolve the ability for individuals to acquire it during their lifetime.

To allow efficient simulation of language evolution in the jungle world, a biologically-inspired algorithm, Evolutionary Reinforcement Learning with Potentiation and Memory (ERL-POM) is proposed. This approach utilizes a genetic algorithm to configure reinforcement learners units. State-action memory and potentiation are introduced to balance exploitation with exploration and improve interactive learning ability.

Using the proposed algorithm, language evolution and acquisition is simulated under a variety of settings of the jungle world. These settings include the scenario where communication is necessary for both tasks, one of the tasks, or neither of the tasks. The paper also presents and analyzes samples of the artificial languages evolved in different settings. Experimental results and analysis show that ERL-POM is effective in simulating language evolution and acquisition, demonstrating that languages can be evolved and acquired in the artificial world if communication is necessary for one or both of the tasks.

The remaining sections of the paper are organized as follows. The next section gives a brief review on prior work in computational simulation of language evolution. The third section introduces rules and settings of the jungle world, and the fourth section provides details on the algorithm. The fifth section presents and analyzes experimental results, and the sixth section points out potential directions for future work.

Prior Work

In a typical simulation of language evolution, a multi-agent system is created to simulate an entire population of agents. Each agent acquires a shared communication system either by using machine learning methods and/or through simulated evolutionary process.

Simulations of language evolution can be divided into situated and non-situated simulations. In a non-situated simulation, an agent's actions consist solely of sending and receiving signals. Such non-embodied agents perceive objects

and events, but do not change the state of the environment. Usually, the agents aim at encoding an arbitrary meaning as a signal and send it to another agent, who decodes the signal back to a meaning. In such simulations, neural networks (Batali, 1998; Kvasnicka and Pospichal, 1999; Smith 2002), lookup tables (Kaplan 2000; Smith 2001), associative memories (De Boer and Vogt, 1999; Steels and Oudeyer, 2000) and finite state machines (MacLennan and Burghardt, 1993; Brighton 2002) are the most commonly used models to represent the behaviors. While they have been employed to demonstrate many interesting properties of communication systems, non-situated simulations are unrealistic in that they do not associate external tasks with communication actions. In contrast, the evolution of language in nature is strongly linked to the need to perform various tasks in which communication helps.

To address this problem, situated simulations can be built. In such a simulation, agents are embodied in an artificial world. Their goal is usually to accomplish tasks that require cooperation or competition among multiple agents. Thus, language serves as a necessary or beneficial tool to achieve higher performance in multi-agent tasks. Situated simulations can be used to test specific assumptions on the role of certain behaviors or environmental factors in the evolution of language (Quinn 2001; Mirolli and Parisi, 2010; De Greeff and Nolfi, 2010; Mitri, 2010; Rawal, Boughman, and Miikkulainen, 2014).

However, prior work on situated simulations is limited in two ways. First, most of them focus on a single task. More specifically, the rewards of actions, be it communicative or non-communicative, do not change throughout the lifetime of agents in all generations. In contrast, in nature, language is used for numerous tasks, and the rewards of actions depend on multiple factors. Second, the language is usually encoded genetically and passed on to the next generation through genotypes. In contrast, language in nature is acquired during lifetime learning and passed on to the next generation through interaction among individuals in the environment. While some of the existing work addresses one of the above problems, to our best knowledge, no prior work on situated simulations evolves artificial languages that are both applicable to different tasks and acquired through lifetime learning. Therefore, the purpose of this work is to introduce a simulation framework that achieves both goals. Such a framework makes the simulation more realistic and should be helpful in discovering deeper insights into the origin and evolution of language.

Simulation Environment

This section introduces the rules and settings of the jungle world – the artificial environment used in the experiments. The goal is to establish a paradigm of situated simulation environment where languages evolved are used in different tasks at different stages of an agent's life, and knowledge must be acquired through lifetime learning. While only a few variations of the jungle world are used in the experiments, the simulation environment can be modified to serve many other experimental goals. In addition, the jungle world does not impose any requirement for the artificial controller of the agents except for an interface that specifies inputs and outputs.

Hence, it can be viewed as a general test environment for evaluating performances of genetic based machine learning algorithms.

Life in the Jungle World

This subsection presents basic concepts and rules in the virtual world. The focus is on actions and rewards during a single generation. Concepts and rules are organized into entries with short definitions and descriptions.

Step and Trial. Time is discretized into steps. At each step, agents receive inputs from the artificial environment including messages from their partner, and take actions accordingly. A trial is a 10-step experiment with two agents. It terminates early if any of the participants receive a positive or negative reward.

Jungle. The jungle is the place where agents hunt for prey and feed themselves. However, if an imprudent agent enters the jungle without its partner at any step, it will be hurt and receive a negative reward.

Agent. An agent has two integer states: fitness and position. Fitness ranges from 10 to 200, with 10 the initial value for new-born agents. Fitness increases by 10 after a successful hunt (defined later) and decreases by 1 after each trial. Position ranges from 0 to 5, indicating the distance between an agent and the jungle.

An agent senses its proximity to the jungle and becomes alert if its position is 1. It becomes ready for mating if its fitness is greater than 100.

At each step, an agent may decide to take the following actions: (1) move towards the jungle, (2) attempt to mate, and (3) send a two-bit message.

Thus, in the typical setting of the jungle world, an agent's brain, i.e. the controller, receives a four-bit input at each time step: position alert, mating readiness, message bit 1 and 2. Based on the input, the controller makes a four-bit decision, indicating whether the agent decides to move towards the jungle, to mate with its partner, or to set message bits to one.

Hunting. An agent succeeds in hunting if it enters the jungle with its trial partner at the same time step. A successful hunt gives a positive reward and increases fitness by 10 for both agents.

Mating. If a pair of agents decides to mate at the same step, and both of them are ready, they succeed in mating and receive a positive reward equal to 1/10 of their partner's fitness. Thus, successful mating always claims more rewards than hunting, especially so when fitness is high.

If an agent decides to mate when its fitness does not exceed 100, it receives a negative reward for cheating its partner. If an agent decides to mate while its partner is not ready, it is embarrassed and receives a negative reward, too.

Idling. If a pair of agents claim no reward at the end of a trial, they receive a negative reward for wasting time.

Population. The population in all generations contains 50 agents with 25 seniors and 25 juniors (except for the first generation in which no seniors exist). A senior is an agent who survived the selection process after the previous generation. A

junior is a newborn agent whose parents are a pair of senior agents.

Generation. A generation contains two phases: parenting and socializing (except for the first generation). In the parenting phase, junior agents pair with each of their parents for 100 trials. Since every junior has two parents, a total of 5000 trials are conducted in the parenting phase. In the socializing phase, every pair of agents in the current generation participates in 100 trials with random ordering of partners. Thus, a total of 122,500 trials are conducted in the socializing phase. Rewards accumulated from the socializing trials are used as the performance measure for all agents.

Selection. After each generation, agents are ranked based on their performance. Top 25 eligible agents survive the selection and become the senior agents in the next generation. An agent is eligible as long as its life spans fewer than four generations.

Reproduction. Before the next generation starts, selected agents must participate in the reproduction process to produce junior agents for the next generation. While the artificial world does not impose any requirement on the mechanisms in which genotypes of selected agents are used to generate new agents, the following approach is employed in the experiments.

Each of the 25 selected agents is paired with a randomly chosen partner. Junior agents are then constructed from them through mutation and crossover. Thus, every selected agent has at least one child, but may have two or more children due to random pairing. Details on genotypes, mutation, and crossover is presented in the "Method" section.

Language Acquisition Requirement (LAR). Junior agents in all generations as well as agents in the first generation must NOT have any knowledge on the rules of the world and the language that is used in it. All agents must acquire such knowledge by participating in trials. The genotype of an agent defines how it learns, rather than representing such knowledge directly.

Implications to Language Evolution

As defined in the previous subsection, the jungle world is an artificial environment for situated simulations of language evolution. This subsection gives a brief discussion on two important features of this artificial environment and points out their implications.

Multitasking. Jungle world requires artificial agents to evolve languages that are applicable to different tasks from three perspectives.

First and most obviously, there are two tasks (actions), namely hunting, i.e. moving into the jungle, and mating, i.e. attempting to mate with a partner, that need to be coordinated through communication. Each action receives negative or positive rewards under different environmental state.

Second, if the long-term reward of these actions remains constant in a changing environment, the so-called "different tasks" are actually a single task interpreted subjectively with the semantics of multiple tasks. After all, the core challenge of multitasking is that possible actions pursuing different tasks must be properly prioritized and coordinated in order to achieve higher rewards in the long run.

Indeed, agents in the jungle world are required to coordinate and prioritize their tasks throughout their lifetime via the language they evolve. For instance, agents in the socializing phase usually have a fitness score higher than 100, which gives them legitimate choices of hunting and mating. Successful mating always yields a higher reward than hunting. However, fitness decreases at the end of each trial. Therefore the agents have to hunt regularly to maintain a high fitness level. Since both actions are always available to the agents, they must learn to prioritize the actions and coordinate with partners in pursuing each task. Note that the messages sent and received are part of the environmental state, and in some variance of the jungle world, messaging is the only way in which agents can communicate state or convey intention. In this sense, the jungle world presents a true multi-task challenge.

Third, due to LAR and the fitness requirement for mating, agents in the jungle world must adapt their strategy at different stages of their life. As in nature, junior agents must first learn to interpret the environment and hunt successfully through communicating with their parents in the parenting phase. Before their fitness score can be maintained at a high level (i.e. greater than 100), mating is not a viable choice. However, as agents proceed into the socializing phase, they must adjust their strategy and learn to balance mating and hunting in order to maximize their cumulative rewards.

Language Acquisition. LAR ensures that agents have to learn to survive the jungle world during their lifetime rather than relying solely on the information encoded in their genes. LAR thus makes simulation of language evolution more realistic: in nature, language and knowledge is largely acquired through lifetime learning rather than genetically encoded. While some theories suggest that the human genotype encodes the universal grammar (Chomsky and DiNozzi 1972), it is commonly accepted that any particular language should be learned.

From a technical perspective, LAR presents interesting challenges. Many powerful methods such as neural networks are usually not directly applicable to learning in real time. Meanwhile, traditional reinforcement learning algorithms such as Q learning require accurate and full observation of environmental state. In contrast, in typical settings of the jungle world, states are only partially observable, i.e. agents cannot observe the fitness or the position of their trial partners. While communication can be leveraged to compensate partial observation, at the early stage of evolution, semantics of the messages are rather unreliable and frequently changing. Even after a language is established among the senior population, messages from junior agents in the parenting phase can mislead or confuse their parents and in the worst case, reverse the progress of language evolution in previous generations.

While reflecting important features of language in nature, both multitasking and LAR present interesting challenges to the method used in such simulations.

Method

This section presents details on Evolutionary Reinforcement Learning with Potentiation and Memory (ERL-POM) – the method adopted to allow efficient and effective simulation in

the jungle world. The reinforcement learner serves as the brain of the agents. In each generation, every individual in the population has its own reinforcement learner. Through lifetime interaction with trial partners, these controllers adjust their policy gradually to achieve higher cumulative rewards.

Controller Structure

This subsection introduces the structure of the controller.

Inputs and Outputs. The controller assumes that both the inputs (i.e. environmental states) and the outputs, (i.e. action decisions) are binary, or can be converted to binary.

Learner Unit. Similar to neurons in a neural network, learner units are basic functional units of a controller. They take binary inputs (multiple bits) and give a single bit output. Each unit consists a policy map and a memory. The policy map is implemented as a Hashmap whose keys are the input patterns and values *activation parameters* (AP).

Memory. Memory is a queue of input-output pairs with a certain size. Inspired by the short term memory in nature, old items in the queue are replaced by new items once the number of items reaches memory size. Memory provides a record on decisions given input patterns and helps the learner unit adjust APs to maximize long term reward.

Expansion. Since both inputs and outputs are binary, learner units can be connected similarly to neurons in a neural network to form various complex structures. They also adjust their weights based on memory and rewards while interacting with the environment. Thus, a controller is essentially a neural network designed for real-time learning.

However, it is worth pointing out that learner units are more powerful and complicated than neurons in typical neural networks. Therefore, it usually takes fewer learner units than neurons to solve a problem. As a matter of fact, given the typical setting of the jungle world, merely four learner units are needed to achieve high average cumulative rewards, which is similar to a single layer neural network if all learner units are replaced by neurons.
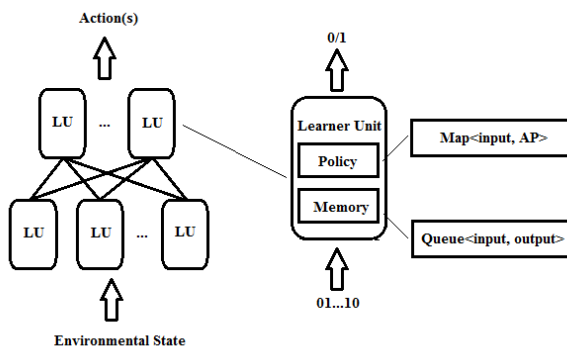

Figure 1: Controller Structure. A controller may contain one or multiple layers of learner units, connected similarly to neurons in a neural network. Each learner unit consists a policy and a memory. The policy is a Hashmap whose keys are input patterns and values activation parameters, and the memory is a queue of input-output pairs.

Real Time Learning and Evolution

Initially, all policy maps are empty. As an agent explores the artificial world, learner units receive inputs and insert a new entry to their policy map for each previously unseen input. The AP in each new entry is set to 0. The probability of activation P_A given the activation parameter AP of an input is computed according to formula 1.

$$P_A = \sigma(AP) = \frac{1}{1+e^{-AP}} \quad (1)$$

Here, σ is the sigmoid function. Note that for each new entry, the probability of activation is 0.5, i.e. the learner units performs random exploration.

Learning with Memory. Whenever a learner unit makes a decision given an input, the event (input-output pair) is pushed into the memory queue. After the queue is full, oldest records are replaced by new ones.

When the agent collects a reward r, each learner unit looks into its memory. For each event (i.e. input-output pair) in the queue, the learner unit updates the AP of the corresponding input in the policy map according to formula 2.

$$AP' = \alpha \cdot AP + r \cdot d^n(2x - 1) \quad (2)$$

Here, AP and AP' are the activation parameter before and after the update, respectively, α is the *decay rate* ($0 \leq \alpha \leq 1$), r is the normalized reward ($-1 \leq r \leq 1$), d is *discount factor* ($0 \leq d \leq 1$), n is event index in the memory queue, with 0 representing the most recent, and x is the binary output.

Intuitively, learner units increase the AP mapped to an input pattern (thus the activation probability given that input) if (1) a positive reward is received, and the unit outputs 1, or (2) a negative reward is received, and the unit outputs 0. The decay rate balances the influence of knowledge from the past with the most recent experience. When $\alpha = 0$, only the latest experience is taken into consideration. The discount factor reflects the contribution of decisions in the past. If $d = 0$, only the last decision is assumed to be the cause of the reward, and if $d = 1$, the reward is assumed to be equally attributable to all events in the memory.

Potentiation. Because junior partner behave randomly, it is possible that senior partners are confused during the parenting phase. Therefore, potentiation is introduced to retain long-term memory, i.e. well-tested knowledge and rules learned in the past.

As in nature, if the brain is confident enough on a decision for a certain input, that decision is fixed. Specifically, if AP of an input satisfies the condition in formula 3, the learner unit fixes the decision on that input to 1 if AP is positive, and to 0 if it is negative.

$$\frac{|\sigma(AP) - 0.5|}{0.5} \geq PT \quad (3)$$

Here PT ($0 \leq PT \leq 1$) is *potentiation threshold*. Intuitively, PT specifies how confident the controller must be in order to fix its decision. If PT = 0, the controller will fix its decision after learning from a single event, and if PT = 1, the controller will never fix its decision.

Evolution. As introduced in the previous subsections, each learner unit is defined by the following parameters: (1) the potentiation threshold, (2) the discount factor, (3) the decay

rate, and (4) the memory size. Thus, a controller with m learner units and a fixed topology has a genome with $4m$ numbers, which include m positive integers with a maximum (i.e. the maximum size of the memory), and $3m$ real numbers [0..1]. Since all controllers have numeric genomes with uniformly defined structure, mutation and crossover can be directly applied to producing controllers for a new generation.

In the above mechanism, the role of evolution is to explore the learner space and optimize parameters for units in the controller. In other words, evolution aims to improve learning ability rather than encode policies. Junior agents in a generation are equipped with potentially better learning tools. Nevertheless, they have no specific knowledge of the rules of the world or the language among the senior agents. Thus, this method satisfies LAR as defined in the previous section.

Experimental Results

This section presents and discusses the experimental results of the situated simulation under four different settings of the jungle world. In the first setting, communication channels are disabled, and the environment is fully observable, i.e. agents can observe the fitness and position of their partners directly. The second settings allows full observation while enabling communication. In the third setting, communication channels are enabled, but agents cannot observe the position or fitness of their partner. The fourth setting is the same as the third, except that partner position is observable. The above four settings aim to address the following questions:

1. When communication channels are disabled and environmental states fully observable, what kind of behaviors emerge as a baseline?
2. If environmental states are fully observable, i.e. communication is enabled but unnecessary, will any language emerge?
3. When communication is necessary for all tasks, can the agents evolve a language to coordinate their actions?
4. If communication is necessary for some of the tasks but not for the others, will any language emerge?

In each of the experiments, any language that emerges is analyzed to understand what it is and how it helps agents to perform their tasks.

Experimental Setup

Table 1 shows the parameter settings used in all simulations. The normal distributions (ND) have a standard deviation of 0.1, with a mean of zero. Mutation rate is applied to each gene independently. A special rule (SR) is applied to the mutation of memory size: it increases or decreases by one with 0.5 probability for each.

After the seniors are selected from the previous generation, each senior has one chance to be paired with another senior randomly to produce a junior for the next generation. The genome of the first senior is used as the initial genome of the child. With a probability equal to crossover rate, each gene (i.e. number) has a chance to be replaced by the corresponding gene of the random spouse. In addition, it can mutate based on

Item	Value/Setting	
Population Size	50	
Senior/Junior	25/25	
Mutation Rate	0.1	
Mutation Rule	Gene	Initial Value (Method)
	MS	1 (SR, $1 \leq MS \leq 10$)
	PT	1.0 (ND, $0 \leq PT \leq 1$)
	α	1.0 (ND, $0 \leq \alpha \leq 1$)
	d	0.0 (ND, $0 \leq d \leq 1$)
Crossover Rate	0.1	

Table 1: Parameters and Mutation Rules

the mutation rate and rules in table 1. Crossover always takes place before mutation.

Note that in table 1, Initial Value refers to the values of parameters of learner units in the first generation. Memory size is set to one so that agents memorize only the last event. Potentiation threshold is 1.0, thus the agents never fix their decisions. Discount factor is 0.0, i.e. reward is assumed to be attributable to only the latest action. In other words, memory and potentiation do NOT exist for the first generation. It is up to evolution to decide if they are desirable. Such a setting serves two purposes: (1) it demonstrates the benefit of memory and potentiation through evolution, and (2) it avoids unnecessary structural and algorithmic complication. However, it has a downside of potentially delaying the emergence of language in the artificial world since such a simple starting point may be far from the best settings.

Experiment 1 (Baseline)

In this experiment, the communication channels are blocked and the environment is fully observable to the agents. Thus, agents cannot send messages to their partner, but they can observe the fitness and the position of their partner directly. The results in this group serve as a reference to evaluate the performance in other experiments. Figure 2 shows the results.
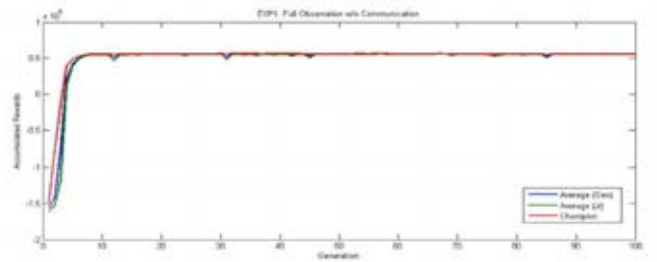

Figure 2: Experiment 1. In approximately eight generations, accumulated rewards become stabilized, and the performance does not differentiate much between juniors, seniors, and the champion. All results, including those of the other experiments are averaged from 20 runs.

Table 2 presents a sample policy of a champion in the 100th generation. According to the policy, the champion mates with its partner whenever they are both ready to mate (i.e. fitness $\geq$ 100). Until then, it moves towards the jungle as long as its position is greater than one. When getting close to the jungle, the champion waits for the partner if it is not in position, and

jumps into the jungle as soon as both of their positions equal to one.

F'	P'	F	P	**A**	**M**
0	0	0	0	1	0
0	0	0	1	0	0
0	0	1	0	1	0
0	0	1	1	0	0
0	1	0	0	1	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	1	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	1	0	0	1
1	0	1	1	0	1
1	1	0	0	1	0
1	1	0	1	1	0
1	1	1	0	1	1
1	1	1	1	1	1

Table 2: Champion Policy – Experiment 1. Regular letters are input states, and bold letters are actions. F represents fitness, i.e. whether the fitness score is greater than 100; P is position, i.e. whether position equals 1; F' and P' indicate partner's fitness and position, respectively; A is the action to approach the jungle; and M is the action to mate.

Lastly, memory size after ten generations averages 1.28, with a majority of learner units having no memory beyond the last decision. This result can be explained by the fact that in a fully observable environment, with a majority of senior agent policies like that in table 2, the challenge faced by a junior agent is largely a Markovian problem. On the other hand, the average potentiation threshold is 0.965 – the agents do fix their actions, but only when they are very confident in their decisions.

Experiment 2 (Unnecessary Communication)

Experiment 2 is different from the Experiment 1 in that agents are allowed to send and receive messages. Since the position and fitness of their partner are still observable, communication is possible but unnecessary in achieving any of the tasks.

Figure 3 presents the results of Experiment 2. Memory size after ten generations averages 1.42, with average potentiation thresholds as 0.971.
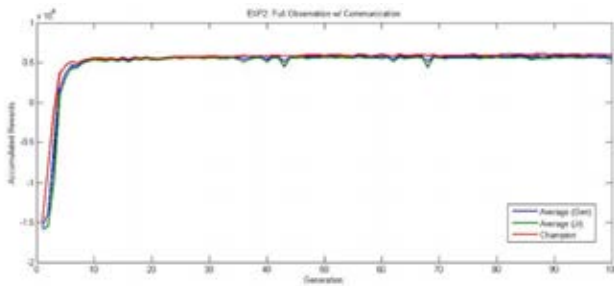

Figure 3: Experiment 2. Average cumulative rewards of each generation are nearly identical to those in Experiment 1.

The champion policies in Experiment 2 are characterized by the following three observations:

1. Given same observation of fitness and positions, actions are the same regardless of the message received.
2. Messaging policies vary from generation to generation, while having no influence on the stability of performance.
3. There is no clear correlation between messages and environmental states in most champion policies.

The results in Experiment 2 suggest that if communication is unnecessary for any of the tasks, messaging policy plays no role in agent performance. Agents evolve no language even when the communication channels are available.

Experiment 3 (Necessary Communication)

In this experiment, agents cannot observe the position and fitness of their trial partners. Therefore, the only means by which agents can coordinate their efforts in mating and hunting is through communication.
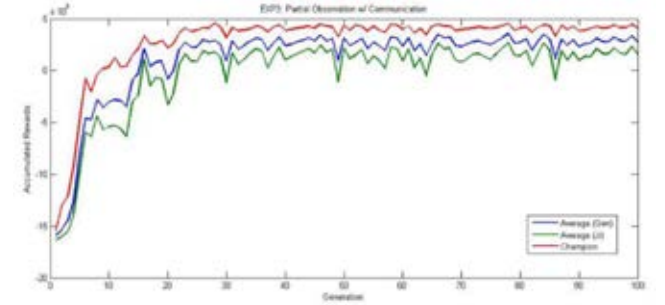

Figure 4: Experiment 3. Champions in the agent population achieve similar performance to that of Experiment 1 after approximately 25 generations. Champion performance (red) stabilizes afterwards. However, compared to Experiment 1, average cumulated rewards among entire generations (blue) and among the juniors (green) are lower and have bigger gaps in between.

The reasons for lower average performance is that multiple languages may occur simultaneously in one generation, causing confusion among the juniors in parenting phase, thus lower their performance in the socializing phase.

Table 3 presents a sample policy of a champion in the last generation. Champion policies after thirty generations encode the fitness and position accurately and consistently in 19 out of the 20 runs. However, in almost all generations, more than 20% of the seniors have a messaging policy that either fails to encode fitness or position states accurately, or differs from the champion policy.

Also, average memory size after 30 generations is 7.10, and the average discount factor is 0.722. The reason for the long memory is that past decisions can be used to complement partial observation and improve learning efficiency. For instance, if an agent keeps sending wrong messages while being in position for hunt, its trial partner can never know that the agent is ready, resulting in a punishment for idling to both

agents in the trial. While potentiation may keep the seniors from "second guessing" their correct policy, a long memory of the past can help the juniors learn faster in such cases.

F	P	R2	R1	**A**	**M**	**S2**	**S1**
0	0	0	0	1	0	1	1
0	0	0	1	1	0		
0	0	1	0	1	0		
0	0	1	1	1	0		
0	1	0	0	0	0	1	0
0	1	0	1	1	0		
0	1	1	0	1	0		
0	1	1	1	0	0		
1	0	0	0	1	1	0	0
1	0	0	1	1	1		
1	0	1	0	1	0		
1	0	1	1	1	0		
1	1	0	0	0	1	0	1
1	1	0	1	1	0		
1	1	1	0	1	0		
1	1	1	1	0	0		

Table 3: Champion Policy – Experiment 3. S1 and S2 represent the first and second message bit; R1 and R2 are the message bits received from the partner. All other letters have the same meaning as those in Table 2. The messages encode fitness and position values.

Although the average potentiation threshold is close to one (0.981), potentiation is crucial in the simulation because the parenting phase is sufficiently long to generate enough confusing interactions to reverse knowledge encoded in the seniors' controllers. In fact, if potentiation threshold is fixed to one (i.e. potentiation does not exist), language simply cannot be established among the agent population, rendering cumulative rewards consistently negative in all generations.

Experiment 4 (Partially Necessary Communication)

In this experiment, the position of a trial partner is observable while the fitness of the partner is hidden. Communication is possible between partners in a trial, however, it is necessary only for mating.
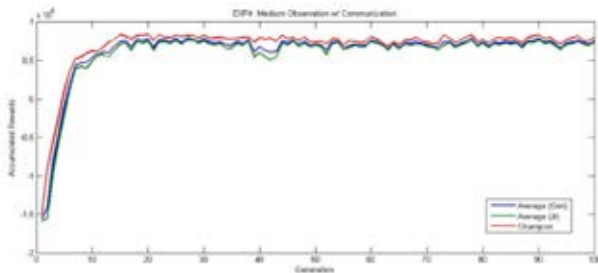

Figure 5: Experiment 4. Average performance in the first 10 generations is similar to that in Experiment 1. After 15 to 20 generations, rewards stabilize at a level approximately 50% higher than that in Experiment 1.

Typical messaging policies in the first ten generations of each run encode fitness in two bits (e.g. "11" for "ready to mate" and "01" otherwise). As evolution proceeds, more advanced policies may emerge, leading to better performance than the full observation baseline. Table 4 shows a messaging policy from a champion in the 100th generation.

F	P	P'	R2	R1	**S2**	**S1**
0	0	-	-	-	1	0
0	1	-	-	-	1	0
1	0	-	-	-	0	0
1	1	-	-	-	1	1

Table 4: Champion Messaging Policy – Experiment 4. Dash indicates that an input has no effect on the outputs.

While the messaging policy in table 4 looks confusing by itself, interestingly, it exploits the setting of the jungle world effectively if combined with corresponding action policy. Although such a policy can be expressed by a table as before, it is translated into the following rule-based policy due to limited space.

1. Run towards the jungle until position is one.
2. Wait until partner position becomes one.
3. Enter the jungle if positions of both agents are one.
4. If "11" is received before entering the jungle, mate and enter the jungle at the same time step.

Note that Rule 1 to 3 are based purely on observation, i.e. they have nothing to do with the messages received because partner position is directly observable. In fact, the only rule that relies on communication is Rule 4. It can be triggered only under the circumstance where both agents are one step away from the jungle and "11" is received; and according to Table 4, "11" is sent whenever an agent is ready for both hunting and mating. Since messages are ignored in all other scenarios, the only messaging rule that a junior agent needs to learn is to send "11" when F and P are both one. This rule has two positive effects. First, the juniors learn the language faster because it tolerates faults on all but a few inputs (i.e. inputs with F and P equal to one). Second, combined with the action policy, it allows agents to mate and hunt successfully in a single trial, making the average fitness of the population close to the maximum (200) all the time.

Among the 20 runs in Experiment 4, approximately half of them (11/20) discovered policies with similar principles, thus achieving higher performance than Experiment 1 and 2. Note that without the emergence of meaningful language, agents in Experiment 1 or 2 cannot discover behaviors that accomplish hunt and mating in a single trial. The fact that languages can be evolved to get around deceptive local optima (e.g. policies that decide to mate whenever both agents are ready) is intriguing.

Additionally, results in Experiment 2 and 4 suggest that for language to emerge, it is essential that language is indeed necessary to perform some of the tasks. However, as long as language appears, it can be evolved into a beneficial tool for all tasks.

490

Future Work

Situated simulation of language evolution provides interesting insights on the origin and evolution of language. The jungle world simulation can be used as a starting point for more advanced simulations in two ways.

First, in nature, spoken language is formed with sequential patterns of utterances. Messages may span multiple time steps rather than contained in a single step. Also, they may start at any step. Such sequential features are essential for simulating the evolution of more complex and structured languages.

Second, languages in the real world are usually structured based on syntax. The emergence of grammatical components and structures such as nouns and verbs, subjects and objects, phrases and sentences is an important aspect of language evolution. A possible approach in the jungle world is to establish social roles in the simulation, and create tasks around them. Grammatical structure might then emerge in order to communicate such role-based information (Bickerton 1990).

Integrating sequential and/or structural features into the jungle world framework will make simulations more realistic and informative.

Conclusion

This paper presents a framework for situated simulation of language evolution. It introduces an artificial environment, the jungle world, which can be used to simulate the evolution and acquisition of multitask languages. The paper also proposes a method: Evolutionary Reinforcement Learning with Potentiation and Memory (ERL-POM) for simulation of language evolution in this environment.

Experimental results indicate that languages can be evolved in the artificial environment if communication is necessary for some or all of the tasks. Languages can be used to coordinate efforts in multiple tasks where communication is required. When communication is not necessary for all tasks, languages can be leveraged to overcome local optima and discover better policies. Experimental results also show that memory and potentiation are necessary for such emergence. Extending the simulation to sequential and structured communication is a most interesting direction of future work.

Acknowledgement

Reference

Batali, J. (1998), Computational simulations of the emergence of grammar. In J. R. Hurford, M. Studdert-Kennedy, & C. Knight (Eds.), *Approaches to the evolution of language*, page 405–426. Cambridge, UK: Cambridge University Press.

Bickerton, D. (1990). *Species and Language*. Univ. of Chicago Press, Chicago.

Brighton, H. (2002). Compositional syntax from cultural transmission. *Artificial Life*, 8, page 25–54.

Cangelosi, A., Parisi, D. (2002), *Simulating the evolution of language*, 2002 edition, page 5 – 8, Springer.

Chomsky, N. (1972). *Language and mind*. New York: Harcourt Brace Jovanovich.

De Boer, B., & Vogt, P. (1999). Emergence of speech sounds in changing populations. *In Advances in artificial life,* page 664-673. Springer Berlin Heidelberg.

De Greeff, J., & Nolfi, S. (2010). Evolution of implicit and explicit communication in mobile robots. In *Evolution of Communication and Language in Embodied Agents* (pp. 179-214). Springer Berlin Heidelberg.

Kaplan, F. (2000). Semiotic schemata: Selection units for linguistic cultural evolution. In M. Bedau, J. McCaskill, N. Packard, & S. Rasmussen (Eds.), Artificial Life VII: *Proceedings of the Seventh Artificial Life Conference*, page 372–381. Cambridge, MA: MIT Press.

Kvasnicka, V. and Pospichal, J. (1999), An Emergence of Coordinated Communication in Populations of Agents. *Artificial Life* 5(4), page 319--342.

MacLennan, B., Burghardt, G. (1993), Synthetic ethology and evolution of cooperative communication, *Adaptive Behavior*, Vol 2, page 167-187.

Mirolli, M., Parisi, D. (2010), Producer biases and kin selection in the evolution of communication: how the phylogenetic and the adaptive problems of communication can be solved. In *Evolution of communication and language in embodied agents*, page 135–159. Springer Verlag.

Mitri, S., Floreano, D., & Keller, L. (2010). Evolutionary conditions for the emergence of communication. In *Evolution of Communication and Language in Embodied Agents*, page 123-134. Springer Berlin Heidelberg.

Nolfi, S., Mirolli, M. (2010), *Evolution of communication and language in embodied agents*. Berlin: Springer.

Quinn, M. (2001). Evolving communication without dedicated communication channels. In J. Kelemen & P. Sosí k (Eds.), *Advances in artificial life: The Sixth European Conference* (ECAL 2001), page 357–366. Berlin: Springer.

Rawal, A., Boughman, J., Miikkulainen, R. (2014), Evolution of communication in mate selection. In *Proceedings of the fourteenth international conference on the synthesis and simulation of living systems* (ALIFE 14), Cambridge, MA: MIT Press.

Smith, A. D. M. (2001). Establishing communication systems without explicit meaning transmission. In J. Kelemen & P. Sosí k (Eds.), *Advances in artificial life: The Sixth European Conference* (ECAL 2001), page 381–390. Berlin: Springer.

Smith, K. (2002). The cultural evolution of communication in a population of neural networks. *Connection Science*, 14(1), page 65–84.

Steels, L., & Oudeyer, P.-Y. (2000). The cultural evolution of syntactic constraints in phonology. In M. Bedau, J. McCaskill, N. Packard, & S. Rasmussen (Eds.), *Artificial life VII: Proceedings of the Seventh Artificial Life Conference*, page. 382–391. Cambridge, MA: MIT Press.

Wagner, K., Reggia, J., Uriagereka, J., Wilkinson, G. S. (2003), Progress in the simulation of emergent communication and language, *Adaptive Behavior*, Vol 11, page 37-69.

Understanding Language Evolution in Overlapping Generations of Reinforcement Learning Agents

Lewys G. Brace[1] and Seth Bullock[2]

[1]Institute for Complex Systems Simulation, University of Southampton, Southampton, United Kingdom.
[2]Department of Computer Science, University of Bristol, Bristol, United Kingdom.
L.G.Brace@soton.ac.uk

Abstract

Understanding how the dynamics of language learning and language change are influenced by the population structure of language users is crucial to understanding how lexical items and grammatical rules become established within the context of the cultural evolution of human language. This paper extends the recent body of work on the development of term-based languages through signalling games by exploring signalling game dynamics in a social population with overlapping generations. Specifically, we present a model with a dynamic population of agents, consisting of both mature and immature language users, where the latter learn from the formers' interactions with one another before reaching maturity. It is shown that populations in which mature individuals converse with many partners are more able to solve more complex signalling games. While interacting with a higher number of individuals initially makes it more difficult for language users to establish a conventionalised language, doing so leads to increased diversity within the input for language learners, and that this prevents them from developing the more idiosyncratic language that emerge when agents only interact with a small number of individuals. This, in turn, prevents the signalling conventions having to be renegotiated with each new generation of language users, resulting in the emerging language being more stable over subsequent generations of language users. Furthermore, it is shown that allowing the children of language users to interact with one another is beneficial to the communicative success of the population when the number of partners that mature agents interact with is low.

Introduction

The fact that children around the world are readily able to learn the language of their given social group, even though these languages are in a constant state of flux (Hopper, 1987) and exhibit high levels of variation, flexibility of usage, and are ever changing over time within dynamic populations (Christiansen and Kirby, 2003) indicates that cultural factors play a crucial role in the shaping of human language. The establishment of the meanings of lexical items and the subsequent change in these meanings over time is, in part, what led Lewis (1969) to work on the conventionality of meaning; how specific arbitrary signals establish themselves as referring to a specific meaning. He introduced a signalling

game in order to explore how meaningful language might evolve from the use of initially random signals. Over the last decade, renewed interest in these ideas has led to a body of work that has explored the evolution of term-based languages through coordination games (Skyrms, 2004, 2009, 2010; Huttegger, 2007; Barrett, 2006, 2009; Argiento et al., 2009).

In this paper, we further develop this work by implementing a reinforcement learning (R-L) model involving a single Sender-Receiver pair, which is then extended into a population-based, multi-generational, simulation. This is both novel and necessary, given that human language persists in a complex social milieu, which is not captured by standard R-L models, and employing the R-L procedure in a population-based model could therefore offer insights into how lexical items become established within the context of the cultural evolution of human language in structured populations with overlapping generations.

This paper presents a model that demonstrates that, if agents only interact with a small number of other agents, then it is easier for these agents to establish a conventionalised system of language usage than in cases where they interact with a larger number of the population. However, by interacting with a smaller subset of the population, these individuals develop a more idiosyncratic language. Thus, it becomes difficult for the children of these agents, who learn from the interactions of their parents, to communicate with children of other mature agents during future epochs. In contrast, allowing individuals to interact with a larger proportion of the population does initially make it more difficult to establish agreed upon conventions of usage, but it does result in an increased amount of diversity within the language learner's training input data. This better enables the children of these mature agents to successfully interact with the offspring of other mature agents, previously unencountered by the agent in question; this aids the negotiation of conventional signalling in the population as a whole. This, in turn, leads to the development of a language that is more stable and consistent over generations of language users, compared to the case where individuals have to rene-

gotiate conventions of use in every generation.

The $n = 2$ game

In a Lewis signalling game there are two players, a Sender and a Receiver. A single bout of the game commences with the Sender knowing that the world is in some random state, t, but the Receiver being ignorant of this information. The Sender then selects a signal, s, with which to convey the world state to the Receiver; the Receiver observes s and has to pick an appropriate action, a. If the action chosen by the Receiver matches the world state (i.e., $a = t$), the bout is considered to have been a success. Here, t, s, and a are drawn from finite sets T, S, and A, respectively, which are all of size n; in Lewis' (1969) original model $n = 2$.

Over successive bouts of the game, both players are expected to adapt their behaviour in order to increase the chance of achieving communicative success, typically through some kind of reinforcement learning. The easiest way to conceptualise this is in terms of urns and balls. At the outset of the simulation run, an unbiased Sender will have n urns, one for each state of the world, each of which contains n balls, one associated with each of the n possible signals. Let's suppose that during the first bout of the game, $t = $ "red". The Sender picks a random ball from their *red* urn. The symbol on this ball dictates the signal to be made, s; in this case, suppose $s = $"fah". Likewise, the Receiver observes $s = $ "fah", and picks a random ball from their *fah* urn, which indicates the action to be taken, a. Both balls are then returned to their respective urns. If $a = t$, the inter-action was a success, and in accordance with the principles of Roth-Erev reinforcement learning (Roth and Erev, 1995), the Sender adds extra balls of type s to urn t and the Receiver adds extra balls of type a to urn s. The number of extra balls, u, added to the urns corresponds to the utility associated with the outcome of the signalling bout; in Lewis' (1969) original game $u = 1$ if a bout is successful and $u = 0$ otherwise. More formally, at any point in time, $b(t, s)$ is the number of balls for signal s in the Sender's urn for state t, and accordingly, $b(s, a)$ is the number of balls corresponding to act a in the Receiver's urn s. Thus, the behavioural strategies for Sender (σ) and Receiver (ρ) are as follows:

$$\sigma(t, s) = \frac{b(t, s)}{\sum_{s' \in S} b(t, s')} \qquad \rho(s, a) = \frac{b(s, a)}{\sum_{a' \in A} b(s, a')} \tag{1}$$

There are a number of possible signalling equilibria that can arise in such a game. Perfect signalling strategies result in optimal pay-offs for the players by mapping each world state onto a unique signal and each signal onto the unique appropriate action (Figure 1). This behaviour constitutes an evolutionarily stable strategy (ESS) because when it is played by the whole population there is no incentive for any individual to change their strategy.
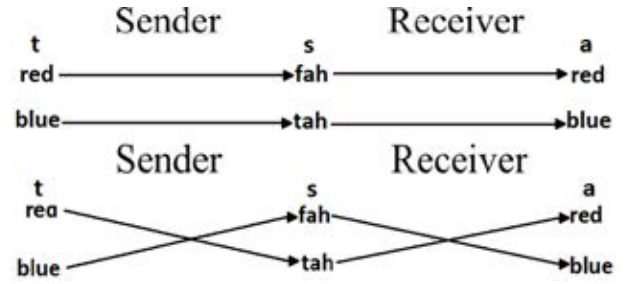

Figure 1: Optimal strategies for the $n = 2$ game.

However, players may spend significant time playing sub-optimal 'pooling' strategies in which Senders employ the same signal for multiple world states (pooling these world states together), making it impossible for Receivers to determine the state of the world from the signals that they receive (Figure 2). Pooling strategies are not ESSs since adjacent strategies often achieve equal fitness (e.g., the two pooling strategies in fig 2). The expected pay-off for such a pooling strategy is 0.5 when $n = 2$.
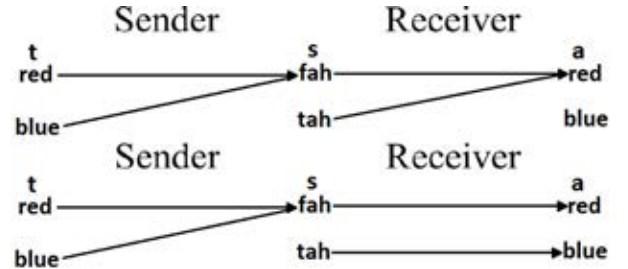

Figure 2: Two of the possible sub-optimal pooling strategies for the $n = 2$ game.

It has been shown by way of both computational simulation (Barrett, 2006, 2009; Skyrms, 2010) and mathematical modelling (Huttegger, 2007; Argiento et al., 2009) that the $n = 2$ game will nearly always converge upon an optimal signalling system. Indeed, Skyrms (2009) went on to demonstrate that this behaviour also holds in a case where there are two Senders and one Receiver. These results are further supported by Table 1, where it can be seen that a computational model of the $n = 2$ game being played for 10^6 bouts will almost always reach a perfect signalling equilibrium.

Higher-n games

However, a successful outcome is not always achieved when the game is played with $n > 2$, i.e., with a higher number of states, signals and actions (Skyrms, 2010; Huttegger, 2007; Barrett, 2006, 2009). Here, we adopt the methodology of Barrett (2006, 2009). The R-L model that formed the basis of the population-based R-L model was run multiple times, for various values of n, with each run consisting

493

of 10^6 bouts, B, of the game, where a run of the simulation is considered to fail if the number of successful bouts is less than 90% of the total number of bouts. Table 1 (left) shows the results of these runs, which agree with those of Barrett (2006, 2009). Table 1 (right) shows the results of a smaller sample of 100 runs, with all other parameters being held constant, and results that are quite similar. It can be seen clearly from Table 1 that, in a $n=3$ game, the players fail to achieve a high enough rate of signalling success roughly 10% of the time, and that this increases to $\approx 20\%$ for $n=4$ games, $\approx 60\%$ for $n=8$ games, and so on. The comparison in Table 1 is important to show, as the extended model presented later is run for 100 runs due to limits on computational power.

	1000 runs			100 runs
$n = 2$	0.999		$n = 2$	0.99
$n = 3$	0.881		$n = 3$	0.87
$n = 4$	0.784		$n = 4$	0.84
$n = 8$	0.391		$n = 8$	0.33
$n = 10$	0.281		$n = 10$	0.22
$n = 20$	0.264		$n = 20$	0.23

Table 1: Table depicting the success rates of the R-L model after 10^6 bouts for various values of n, with 1000 runs (left) and 100 runs (right).

Generational Reinforcement Learning Model

Although interesting in their own right, the dyadic setting considered so far limits the conclusions that can be drawn. After all, human language persists in a highly complex social milieu, and it has been shown that the structure and composition of a population can influence the dynamics of language change over time (Brace et al., 2015). Thus, the original reinforcement learning model (R-L) was extended in a number of ways. First, whereas the original model focused on a single Sender and Receiver playing for B bouts, in the population model (R-L-P), there are a population of agents. This population is divided into a number of mature agents, N_M, and a number of immature agents, N_I; with all agents starting life as immature and then being promoted to mature status after the first epoch of their existence. Mature agents play bouts of the language game with one another, updating their language behaviour according to game outcomes. By contrast, while agents are immature they merely observe the language bouts played by their mature parent, and update their language behaviour on the basis of the outcomes of these observed games. The lifespan of agents is two epochs; the first as an immature agent and the second as a mature agent, after which they are removed from the simulation.

It is important to emphasise here that throughout the simulation, when new immature individuals are introduced to the population, as in the standard R-L model, they have *no knowledge of the language currently being used*. This is true for the initial population of mature agents, and also true for new immature agents born into all subsequent epochs. For each immature agent, each world state, t, is associated equally with each signal, s, when playing as Signaller, and each signal, s, is associated equally with each action, a, when playing as Receiver, i.e., each of an immature agent's n state urns contains a single ball for each possible signal, and each of their signal urns contain a single ball for each possible action. Thus, any change in communicative performance or language use over generations is the result of language evolution; there is no biological evolution on the part of the agents.

Secondly, instead of agents merely interacting B times, the R-L model is extended to include a generational aspect. In other words, the model is set up to run for a number of epochs, E, and during each epoch, every mature agent plays B bouts as the Receiver with other mature agents; with the amount of bouts it plays as the Speaker being the result of how many other agents it is partnered with divided by B. The number of different mature agents that a mature agent interacts with, P, is a key parameter of the model; with each mature agent's total number of interactions, B, being equally divided amongst its P unique partners, i.e., the number of interactions that an agent has with each partner is B/P (rounding up). A key feature of the model is therefore that varying P does not vary the number of bouts played, just the number of players that the bouts are played with.

The R-L-P model thus proceeds as follows. At the start of the simulation run, an initial population of $N_M = 15$ unbiased mature agents are created, with an equal chance of generating each signal for each world state. For each epoch, E, a fresh population of $N_I = 15$ unbiased immature agents is created, each having an equal chance of generating each signal for each world state. Each immature agent is assigned a randomly selected mature agent to act as their parent; with each mature agent acting as a parent to only one immature agent. Each mature agent is assigned P unique randomly selected mature partners with which to play the signalling game. Each of the mature agents then engages in B/P bouts with each assigned partner, with each participant updating their signalling or receiving strategy at the end of each bout through reinforcement learning. Each child will update their behaviour based on the outcome of the bouts that their parent are involved in; i.e., at the end of a successful bout, a Sender's child will add a ball of type s to urn t, and a Receiver's child will add a ball of type a to urn s. At the end of an epoch, all mature agents are removed, all immature agents are promoted to mature agent status, and a new set of unbiased immature agents is created.

Results

The R-L-P model does not achieve a successful signalling system as often as the standard R-L model. Indeed, compar-
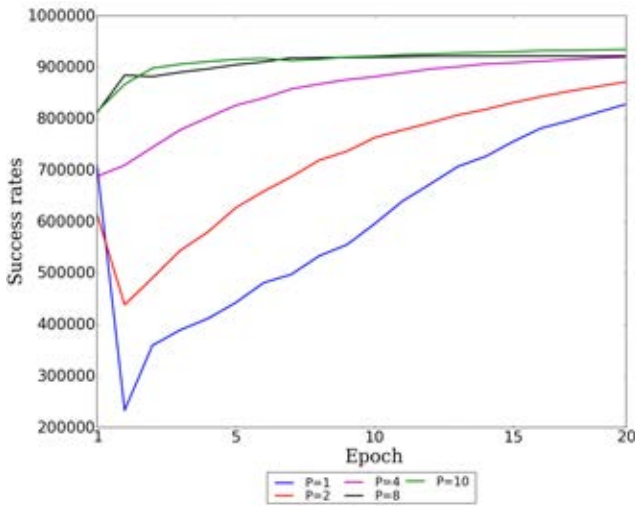
Figure 3: Graph depicting the average number of successful bouts across epochs for $P \in \{1, 2, 4, 8, 10\}$ for a $n = 20$ game with $N_M = 15$, $N_I = 15$, $B = 10^6$, and $u = 1$. Averaged over 30 runs.
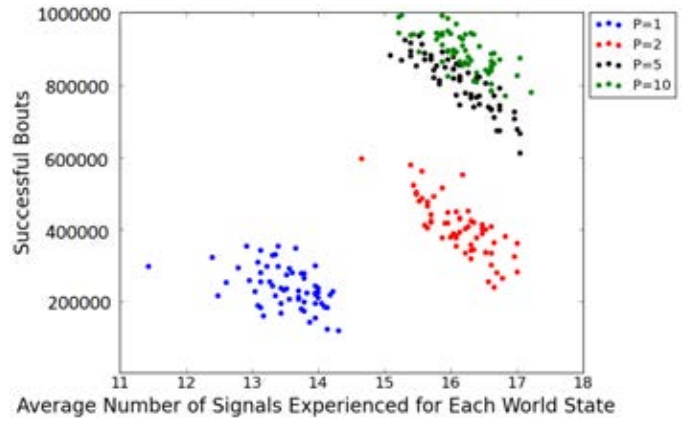

Figure 4: Graph depicting the average percentage of successful communicative bouts between all mature agents plotted against the number of unique signals presented to them during said bouts in the second epoch for a $n = 20$ game, for P=1, 2, 5, 10 and $N_M = 15$, $N_I = 15$, $B = 10^6$, and $u = 1$. Averaged over 60 runs.

ing Table 2 to Table 1 shows how, with $P = 1$, success rates are lower for all n-games than in the standard R-L model.

	Partners=1	Partners=2	Partners=5	Partners=10
$n = 4$	0.3	0.36	0.77	0.81
$n = 8$	0.0	0.3	0.48	0.62
$n = 10$	0.0	0.2	0.20	0.46
$n = 20$	0.0	0.0	0.29	0.57

Table 2: Language evolution success rates after 100 R-L-P model runs of $E = 20$ epochs each for various values of n and P, where $N_M = 15$, $N_I = 15$, $B = 10^6$, and $u = 1$. With success being measured using the aforementioned metric used in Table 1.

However, increasing the value of P does increase the rate of success (Table 2 and Figure 3). In Figure 3, we see that, with $P = 1$ or 2, there is an initial level of success, which corresponds to the number of successful bouts that would be seen in the normal R-L model for a n=20 game.

Low P values create a situation where mature agents form a communicative system based upon conventions agreed upon between themselves and only a small number of other agents. Thus, in subsequent epochs, when the children of a mature agent have to interact with the children of another mature agent, who has not previously interacted with the mature agent in question, the agreed upon conventions that both parties formulated during the first epoch are likely to be of little use, due to different agents forming conventions based upon their idiosyncratic experiences. This gives rise to sub-optimal behaviour at the population-level. However, any immature agents that are present learn from the success-

ful bouts of their respective parents, hence the steady increase in success rates for these lower P values[1].

In contrast, with high P values, we see an obvious and immediate increase in communicative success. This is due to the way an increase in P leads to the children of the mature agents having more diversity in their training input, which better enables these individuals to communicate with a larger number of other agents upon being promoted to mature agent status (Figure 4).

Imagine a hypothetical mature agent from epoch one, who is partnered with ten other randomly selected agents, who in turn, are partnered with ten other agents. In the simulation, bouts are scheduled in such a way that $agent_1$ will have one of the allocated bouts with one of its randomly selected partners, then $agent_2$ will do the same; and so on, until we reach $agent_{M_N}$. At which point we go back to $agent_1$ and allow it to have its second bout, again with a randomly selected partner; and so on until each partner of every agent has played B/P bouts with the agent. In the P=1 case, unsurprisingly, we see higher levels of initial success during the first generation than in the P=10, due to the establishment of a convention involving fewer agents having to negotiate with one another (Figure 5, left).

In contrast, with $P = 10$, it is slightly harder to establish a conventionalised usage because each agent has to negotiate with an increased number of different agents, which results higher levels of signal diversity (Figures 4 and 5, left). However, when the offspring of the first epoch's mature agents are forced to interact with a different subset of the population in the second epoch, populations with higher

---

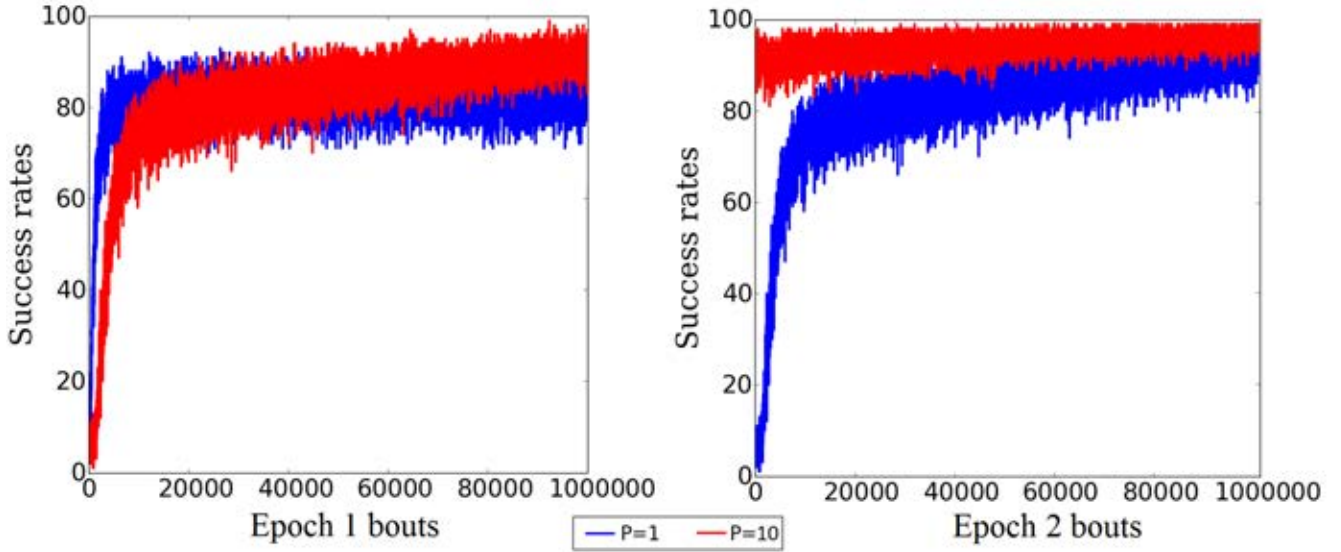[1]Given enough epochs, it is likely that the agents would give rise to a successful communicative system.

Figure 5: Graph depicting the number of successful bouts out of every 100 bouts, over all 10^6 bouts of a random agent during the first epoch (left) and the second epoch (right); with P=1 (blue line) and P=10 (red line). Where $n = 20$, $N_M = 15$, $N_I = 15$, $B = 10^6$, and $u = 1$.

P values exhibit higher communicative success because the increased signal diversity in the previous epoch, combined with the immature agents learning from the successful bouts of their parents, has resulted in these agents having established a conventionalised usage that requires less renegotiating when speaking to previously unencountered agents than in the $P = 1$ case, where agents have a more idiosyncratic language that requires them to renegotiate the conventions established by their parents (Figure 5, right).

This is why Figure 4 shows an increase in communicative success with higher values of P, while also indicating a negative trend in each of the data clusters for each specific P value; although it is harder to establish a language when negotiating meaning-signal pairs with more individuals, doing so makes it more stable across generations (Figure 5). Indeed, as Figure 5 (right) shows, the agreed upon convention of usage in cases of lower P values has to be renegotiated in subsequent epochs due to it offering little communicative success to agents when communicating with newly encountered individuals.

It is important to note that the increase in communicative success is the result of higher P values and not of another variable, such as B. Indeed, Figure 6 demonstrates the average level of communicative success over twenty epochs is significantly lower for $P = 1$ or 2, as compared to $P = 4$; a trend that continues as P is increased. Furthermore, it can be seen from Figure 6 that higher P values allow for an increased amount of communicative success even when agents have significantly fewer training sessions (lower B values).

However, in the real world, children are not just passive receivers of linguistic input. They interact with others, in-cluding other children, who may not yet be fully linguistically competent. Thus, a number of model runs were conducted whereby immature agents had B bouts with P other immature agents while witnessing their parents bouts (Figure 7). In other words, bouts are scheduled in a similar manner to that described above, in that we allow each agent to have one bout with a randomly selected partner, starting with $agent_1$ and cycling through to $agent_{N_M+N_I}$, before going back to $agent1$ again. In these runs, mature agents only interact with mature agents and immature agents only interact with other immature agents. Although, immature agents still learn from their parent's interactions, in the manner detailed above.

Figure 7 demonstrates how, performance in the $P = 10$ case is impeded by allowing interactions between immature agents. This is to be expected as linguistically underdeveloped individuals interacting with one another will add a degree of noise into the communicative system.

However, with $P = 1$, allowing immature agents to interact with one another dramatically increases communicative success. This difference in behaviour can again be attributed to signal diversity. While in the above results immature agents only learned from the interactions of their parents, meaning they got a degenerative sample of the language because they only ever witnessed the same two individuals communicating during their first epoch, here they are also interacting and learning with another individual who is likely to have witnessed two different mature agent's interacting with one another. This would increase the amount of signal diversity in the immature agents training data.
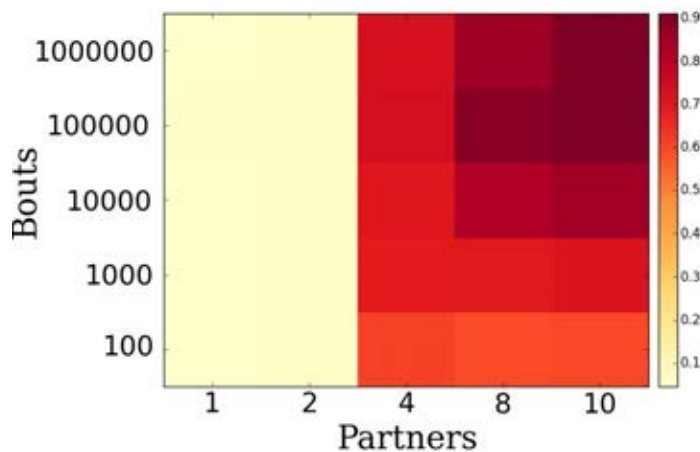
Figure 6: Graph depicting the average amount of communicative success over 20 epochs for various values of B and P. Where $n = 20$, $N_M = 15$, $N_I = 15$, $B = 10^6$, and $u = 1$. Averaged over 30 runs.

Discussion

The results presented here build upon a larger body of work; both in regards to signalling conventions (Skyrms, 2004, 2010; Barrett, 2006, 2009) and expression/induction models research in general (Hurford, 2002). It has been shown that a signal can acquire a conventionalised meaning without the Sender intending for it to do so. Moreover, the meaning of such simple signals is dependent upon the stabilisation of usage conventions, which emerge from functional historical signal production. Thus, even the most automatic or reflexive signals can acquire meaning, so long as the production and response mechanisms are co-adapted to coordinate their behaviours in accordance with such an arbitrary signal (Harms, 2004).

More interestingly, it has been shown that a population structure that allows for interaction between more of its members is beneficial in allowing it to evolve an efficient term-based language.

More specifically, it has been shown that, as intuition dictates, while it is harder to establish a conventionalised system of usage with larger numbers of individuals, doing so enables the emerging language to persist in subsequent epochs. This is due to the input into language learners being initially more diverse, which prevents these learners from developing a more idiosyncratic, communicative system that makes it harder to communicate with previously unencountered individuals.

In addition, the results and behaviour reported here can be seen to be linked to the concept of a linguistic bottleneck. This refers to how the input data for a language learner will only be a subset of the potentially large range of grammars of the Speaker from which it is learning.
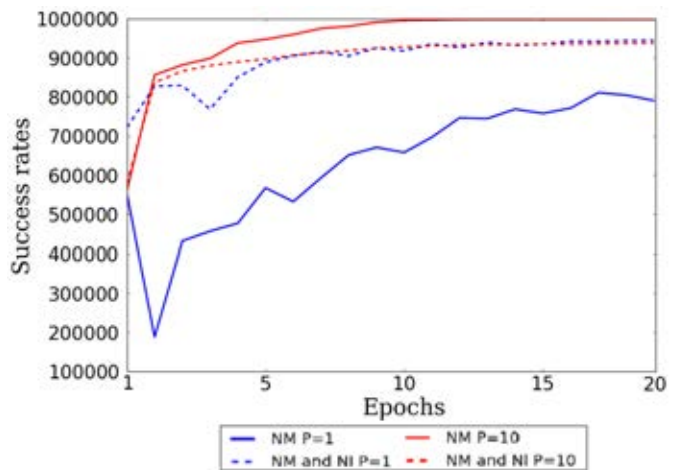
A series of computer-based simulations that use the

Figure 7: Graph depicting the average amount of communicative success over 20 generations for $P = 1$ where only N_M interact with one another (blue solid line) and where both N_M and N_I interact with other mature and immature agents, respectively (blue dashed line), and likewise for $P = 10$ (red solid line and red dashed line, respectively). Where $n = 20$, $N_M = 15$, $N_I = 15$, $B = 10^6$, and $u = 1$. Averaged over 30 runs.

method of *iterated learning* have shown that the linguistic bottleneck is crucially important in regards to whether or not language can be successfully passed from one generation to the next and, in situations where this transmission can be achieved successfully, show that it is also crucial to the linguistic structure that arises (Kirby, 2002b,a; Kirby and Hurford, 2002; Kirby et al., 2014; Smith, 2002; Smith et al., 2003; Brace et al., 2015).

Although a similar effect to the bottleneck is seen in other types of uni-generational models, such as the *naming game* (Steels, 1995), the model presented here is novel in that it demonstrates the impact of bottleneck-like behaviour in a generational-based simulation that explores term-based languages. Here, this bottleneck-like behaviour takes the form of the way in which internal representations of individuals are induced from limited examples of the behaviour of other agents (Hurford, 2002).

This supports other work that has demonstrated a link between the linguistic bottleneck and the number of linguistic tutors (Brace et al., 2015). Indeed, the behaviour seen in Figure 7 indicates that the factors underpinning the cultural transmission of language change and linguistic variation are perhaps too complicated to be understood by analysing the nature of just inter- and intra-generational transmission; and that further research into linguistic change should focus on the nature of the social network the underpins linguistic populations (Wichmann and Holman, 2009; Lupyan and Dale, 2010; Reitter and Lebiere, 2010; Milroy, 2013).

This point becomes more important given that traditional

expression/induction models have largely ignored population dynamics so as to function on other aspects of language. Although, given the aims of such models, it made logical sense to opt for more simplistic population structures, it has been shown here that population dynamics can have a significant impact upon communicative behaviour.

Indeed, it would be interesting to explore how an expanding and contracting population size, with varying numbers of mature language users and immature language learners, could impact the emergence and form of a language (Johansson, 1997; Hurford, 2002). An expression/induction model geared towards this interest could provide valuable insights for a growing body of research that is interested in the nature of the relationship between language and population change; such as the impact of population size on linguistic forms or the way in which periods of linguistic simplification tend to coincide with periods where there are a higher number of language-learners within a population (Johansson, 1997; Nettle, 1999; Wichmann and Holman, 2009; Lupyan and Dale, 2010; Milroy, 2013; Trudgill, 2013). These themes will form the basis of future work.

References

Argiento, R., Pemantle, R., Skyrms, B., and Volkov, S. (2009). Learning to signal: Analysis of a micro-level reinforcement model. *Stochastic processes and their applications*, 119:373–390.

Barrett, J. (2009). The evolution of coding in signaling games. *Theory and decision*, 67:223–237.

Barrett, J. A. (2006). Numerical simulations of the Lewis signaling game: Learning strategies, pooling equilibria and the evolution of grammar. *University of California eScholarship*, page https://escholarship.org/uc/item/5xr0b0vp.

Brace, L., Bullock, S., and Noble, J. (2015). Achieving compositional language in a population of iterated learners. *Proceedings of the European conference on artificial life 2015*, 349-356.

Christiansen, M. and Kirby, S. (2003). Language Evolution: the Hardest Problem in Science? In Christiansen, M. and Kirby, S., editors, *Language Evolution*. Oxford University Press, Oxford.

Harms, W. (2004). Primitive content, translation, and the emergence of meaning in animal communication. In O., K. and Griebel, U., editors, *Evolution of communication systems: a comparative approach*. The MIT Press, London.

Hopper, P. (1987). Emergent grammar. *Berkeley Linguistics Society*, 13:139–157.

Hurford, J. R. (2002). Expression/induction models of language evolution: Dimensions and issues. In Briscoe, T., editor, *Linguistic evolution through language acquisition: formal and computational models*. Cambridge University Press, Cambridge.

Huttegger, S. (2007). Evolutionary explanations of meaning. *Philosophy of science*, 74:1–27.

Johansson, C. (1997). *A view from language: growth of language in individuals and populations*. Lund University Press, Lund.

Kirby, S. (2002a). Learning, bottlenecks and the evolution of recursive syntax. In Briscoe, T., editor, *Linguistic evolution through language acquisition: formal and computational models*, pages 173–204. Cambridge University Press., Cambridge.

Kirby, S. (2002b). Natural language from artificial life. *Artificial life*, 8(2):185–215.

Kirby, S., Griffiths, T., and Smith, K. (2014). Iterated learning and the evolution of language. *Current opinion in Neurobiology*, 28:108–114.

Kirby, S. and Hurford, J. (2002). The emergence of linguistic structure: an overview of the iterated learning model. In Cangelosi, A. and Parisi, D., editors, *Simulating the Evolution of Language*. Springer, London.

Lewis, D. (1969). *Convention: a philosophical study*. Harvard University Press, Harvard, USA.

Lupyan, G. and Dale, R. (2010). Language structure is partly determined by social structure. *PloS one*, 5(1):e8559.

Milroy, L. (2013). Social networks. In Chambers, J. and Schilling, N., editors, *The handbook of language variation and change*. Wiley-Blackwell, Oxford.

Nettle, D. (1999). Is the rate of linguistic change constant? *Lingua*, 108:119–136.

Reitter, D. and Lebiere, C. (2010). Did social networks shape language evolution?: a multi-agent cognitive simulation. In *Proceedings of the 2010 Workshop on Cognitive . . .* , pages 9–17. Association for Computational Linguistics, Stroudsburg, USA.

Roth, A. and Erev, I. (1995). Learning in extensive form games: experimental data and simple dynamical models in the intermediate term. *Games and economic behaviour*, 8(1):164–212.

Skyrms, B. (2004). *The stag hunt and the evolution of social structure*. Cambridge University Press., Cambridge.

Skyrms, B. (2009). Evolution of signalling systems with multiple senders and receivers. *Philosophical transactions of the royal society of London. Series B*, 364:771–779.

Skyrms, B. (2010). *Signals: Evolution, Learning, and Information*. Oxford University Press, Oxford.

Smith, K. (2002). Natural selection and cultural selection in the evolution of communication. *Adaptive behavior*, 10(1):25–45.

Smith, K., Kirby, S., and Brighton, H. (2003). Iterated learning: a framework for the emergence of language. *Artificial life*, 9(4):371–86.

Steels, L. (1995). A self-organizing spatial vocabulary. *Artificial Life*, 2:319–332.

Trudgill, P. (2013). Linguistic and social typology. In Chambers, J. and Schilling, N., editors, *The handbook of language variation and change*. Wiley-Blackwell, Oxford.

Wichmann, S. and Holman, E. (2009). Population size and rates of language change. *Human biology*, 81(2):259–274.

Modeling the Evolution of Gene-Culture Divergence

Chris Marriott[1] and Jobran Chebib[2]

[1]University of Washington, Tacoma, WA, USA 98402
[2]University of Zürich, Zürich, Switzerland 8057
dr.chris.marriott@gmail.com

Abstract

We present a model for evolving agents using both genetic and cultural inheritance mechanisms. Within each agent our model maintains two distinct information stores we call the genome and the memome. Processes of adaptation are modeled as evolutionary processes at each level of adaptation (phylogenetic, ontogenetic, sociogenetic). We review relevant competing models and we show how our model improves on previous attempts to model genetic and cultural evolutionary processes. In particular we argue our model can achieve divergent gene-culture co-evolution.

Introduction

Evolutionary computation, a field that exploits the power of evolution, is a powerful tool for optimization, creativity and the study of evolutionary forces. Holland (1975) helped popularize evolutionary computation by applying the principles of a basic evolutionary model to a computational task. Dawkins (1976) also considered the minimal requirements of evolution and proposed a simple model.

Dawkins applied his model to the domain of human culture introducing the field of memetics (see also Dennett (1995)). Dawkins suggests that as separate domains the realm of genetics and the realm of memetics both follow the same evolutionary principles. Theories of genetic and cultural co-evolution recognize that these two domains are not separate but parts of the same evolving system. While Dawkins' simplified model addresses a single evolving system in isolation, we are interested in a model that incorporates the interactions between genes *and* memes as both evolve.

There have been many attempts to characterize the nature of cultural evolution coming from diverse motivations. For instance the term memetic algorithms now refers to a field of combinatorial optimization. Moscato et al. (1989) defined this new model in this way:

> Memetic algorithms is a marriage between a population-based global search and the heuristic local search made by each of the individuals.

While the focus of Moscato was combinatorial optimization, where this model has proven of value, he has not created a model incorporating both genetic and cultural evolution. A model of cultural evolution must also incorporate an *exchange of learned information between individuals*. In this sense Moscato's model falls short of our aims. (It is worth noting that some implementations of memetic algorithms do incorporate an exchange of information.)

In developing our own model of genetic and cultural evolution (Marriott and Chebib, 2014) we have considered the characteristics of an acceptable model. In our opinion any acceptable model will incorporate three modes of adaptation: phylogeneitc (biological), ontogenentic (individual), and sociogenetic (social). Phylogenetic adaptation is the well known adaptation of genetic material through natural selection, which also known as biological evolution. Ontogenetic adaptation is adaptation of the individual over its lifetime and is often split into development (adaptation of morphology) and learning (adaptation of behavior). Sociogenetic adaptation is adaption of cultural information that is communicated through social learning mechanisms.

It is clear that these three modes of adaptation will be coupled, that is, they will impact one another (Hinton and Nowlan, 1987; Sznajder et al., 2012). We think that any acceptable model of genetic and cultural evolution must support divergence in the genetic and cultural evolutionary trajectories. There are types of divergence possible. If the selection pressures on the genetic information and the memetic information pull in the same direction we see divergence in the speed of evolution. Cultural evolution is typically much quicker in this case. Yet when the selection pressures on genetic information and cultural information pull in different directions the model should allow the genetic information and the cultural information to diverge.

In human culture we can see that this divergence can occur in individual humans. For instance, a Catholic priest may swear an oath of celibacy because his culture rewards him for it. A Samurai may kill himself if he feels his cultural obligations have not been met. Refusing to reproduce and killing oneself are both acts that are contrary to the ge-

netic imperative but in these cases support the individual's cultural imperative.

Other more drastic cases occur when a whole culture adopts behavior contrary to the biological imperative. This includes the celibate religious sect the Shakers from the 1770s and a number of mass suicides including Jonestown in 1978 and the Heaven's Gate cult in 1997. The tragic end of these cultures is usually their own destruction.

Other models of genetic and cultural evolution have been presented but very few satisfy all of our desired properties. In this paper we will review evolutionary models and evaluate them as models of gene-meme co-evolution using the two principles outlined above. After reviewing relevant models we will present our model and elaborate on how it integrates benefits of multiple models and satisfies our aims. Finally we will discuss the potential for divergence of genetic and cultural evolutionary trajectories with our model.

Prior Models

Dawkins suggested evolution can occur in any population of information bearing agents so long as the population had three properties: heredity, variation, and selection. This simple model of evolution we will call the *basic evolutionary model*.

To demonstrate the applicability of this model to non-biological populations Dawkins coined the term *meme* to name the unit of cultural selection. He argued that populations of memes satisfy the three properties and thus also undergo a process of evolution.

Dawkins application showed that evolutionary principles can be applied to other domains. However his model did not describe the dynamics of a system undergoing both genetic and cultural evolution. In this section we will review the basic evolutionary model as presented by Dawkins and then review models that expand upon this model. Our goals is to evaluate these expanded models for their value as models of gene-culture coupled systems.

Basic Evolutionary Model

Dawkins' evolutionary model is the basis of all the models we will consider in this section. The model exists as a set of minimal conditions for a system to evolve. The other models in this section are explorations of additional properties and mechanisms that enrich this basic model.

As mentioned above the basic evolutionary model requires a population of agents. These agents must bear some type of information that is critical to their survival, and they must possess a means of replicating this information. Dawkins calls these types of agents replicators.

A population of replicators is not sufficient. The population must also have three properties. When information is replicated it must be replicated with some degree of fidelity, that is, it must be replicated within some reasonable error rate (heredity). The information among agents in the
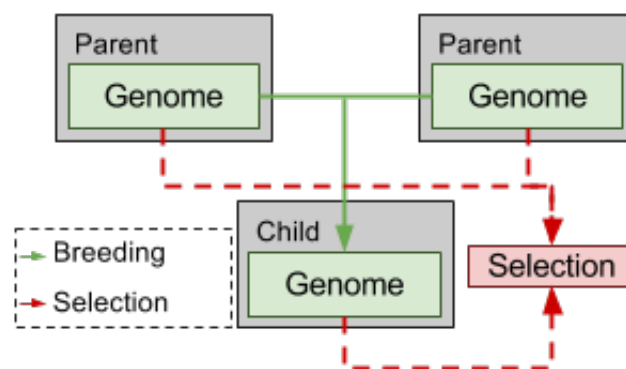

Figure 1: Basic Evolutionary Model

population must be varied (variation). Lastly, the information among agents in the population must determine which agents are pruned from the population and which agents can replicate (selection).

At an abstract level this model makes little commitment to how replication occurs and what the rules for selection are. This means, for instance, that both natural selection and artificial selection seem to satisfy the conditions. Nonetheless, the standard analogy is to biology so the basic evolutionary model tends to be characterized in terms of biological evolution.

In the basic evolutionary model the information is encoded in the genome (Fig. 1). One or more parents contribute information to a replication process that creates a new agent and genome. In the simplest model the information in the genome is directly referenced for the selection process (i.e. there is no interpretation of the information).

Dawkins suggests that this model is the minimum required for the force of evolution to occur, not that this is a complete or proper model of biological evolution. However researchers have implemented this basic model many times in silico demonstrating that this minimal model can lead to evolution. Many researchers, including Dawkins, have expanded on this standard model to describe biological as well as cultural phenomena (Lumsden and Wilson, 1981; Dawkins, 1982; Boyd and Richerson, 1983; Henrich and McElreath, 2003).

Agent Based Models

Agent based models are commonly used in biology and social sciences for modeling phenomena involving populations of agents (Bonabeau, 2002; Epstein, 2006; Niazi and Hussain, 2011; Smaldino et al., 2012). A typical agent based model consists of a population of agents. These agents, as in the basic evolutionary model, bear information that is used to make decisions or select behavior as they interact with their environment and each other.

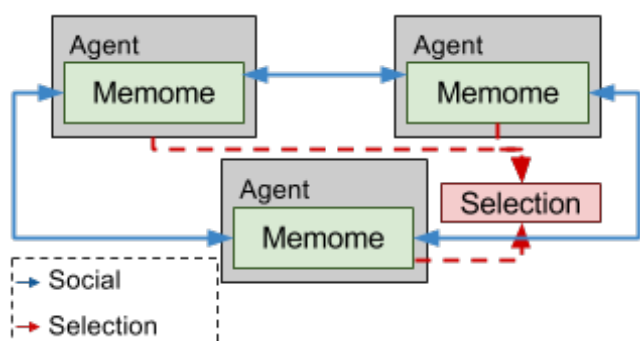Information is replicated in episodes of social interaction.

Figure 2: Agent Based Model

Figure 3: Horizontal Transfer Model

Agents exchange information with each other. Selection of behavior results in better or worse performance in the environment and so the information can evolve over time.

The information in an agent based model is now meant to represent learned information (not genetic) so we call it the *memome* (Fig. 2). While these models do not follow the standard biological characterization of the basic evolutionary model we suggest they nonetheless are explorations of variations on the basic evolutionary model.

Agents do not die and are not born in the simplest agent based models (some do incorporate this and some kind of genetic evolution but we would classify them in one of the categories that follow).

These agent based mechanisms satisfy the three properties of the basic evolutionary model. Information is copied with some fidelity in the replication process. Agents clearly have different information by the design of agent based models. Third, as argued above, selection also operates in these models. Our conclusion is that agent based models are explorations of the parameter space of the basic evolutionary model.

The value of agent based models is unquestioned. Nonetheless most agent based models exemplify the basic evolutionary model with a non-standard (i.e. non-biological) interpretation.

Horizontal Transfer Model

Horizontal transfer of information has been suggested as a hallmark of cultural evolution (Gonzalez et al., 2014). Horizontal transfer models are a blend between the basic evolutionary model and the social mechanisms used in agent based models. The horizontal transfer model (Fig. 3) still relies on a standard parent to child information transfer during reproduction (since this transfer is unidirectional and always passes from parents to children this is called a *vertical transfer*).

The social mechanisms of agent based models also transfer information from agent to agent but these transfers are bidirectional and can typically occur anytime, not just dur-
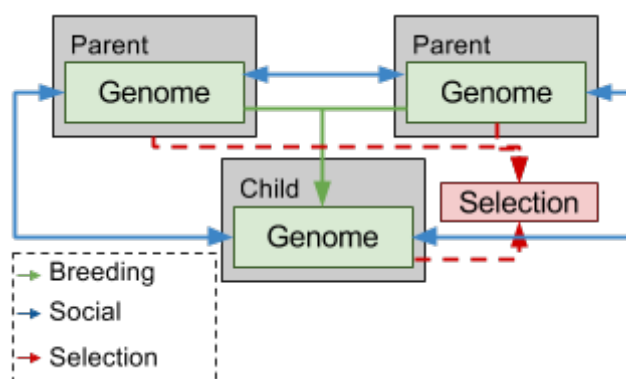
ing instances of reproduction. In particular this means that during the lifetime of an agent it can change its internal information through these social transfers.

This transfer is called *horizontal transfer* because it can occur during the lifetime of the agent and can occur between agents of the same generation. In particular information can be transferred in any direction including parent-to-child, child-to-parent, sibling-to-sibling and in general, agent-to-agent.

While horizontal transfer is a characteristic of cultural evolution we do not think that it is a sufficient characteristic. It can be shown that evolutionary models with horizontal transfer have some benefits over strictly vertical transfer models (Tomko et al., 2013). These simulated results are also backed by research on horizontal transfer of genetic material among bacterium, plants and fungi (Syvanen and Kado, 2001; Syvanen, 2012). The biological results stress that as horizontal transfer of genetic material does occur in the natural world we should treat horizontal transfer models as models of biological evolution. That is, the horizontal transfer model described here is a valuable enhancement of the vertical transfer interpretation in the basic evolutionary model but it does not model the gene-culture coupled system.

Evolutionary Developmental Model

Earlier we introduced the field of memetic algorithms. The memetic algorithm model adds an additional stage to the naive evolutionary model. Agents are bred and born as in the basic model. However, the genetic information is not the information used for selection (as it is in the previous models). Instead a local search is conducted around the genetic information for possibly better information. This information is instead used for selection.

In biology this is called the genotype-phenotype distinction and the process of mapping a genotype to a phenotype is called development (Hall, 2012). Development in biology is commonly split into morphological development (growth)
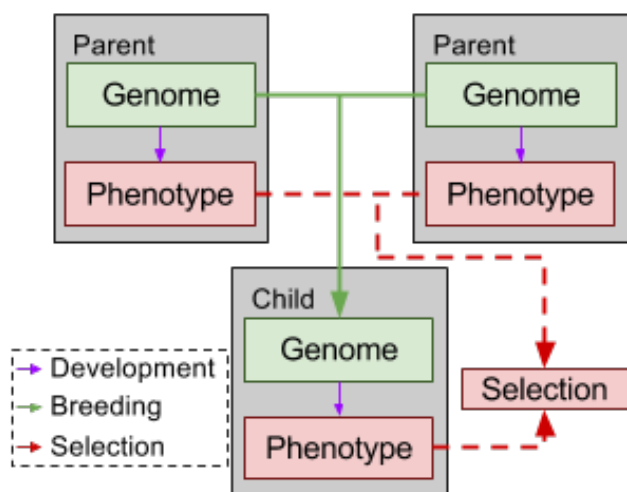
Figure 4: Evolutionary and Developmental Model

Figure 5: Lamarck Model

and behavioral development (learning). Adding both growth and learning to evolutionary simulations has been a common improvement over the basic evolutionary models (Hinton and Nowlan, 1987; Sznajder et al., 2012; Marriott and Chebib, 2014). The biological model that best captures evolution and development is called the evolutionary developmental model or evo-devo for short.

Fig. 4 shows the standard evo-devo model. The genome interacts with the environment through a process of development to produce the phenotype. The phenotype is a second store of information that is used in selection. However if reproduction occurs, it is only the genetic information that is passed on.

This makes a distinction between the information transferred in transfer events and the information used in evaluation for the purposes of selection. In the basic model this was the same information. In the evo-devo model we separate these two kinds of information into two different information stores as well as provide rules for how to develop a phenotype from a genotype.

A variant of the evo-devo model that is commonly found in memetic algorithms and other simulations corresponds to Neo-Lamarckian evolutionary theory. The primary distinction between Neo-Lamarckian and Darwinian evolution is that learned traits can be passed on in Neo-Lamarckian models. Fig. 5 shows a Neo-Lamarckian evo-devo model in which the information in the phenotype is transferred during reproductive events instead of the genotype.

A Neo-Lamarckian evo-devo model can be implemented with only a single information store (genetic) in which development is internal changes to the genome through interaction with the environment. When reproductive events occur the current state of the genome is transferred. In these models the information added through development can be passed on in reproduction.
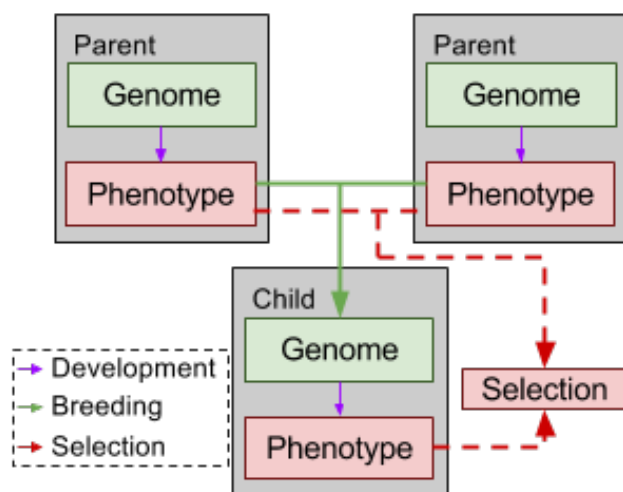
Evo-devo models are well studied in biology and are popular evolutionary models for simulations and optimization. These models capture phylogenetic and ontogenetic adaptation but do not model sociogenetic adaptation.

Evolved Social Learning Neural Networks

The models reviewed so far have been variants or expansions of the basic evolutionary model. Most have clear biological instances in the natural world. While all model important processes, none succeed at modeling the interplay between genetic and cultural information.

In our opinion the best attempts to model a coupled genetic and cultural system through simulation so far have come from researchers trying to evolve neural networks that also engage in phases of social learning (Gabora, 1995; Denaro and Parisi, 1997; Baldassarre, 2001; Smith, 2002; Acerbi and Parisi, 2006; Curran and O'Riordan, 2007; Borg et al., 2011). While these experiments have had varied levels of success we believe this type of model is on the right track.

Agents in this model have a genome that encodes an artificial neural network (typically the weights of a predetermined network topology). Evolution of this information is carried out following the basic evolutionary model.

However, during the lifetime of the neural network the network can engage in learning. A basic type of neural network learning is backpropagation learning. Backpropagation training requires a supervisor and most environments are not designed to supervise learning. In these models other networks provide the expected output to the learning network in a stage of social learning. Note that there are other possible means of training a network like neuromodulated plasticity (Soltoggio et al., 2008).

The primary advantage of this model is that there are two distinct information stores for genetic and cultural informa-
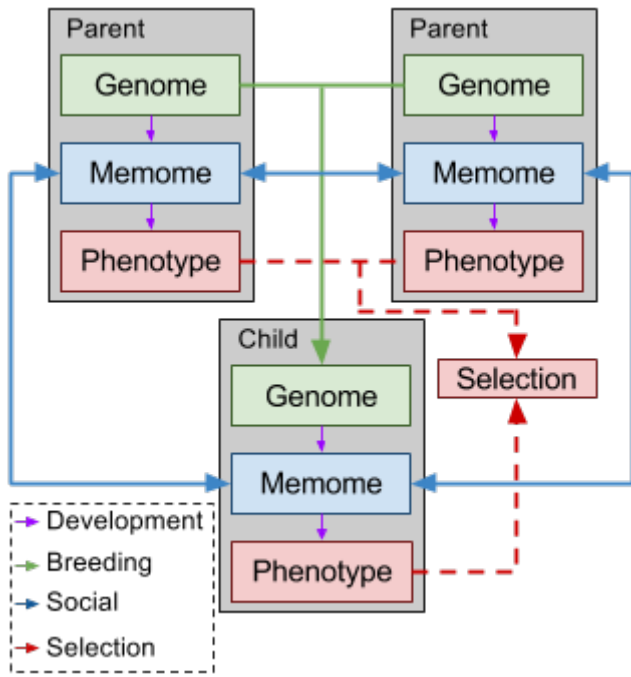
Figure 6: Dual Inheritance Model

(Fig. 6). Through a process of development the genome produces the newborn's memome. The genome remains inert over the lifetime of the agent while the memome is active in behavior selection and adaptive through learning. The memome interacting with the environment creates the phenotypic information which is active in selection. The memome is modified through interaction with the environment (learning) and through interaction with other agents (social learning).

We have implemented this model once before (Marriott and Chebib, 2014). Here we describe relevant implementation details from our current system (Marriott and Chebib, 2016b,a). Agents in our simulation exist in a random geometric network of food sites. During the day they spend energy moving around, foraging for food, breeding, learning and social learning. At the end of the day their food is converted to energy. An agent that runs out of energy dies, and one that stores enough surplus energy can reproduce.

Genome

The genome of an agent represents a path of sites through the random geometric network. At each site the gene determines what strategy to use to gather food, and whether to engage in breeding, learning or social learning at that site. The entire genome represents a very long path that cannot be completed in one day.

During an agent's lifetime the genome plays two roles. It must create the memome and during reproductive events it is used in recombination. Details of our agent's genetics can be found in (Marriott and Chebib, 2015).

If an agent has a surplus of energy and is prepared to breed then it must spend time breeding during the day. It does this at a particular site at a particular time of day. If there is another agent also performing the breed action at the same site at the same time then sexual reproduction occurs. If not the agent must wait for its next opportunity to breed.

Upon birth the genome copies short segments of itself into the memome. Each segment represents a path through the network long enough to be completed in a single day and starting at a specific site. We call these segments memeplexes and we copy every possible memeplex from the genome into the memome during initial development. We consider this technique is similar to the MAP-elites Mouret and Clune (2015) strategy for multi-objective optimization. We want to find the best memeplex given a particular starting site so we store the best memeplex for each starting site.

Memome

The memome is a collection of memeplexes. During behavior selection an agent selects the best memeplex given the current site. First an agent gathers the memeplexes that start at this site. The agent then selects the memeplex with the highest expected resource reward at the lowest energy cost. This is the agent's behavior for the day.

tion. The genome is inert and is only used to build the initial network. The weights of the network are stored independently (from the genome) and can be seen as a second information store. This store changes over the lifetime of the agent and is active in selecting behavior (i.e. in generating the phenotype that is relevant to selection).

Despite properly modeling the relationship between genetic and cultural information these models suffer from drawbacks due to the choice of artificial neural networks. Artificial neural networks still represent a very simple model of biological neural networks. Evolving ANN weights and topology is very challenging despite new advances like the NEAT algorithm (Stanley and Miikkulainen, 2002). Transferring information from one ANN to another using supervised training is slow, unreliable, prone to errors, and artificially requires training sessions. We pick up where these experiments leave off by clearly describing the model and presenting a different implementation choice that is easier to work with than neural networks.

The Dual Inheritance Model

The dual inheritance model exploits the advantages of the evolved social learning neural networks. Genetic and cultural information are stored separately and perform separate roles. We pivot from neural networks and embrace evolutionary processes. That is, we model phylogenetic, ontogentic, and sociogenetic adaptation as populations of individuals competing for fitness.

At birth a new agent inherits its genome from its parents

504

The memome is not inert over the lifetime of an agent. A newborn agent has memeplexes that are directly copied from the memome. Over time new memeplexes are added to the memome through individual learning and social learning.

Individual learning in our model occurs only if the agent spends time engaged in learning during the day. This requires the agent to select a memeplex for the day in which the agent spends time learning at at least one site. If it does so then the memeplex will add a possibly mutated copy of itself to the memome. This allows the agent to, among other things, optimize its foraging strategies.

Individual learning is a process in which the memome can improve itself via interaction with the environment. This is a developmental process that is analogous to the one from the evo-devo model above. In our experiments agents with only individual learning can improve their behavior with this mechanism but these improvements are lost when the agent dies.

Social learning is the important additional feature our agents need to achieve cumulative cultural evolution. In order for an optimized memeplex to survive the death of its host it must be shared with another host. Social learning in our implementation is similar to breeding. For two agents to learn from one another they must both perform the social learning action in the same site at the same time during the day. If they do they exchange a possibly mutated copy of the memeplex that they used that day. We treat this exchange as roughly approximating the agents telling each other what they did that day.

Individual learning is the power that allows the agent to optimize its behavior. This optimization is lost if it is not shared. Social learning is the power that allows a population to share optimized behaviors. The collection of all agents' memeplexes is called the memosphere.

A memeplex's evolutionary goal is to maintain copies of itself in the memosphere. To do this it must be optimized and so this usually means it spends time learning. Further more it is critical that it spend time social learning so that it can spread itself to other agents. Note that a memeplex is not concerned with spending time breeding. Breeding does not help the memeplex spread or optimize. This will be a source of divergence.

Divergence

We have observed divergence of trajectories in both our prior work and in the model described here. There are two types of divergence that we have observed. First both the genome and the memeplexes are trying to optimize themselves. The memeplexes have many opportunities to improve while the genome only has an opportunity when it reproduces. This means that the memeplexes will optimize more rapidly than the genome. Both trajectories are in the same direction but one gets there much faster. We call this divergence under cooperative selection pressures. Fig. 7 summarizes re-
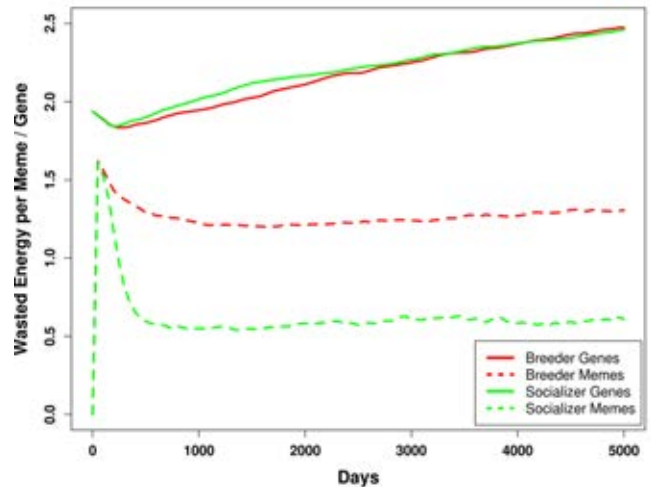

Figure 7: Contrasting the average genetic optimization and average memeplex optimization over time for two population of agents. Breeders are a control group using the evo-devo model. Socializers follow the dual inheritance model.

sults from our current implementation (Marriott and Chebib, 2016b).

The second type of divergence occurs when the selection pressures are competitive. We mentioned above that breeding does not help the memeplex. In fact it hinders it as time is better spent foraging than breeding and the memeplex needs to be as optimal as possible. Selection pressure against breeding in the memosphere is strong and we see many optimized memeplexes that spend no time breeding. Contrary to this selection pressure for breeding is very strong in the genome. Most agents have genomes that allocate a lot of time to breeding. This is a case of divergence under competitive selection pressures.

This divergence has an interesting effect. Memeplexes that spend no time breeding will suppress breeding in the agents that select them. This can create a culture of celibacy. If this culture becomes dominant it runs the risk of wiping out the population. We do indeed observe this in our current implementation. 21 out of 100 runs ended with complete population collapse before 5000 days. This was not observed in our prior work. In (Marriott and Chebib, 2014) agents always have the chance to reproduce and so colony collapse was not possible for this reason.

This divergence also occurs in both implementations relative to learning and social learning. We observe that there is strong selection pressure for learning and social learning in the genome. This is due to the benefits these adaptive mechanisms grant the agent from an evolutionary perspective. While there is selection pressure for learning and social learning in the memosphere there is also selection pressure to optimize these processes as much as possible. This means that once learning no longer pays off it is also com-
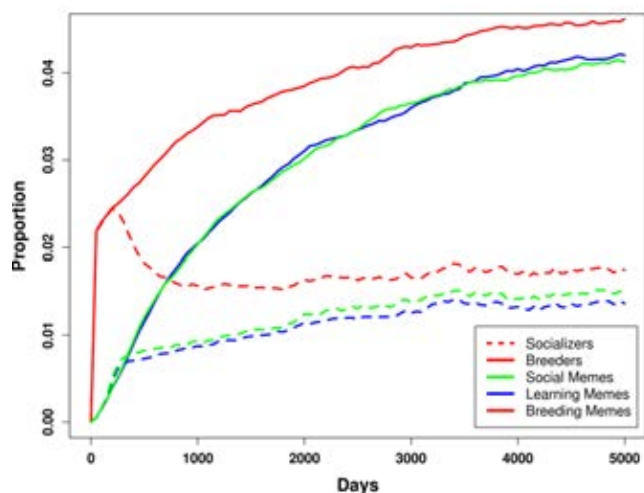
Figure 8: Contrasting the average time spent breeding, learning and socializing between two populations of agents. Breeders are a control group using the evo-devo model. Socializers follow the dual inheritance model.

monly eliminated. Social learning is commonly minimized as much as possible to ensure a very optimal memeplex that can still spread itself. Fig. 8 summarizes results from our current implementation (Marriott and Chebib, 2016b).

This divergence leads to a number of phenomena we wish to explore in more detail. Young agents have behaviors dictated largely by their genome. This means they spend a lot of time breeding, learning, and social learning. In a mature culture at some point the young agent will learn optimized memeplexes from others and alter its behavior. It will often no longer spend time breeding or learning. In any case it will spend as little time as possible breeding, learning and social learning. When it engages in social learning it usually does not get a memeplex better than its current ones, instead it is sharing its collection of memeplexes with others.

This causes a phenomena where young agents breed, learn and social learn more often than older agents. We observe that this is the case also in humans. Young humans engage in a considerable amount of learning and social learning. Young adults are more likely to have children than older adults. We hope to explore if this is an artifact of our implementations or a feature of other dual inheritance models.

Discussion

We have incorporated the benefits of all of the discussed models into our own with an attempt to capture a model that includes both genetic and memetic co-evolution as accurately as possible. Our model has the vertical transfer of the biological model as well as the horizontal transfer of the cultural model while avoiding the drawbacks of a single information storage that previous models suffer.

The key benefit of additional information storage is that

the two stores can diverge. This idea is already present in the evo-devo model. In this model the information of the genotype is passed on through reproduction while the information in the phenotype is used for selection. The benefit of this model (in both biology and simulation) is that the information in the phenotype can adapt over the lifetime to benefit passing on the information in the genotype of an agent.

This has two-fold benefit. First, the phenotype is free to adapt to any circumstance facing the agent during its lifetime. Second, the genotype is not disturbed in this adaptation and can be passed on intact to the next generation.

We can consider this benefit as a divergence of adaptive trajectories between the genotype and the phenotype. The genotype can still focus on replicating itself while the phenotype can focus on keeping the agent alive.

One advantage to the genotype in this arrangement is that the phenotype has a limited lifespan. After the agent dies the adaptations in the phenotype are lost and cannot upset the genotype's goal of replication. In this case, the phenotype is subordinate to the genotype and can only adapt within the boundaries dictated by the genotype. While these information stores can diverge from one another we say the phenotype in these models is *tethered* to the genotype. It can diverge but no further than the genotype allows (Marriott and Chebib, 2014).

Our model adds a new layer of information: the memome. The memome is also distinct from the genome and so can evolve on its own trajectory. Thanks to social learning it can avoid the death sentence of the phenotype. When the agent dies, if it has spread its cultural information to another agent, its memes can still live on. Unlike the information in the phenotype that exists tethered to the genotype, the memotype is free to evolve along its own trajectory.

The information in the memome is not completely free of the genome. In human culture, and in our simulations, the existence of memes is still dependent upon the existence of the agents that house them. These agents are biological and thus if the memome diverges so far as to endanger the genome it may endanger itself as well. So when we discuss divergence we do not mean a complete decoupling of genetic and cultural systems, but rather a very long leash. Cultures are able to destroy their host but in so doing they destroy themselves as well.

It is worth noting that our model does not expand on the implicit model of evolved social learning neural networks mentioned above. The implicit model in these experiments is identical to our model. One improvement we have made in terms of implementation is to make better choices of underlying structures and mechanisms. In particular, for simplicity, we have modeled the genome and memome as storing the same type of information. As a result our learning and social learning processes mimic the underlying genetic mechanisms of mutation and recombination. This makes modeling and implementing these processes much simpler

and we believe this contributes to the success of our agents.

Acknowledgments

References

Acerbi, A. and Parisi, D. (2006). Cultural transmission between and within generations. *Artificial Societies and Social Simulation*, 9.

Baldassarre, G. (2001). Cultural evolution of guiding criteria and behaviour in a population of neural-network agents. *Journal of Memetics*, 4(2).

Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7280–7287.

Borg, J. M., Channon, A., Day, C., et al. (2011). Discovering and maintaining behaviours inaccessible to incremental genetic evolution through transcription errors and cultural transmission. In *Advances in Artificial Life, ECAL 2011: Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems*, pages 101–108. MIT Press.

Boyd, R. and Richerson, P. J. (1983). The cultural transmission of acquired variation: Effects on genetic fitness. *Journal of Theoretical Biology*, 100(4):567–596.

Curran, D. and O'Riordan, C. (2007). The effects of cultural learning in populations of neural networks. *Artificial Life*, 13(1):45–67.

Dawkins, R. (1976). *The selfish gene*. Oxford University Press.

Dawkins, R. (1982). *The Extended Phenotype*. Oxford University Press.

Denaro, D. and Parisi, D. (1997). Cultrual evolution in a population of neural networks. In Marinaro, M. and Tagliaferri, R., editors, *Proceedings of neural nets wirn-96*, pages 100–111. Springer.

Dennett, D. C. (1995). Darwin's dangerous idea. *The Sciences*, 35(3):34–40.

Epstein, J. M. (2006). *Generative social science: Studies in agent-based computational modeling*. Princeton University Press.

Gabora, L. (1995). Meme and variations: A computational model of cultural evolution. *1993 Lectures in complex systems*, pages 471–485.

Gonzalez, M., Watson, R., Noble, J., and Bullock, S. (2014). The origin of culture: Selective conditions for horizontal information transfer. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 408–414.

Hall, B. K. (2012). *Evolutionary developmental biology*. Springer Science & Business Media.

Henrich, J. and McElreath, R. (2003). The evolution of cultural evolution. *Evolutionary Anthropology: Issues, News, and Reviews*, 12(3):123–135.

Hinton, G. E. and Nowlan, S. J. (1987). How learning can guide evolution. *Complex Systems*, 1:495–502.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press.

Lumsden, C. J. and Wilson, E. O. (1981). *The coevolutionary process*. World Scientific.

Marriott, C. and Chebib, J. (2014). The effect of social learning on individual learning and evolution. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, pages 736–743. MIT Press.

Marriott, C. and Chebib, J. (2015). Emergence-focused design in complex system simulation. In *European Conference on Artificial Life*. MIT Press.

Marriott, C. and Chebib, J. (2016a). Divergent cumulative cultural evolution. In *Proceedings of the 2016 Conference on the Synthesis and Simulation of Living Systems*. MIT Press.

Marriott, C. and Chebib, J. (2016b). Finding a mate with eusocial skills. In *Proceedings of the 2016 Conference on the Synthesis and Simulation of Living Systems*. MIT Press.

Moscato, P. et al. (1989). On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. *Caltech concurrent computation program, C3P Report*, 826:1989.

Mouret, J.-B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*.

Niazi, M. and Hussain, A. (2011). Agent-based computing from multi-agent systems to agent-based models: a visual survey. *Scientometrics*, 89(2):479–499.

Smaldino, P., Pickett, C., Sherman, J., and Schank, J. (2012). An agent-based model of social identity dynamics. *Journal of Artificial Societies and Social Simulation*, 15(4):7.

Smith, K. (2002). The cultural evolution of communication in a population of neural networks. *Connection Science*, 14(1):65–84.

Soltoggio, A., Bullinaria, J. A., Mattiussi, C., Durr, P., and Floreano, D. (2008). Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios. *Artificial Life*, (11).

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127.

Syvanen, M. (2012). Evolutionary implications of horizontal gene transfer. *Annual review of genetics*, 46:341–358.

Syvanen, M. and Kado, C. I. (2001). *Horizontal gene transfer*. Academic Press.

Sznajder, B., Sabelis, M. W., and Egas, M. (2012). How adaptive learning affects evolution: Reviewing theory on the baldwin effect. *Evolutionary Biology*, 39:301–310.

Tomko, N., Harvey, I., and Philippides, A. (2013). Unconstrain the population: The benefits of horizontal gene transfer in genetic algorithms. In *SmartData*, pages 117–127. Springer.

Divergent Cumulative Cultural Evolution

Chris Marriott[1] and Jobran Chebib[2]

[1]University of Washington, Tacoma, WA, USA 98402
[2]University of Zürich, Zürich, Switzerland 8057
dr.chris.marriott@gmail.com

Abstract

Divergent cumulative cultural evolution occurs when the cultural evolutionary trajectory diverges from the biological evolutionary trajectory. We consider the conditions under which divergent cumulative cultural evolution can occur. We hypothesize that two conditions are necessary. First that genetic and cultural information are stored separately in the agent. Second cultural information must be transferred horizontally between agents of different generations. We implement a model with these properties and show evidence of divergent cultural evolution under both cooperative and competitive selection pressures.

Introduction

Social learning is a form of learning that arises from social *situatedness* (Lindblom and Ziemke, 2003) and is characterized by agents interacting with one another in order to learn. Social learning can accelerate learning beyond that of individual learning strategies (see Marriott and Chebib (2014); Marriott et al. (2010)) and most notably is its ability to support cumulative cultural evolution (Whiten et al., 2011; Mesoudi et al., 2006; Henrich and McElreath, 2003; Boyd and Richerson, 1996). Cumulative cultural evolution is an adaptive process in which each generation can make improvements on the learned information inherited from their parents' generation (Dean et al., 2014; Kempe et al., 2014).

Genetic evolution and cultural evolution are parallel processes that optimize information in a population. It is common to consider the interaction between these parallel processes. Two effects have been well discussed with respect to learning: the *hiding effect* is when learning shields genetics from selection pressure, thus slowing the evolutionary process, and the Baldwin effect is when learning stimulates genetics, increasing particular selection pressures, and thus speeding up evolutionary adaptation (Sznajder et al., 2012; Paenke et al., 2006; Baldwin, 1896). These effects help describe the interaction between these parallel processes when they cooperate to improve fitness.

Another way we can compare genetic and cultural evolution is according to the direction of the evolutionary trajectory. It is possible for the cultural evolutionary trajectory to diverge from the biological evolutionary trajectory. In particular, this means that the culture may evolve in directions that are neutral or even detrimental to the biological imperatives of an agent or its genes. The biological imperatives of genes are merely survival and reproduction (Dennett, 1995; Dawkins, 1976). A divergent culture may be one that impedes an agents' abilities to survive and/or reproduce.

A simple non-human example of divergent evolutionary trajectories is sexual selection. Females could select for traits that correlate with fitness. In this case the culture and the genetic evolution agree. However, females could select for traits that do not correlate with fitness or correlate negatively with fitness. In certain birds of paradise the power of female selection has frustrated the fitness of males. Technically, this is not a case of cultural divergence since in sexual selection divergence occurs between two genetic evolutionary trajectories (male and female) within a single species.

Some human cultures may frustrate the reproductive or survival capabilities of some of their members (usually for the apparent benefit of the culture). For instance, a Catholic priest will abstain from reproducing according to the rule of his culture. Also, a Samurai might kill himself out of shame for failing to meet his cultural obligations. These cultural practices impede the genetic imperative of the genes in these individuals.

An extreme and rare case of cultural divergence would be a case where every individual of the culture engages in detrimental cultural practices. This would include cases of mass abstinence or mass suicide. Mass abstinence was a cultural belief of the Shakers (in the 1770s-1780s). Some mass suicides are caused for fear of capture by an enemy culture (as in Masada, Israel in 74 CE or in Pilenai, Lithuania in 1336 CE). Others are caused by a cultural belief that the suicide will grant reward in the afterlife (as with the Heaven's Gate mass suicide in 1997) or avoid punishment in this life (as in Jonestown in 1978).

We believe divergent cultural evolution requires at least two properties. First, genetic and cultural information must be stored in separate information stores. This rules out models with horizontal transfer of genetic material. Second, hor-

izontal transfer of cultural information occurs between individuals of the same generation or across generations. This property rules out evolutionary development models where phenotypic information is not transferred between individuals.

We believe these are necessary conditions and we believe they are probably not sufficient conditions. It is difficult to test this hypothesis since our implementation has many other implicit conditions that may play an important role. Our experiment is designed to test whether these conditions (plus implicit others) can lead to divergent cultural evolution in our population.

In Marriott and Chebib (2014) we demonstrated a simple instance of divergent genetic and cultural evolution in a population with these properties. The experiment involved a simple optimization problem, asexual agents, and no spatial environment. Agents in that experiment showed accelerated optimization and divergence of selection pressures for particular genes and memes.

We have reproduced this experiment in a virtual environment representing real space. Our agents engage in sexual reproduction and are subject to natural selection. Our first experiments in this environment involved simple agents with no learning capabilities (Marriott and Chebib, 2015a,b). We have augmented these agents with individual and social learning mechanisms.

Divergent Cultural Evolution

In Marriott and Chebib (2014) we implemented a simple proof of concept and demonstration of divergent cumulative cultural evolution. Agents in our model engage in all three modes of adaptation: phylogenetic, ontogenetic, and sociogenetic. Phylogenetic adaptation is adaptation by genetic evolution. Ontogenetic adaptation is lifetime adaptation. Sociogenetic adaptation is the result of exchanging learned material between agents.

The dual inheritance model (see Fig. 1) describes how these three modes of adaptation interact to create the agent (Marriott and Chebib, 2016b). Genetic information is inert over the lifetime of the agent in our model. It is transmitted vertically (from parent to child) during reproductive events and is responsible for the creation of initial cultural (memetic) information that is active during an agent's life. Memetic information is used during the lifetime of the agent to select a behavior for a particular situation and this information is also adaptive.

As agents are units of selection in our simulation natural selection occurs on the lifetime behavior of an agent (i.e. its phenotype). This behavior is determined by an interaction of an agent's genome, memome and environment. As a result both genetic and memetic information is important in determining if an agent lives and reproduces. In our model memetic selection also occurs. However, this selection is carried out by agents when they select what information to
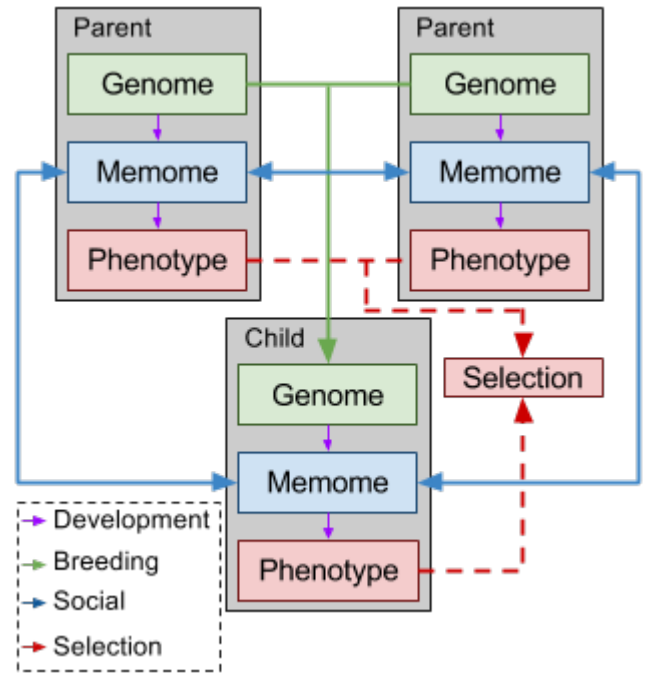

Figure 1: The dual inheritance model.

use, what information to share and whether or not to share their information.

There are two important ways that divergence can occur between genetic and cultural evolution. It is common that evolutionary trajectories in both the genetic and cultural realm are aligned. This is common when they are both trying to optimize a behavior. In these cases it is expected that cultural optimization of the behavior will outpace genetic optimization primarily due to the different the different timescales of these adaptive mechanisms. The only divergence here is in terms of the rate of optimization. We call this divergence under *cooperative selection pressures*.

The second type of divergence occurs when genetic selection pressures and cultural selection pressures are contrary. For instance, sexual reproduction is favored by genetic selection but suppressed in many (human and non-human) cultures. We call this divergence under *competitive selection pressures*.

We believe that both types of divergence require the properties stated above. That is, genetic and cultural information must be separate and cultural information must be transmitted horizontally. We will test our implementation for both types of divergence.

Model

We have improved on our proof of concept by placing our agents in an environment in which they compete for resources and are subjected to a form of natural selection (i.e.
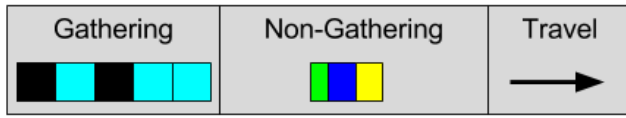
Figure 2: A typical gene consisting of gathering, non-gathering, and travel components.

compete for mates) instead of artificial selection (i.e. face a fitness function).

Our agents live in a random geometric network of resource sites (Penrose, 2003). Random geometric networks are an approximation of two dimensional physical space. At each site agents can spend time gathering the resources available at that site. Sites in our current model have one, two or three resources available to an agent that gathers at that site. Agents in our model have genetically or memetically encoded strategies for gathering at a site and the strategy determines the energy cost to the agent. The energy cost is always at least the number of resources available at that site (one, two or three) and at most five.

Agents have a simple metabolism in which resources are converted into energy. Energy is used to move around the environment, gather resources, and perform actions like breeding, learning and social learning. Additional small daily energy penalties are administered for idle activity, old age, and length of genome (only during reproduction). At the end of each day an agent has a net gain or loss of energy and this contributes to whether the agent lives or dies and whether it has enough energy to reproduce.

Genome

As mentioned above, in the dual inheritance model an agent's genome is inert during its lifetime and therefore is not adaptive nor directly active in behavior selection. The primary purpose of its genome is to spread genetic information in reproductive events. The secondary purpose of its genome is to produce an agent's memome upon its birth.

A genome of an agent represents a path of resources sites in the random geometric network. At each site on this path is also encoded possible behaviors for an agent at that site. That is, a genome represents a single long path through the network and the actions an agent might take at each site.

We call each site in this path a *gene* in the genome. A typical gene consists of three parts. A gathering component encodes a strategy for gathering resources at the respective site. A non-gathering component encodes the energy spent on non-gathering actions like breeding, learning and socializing. Energy in our model correlates with time (except in reproduction). In general, more time (energy) spent breeding, learning or socializing increases the likelihood of these activities being successful (more on this below). Finally, a gene has a travel component which encodes the energy cost

of traveling to the next site in the path.

Breeding in our model occurs by sexual reproduction so breeding is a social activity. In our environment two agents must be at the same site at the same time to breed. If both agents are performing the breed action for overlapping periods of time then a successful sexual reproduction occurs. We can see that more time spent breeding during the day will increase an agent's chances of reproducing. However, spending time breeding comes at an energy cost to the agent as well so it can't afford to spend all of its time breeding.

During sexual reproduction an offspring's genome is created as a recombination of its parents' genomes. Recombination uses the longest common subsequence of the two genomes. The offspring's genome also has an opportunity to mutate in this process (see Marriott and Chebib (2015b) for more details of genetic mechanisms). Only in these cases is a genome active during an agent's lifetime.

At birth each agent's genome creates a memome. A memome consists of a collection of memeplexes. Each memeplex in our model represents a possible set of activities for a single day. Memeplexes are subsequences of a genome. During memome generation we start at each gene in a genome and we copy gene by gene into a memeplex. This continues until the total energy of a segment approaches the maximum daily energy for an agent. If copying the next gene would exceed the maximum energy the segment is complete and the memeplex is stored. Memeplexes are stored in the memome along with other memeplexes starting with the same initial site for behavior selection (see below).

Additionally, segments are copied in a backwards direction from every gene. This means every gene in a genome is responsible for two memeplexes in its memome except the endpoints that are responsible for only a single memeplex. Notice that since every site in the environment is not necessarily represented in a genome there may be sites that do not have corresponding memeplexes.

Memome

Our agent's cognitive model is inspired by the pandemonium model (Jackson, 1987; Franklin, 1997; Marriott et al., 2010). Each memeplex is a sub-path of a genome and thus is a path in the random geometric network. A memeplex represents a single day's worth of activities. In the pandemonium model a memeplex is referred to as a daemon. Daemons compete for control of an agent and in our model memeplexes compete for control of an agent.

Behavior selection is also quite similar to the MAP-elites strategy for multi-objective evolutionary optimization (Mouret and Clune, 2015). We have memeplexes organized in the memome based on the starting site. An agent's day begins by selecting all memeplexes in its memome that begin at the agent's current site. Recall that these memeplexes represent a full day's worth of activities. The memeplex from this set that rewards the maximum (expected) resources for

the day while minimizing the energy expenditure is selected.

This is the primary force of cultural selection in our current implementation. It means that memeplexes with maximum resource-to-energy ratio are selected as behaviors. Since in our social learning mechanism agents that engage in social learning share only the current day's memeplex it means this selection mechanism is also used to select which memeplexes are shared during social learning.

An agent can engage in individual learning only if its selected memeplex includes at least one meme that has a non-zero learning component. This means that an agent must spend time engaged in learning at at least one site during a day. When this occurs an agent will clone its memeplex for the day and apply a mutation. The new memeplex is added to its memome. This allows an agent to possibly generate a memeplex that is more efficient at the same activities or generate an alternative sequence of activities.

We can see that having non-zero learning components in memes would benefit the agent. However, spending time learning during the day comes at an energy cost, as with breeding. Further, as our current implementation only allows a single learning event in a day it is not beneficial for an agent to spend more than the minimum amount of time learning.

An agent can engage in social learning once per day. The process for social learning is very similar to the process for sexual reproduction. Social learning can only occur if an agent spends time engaged in a social learning action at least one site during a day. However, for social learning to occur, another agent must also be at the same site at the same time engaged in social learning. If this occurs then the two agents will swap mutated copies of the memeplexes they used for that day. We treat this mechanism as roughly equivalent to telling the other agent what they did for the day.

Again we see a benefit to having social learning in memes as this will increase the chances of an exchange of memeplexes. However, as with learning and breeding, an agent cannot afford to spend too much time performing social learning during a day.

Both of these learning mechanisms allow for new memeplexes to be added to the memome which means an agent can adapt its behavior. When it begins its day at the same site again it may select one of the new memeplexes instead.

Experimental Setup

We conduct three similar experimental runs with agents of different capabilities. The first control group we call breeders, and while they still have memomes, their learning and social learning mechanisms have been turned off. In these agents, learning and social learning components of genes/memes are still present but inactive. The second control group we call learners. They are similar to breeders as they still lack social learning mechanisms, but they have their individual learning mechanisms intact. Like

the breeders, they still have social learning components of genes/memes but they are inactive. The third group is our experimental group. We call them socializers and they have all the functionality described above.

Each run is seeded with one hundred randomly generated agents. All genes in the randomly generated genome have learning and social learning components initialized to zero. Since it would be impossible for agents to breed if this were true of breeding components, we instead have a chance of initializing breeding components to non-zero values. As a result, agents must mutate learning and social learning in order to take advantage of these mechanisms. We allow our simulations to run for 5000 days. Under these settings every initial population is viable although some of our socializer simulations terminate early due to a catastrophic colony collapse (see below).

We gather data on many aspects of our agents' lives. In particular we gather information on the proportion of a genome or memeplex devoted to breeding, learning or social learning. Recall that genomes and memeplexes are both paths of sites and they can vary in length from agent to agent. We can measure the length of a genome or memeplex in two ways. We can count the number of genes/memes (representing sites) or we can measure the energy cost of a genome/memeplex as a whole. We record the proportion of a genome/memeplex devoted to breeding as the total energy devoted to breeding in a genome/memeplex over the total energy cost of the genome/memeplex. We do the same for learning and social learning.

We use a slightly different method to calculate the optimization of a genome/memeplex. For each site there is a gathering action. Recall the action will cost energy at least the number of resources rewarded at a site (one, two or three) and at most five. We can count all energy used above the minimum as wasted energy. For each genome/memeplex we compute the average wasted energy per gene/meme.

The control groups of breeders and learners should not display divergent cumulative cultural evolution. Among the control groups we expect differences in genome and memeplex measurements but we expect these differences to be small. In the socializers we expect to see evidence of divergent cumulative cultural evolution. We expect memeplexes will show evidence of greater optimization than the genomes, at least after enough time for cumulative cultural evolution to emerge. Recall this is a case of divergence under cooperative selection pressures.

We also expect to see divergence under competitive selection pressures (i.e. when they pull in different directions). We expect that genetic selection will select for breeding while also having indirect selection for learning and social learning. We expect that there will remain some selection pressure for social learning and learning in the memeplexes, but the pressure to optimize these actions out of a memeplex will also be strong. Breeding is not to the advantage of the
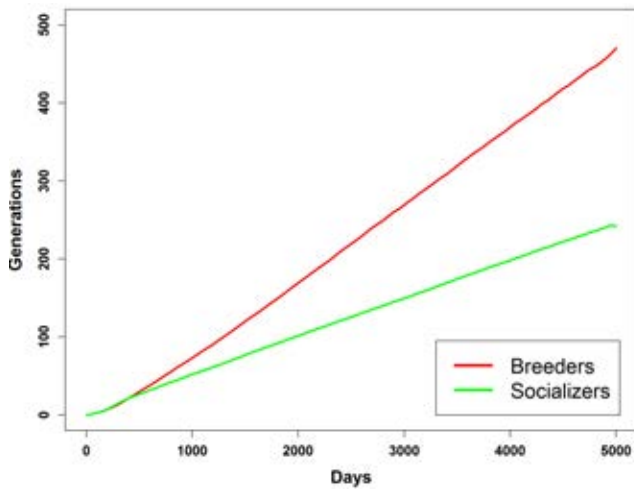
Figure 3: Generation over time for breeders and socializers. Generation is defined as the maximum generation in the population. A child's generation is one greater than the max of its parents' generations.

memetic selection mechanism so we expect it to be selected against by memetic selection.

Observations and Discussion

Our experiment was replicated 130 times on a variety of random geometric networks. All data presented in this section is averaged over these 130 runs.

We wish to begin with a discussion of an apparent slow-down of genetic evolution caused by cultural evolution. When we first investigated the breeding, learning and social learning genes in the genomes of socializers we found that gene concentrations for these components grow at the same rate for our two control groups but at a slower rate for the socializers. We thought this could be due to a hiding effect occurring between cultural evolution and genetic evolution.

We saw a similar effect on genome length over time and other data we gathered. However with further investigation we were able to determine the source of the slowdown. In both control groups the generation of agents increased at identical rates. The generations of socializers increased at about half the rate (Figure 3). This is due to an emergence of eusocial breeding culture in our agents (Marriott and Chebib, 2016b).

We think this evidence suggests that cultural evolution can slow genetic evolution over time, but possibly not over generations. That is, the shielding of genetic selection pressure is not really there. Instead there is a selection pressure for longer generations which has the result of slowing genetic evolution over time.

To confirm this we plotted gene concentrations over generations instead of days (Figure 4). We can see that the socializers actually have weak acceleration of evolution over
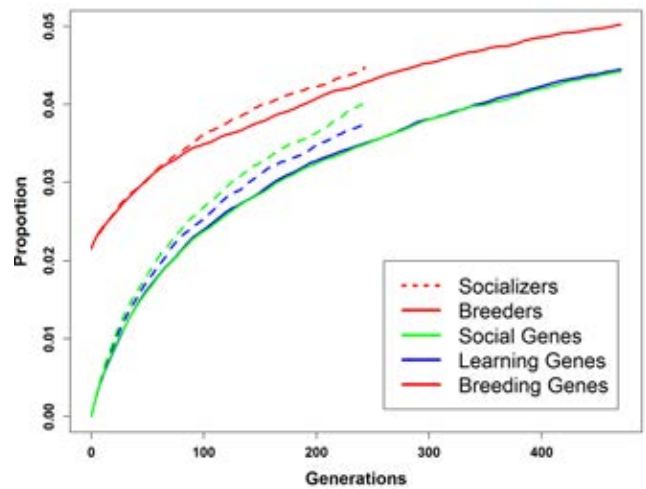

Figure 4: Concentration of breeding, learning, and socializing genes in the genomes of breeders (dotted) and socializers (solid) over generations.
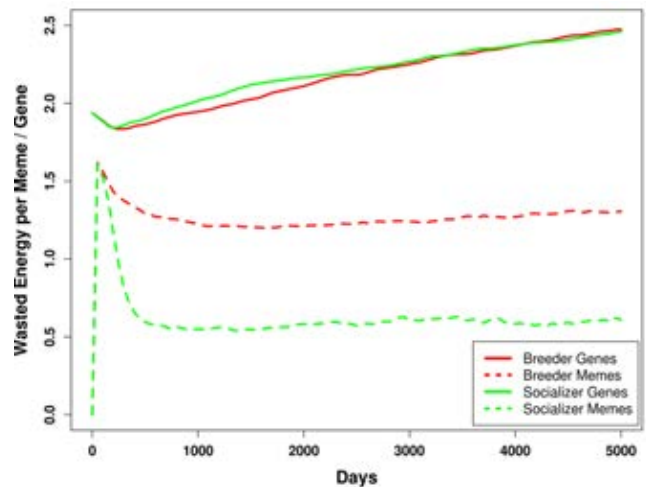

Figure 5: Average wasted energy per gene/meme in the genomes and memomes of breeders and socializers.

time according to this plot. These differences are small. We suspect that a greater significant difference might occur if we allowed our simulation to run for more days.

Now we wish to consider evidence for divergence under cooperative selection pressures. So we turn to a discussion of the relative optimization of genomes and memeplexes. Breeders and socializers showed a slight trend to less optimized genomes over time (Figure 5). This is not unexpected as much of the genome is not actually used during the agent's lifetime and thus is not subjected to selection at all. Further the genomes of agents increase overtime so this increases the size of the unused region. This has a result of stagnation of optimization in the genome except for a very small region. We can see in the control groups that this re-
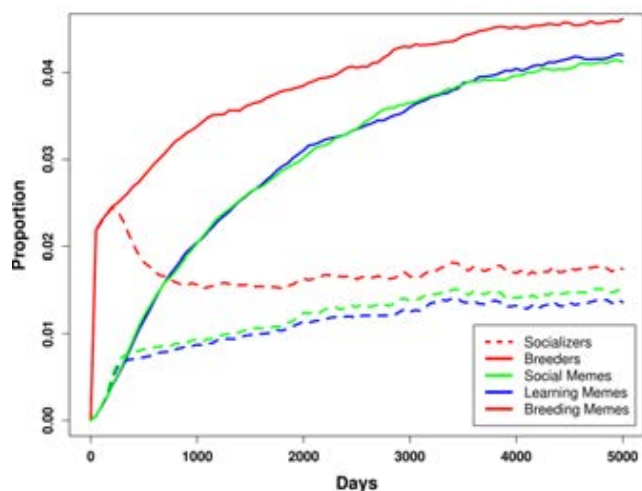
Figure 6: Concentration of breeding, learning, and socializing memes in the memomes of breeders (dotted) and socializers (solid).

gion is indeed optimized. We see this in the measurement of optimization of the daily memeplexes. The memeplexes selected for activity are more optimized than the genome as a whole. The memeplexes in the control runs undergo an early stage of optimization before stagnating.

In the socializers there is also an early stage of optimization before stagnation. However the stage of optimization is considerably greater in socializers than non-socializers. Stagnation in the control runs is in part due to a weak genetic selection pressure for optimization. Selection pressure for optimization in the memome is stronger and most evolved memeplexes are highly optimized (nearing zero wasted energy). However the data shows an average of all agents in the population. Only older agents have the evolved memeplexes and the younger agents have memeplexes that are still just close copies of regions of their genomes. When we average over all agents we will never optimize to zero.

Now let's consider divergence under competitive selection (see Fig 6). The strongest competitive pressure is relative to breeding actions. Treating memeplexes as a type of parasitic organism we can see that they are only concerned with their replication into new hosts. They are not concerned with their host's reproduction, even if this leads to fewer available hosts in the long run. A common end for this type of parasite is to die off after killing all available hosts.

We observe that soon after social learning emerges in the population memeplexes diverge from the genome. Breeding actions in a genome continue to be selected for, but breeding actions in a memome are selected against and their concentration decreases before stagnating. Notice again this stagnation is due in part to the averaging over the population. Some agents still are young and have a higher concentration of breeding actions than more evolved memeplexes.

We observe that in many socializers, breeding actions in their memeplexes are reduced to zero. This is also clear by noting that about 54% of breeders and learners have children while only 36% of socializers do. Memeplexes co-opt the agent for their reproductive ends instead of the genome's reproductive ends.

Interesting cases of collapsed colonies occur when these memeplexes spread to all agents before they can breed. In many runs of socializers we witness isolated colonies completely dying out. Out of 100 runs 21 ended when all agents died out before the 5000 day limit is reached. This only occurs in socializers.

In breeders and learners, colonies can face extinction due to a shortage of resources caused by overpopulation. In these situations agents can't get enough resources to reproduce and in some cases can't get enough to survive. However this situation cures itself as agents die out. As agents die, they no longer collect resources. The resources can instead go to the young and eventually young agents can reproduce. This causes cycles in population density but never a collapse.

Consider learning and social learning actions. There is strong early selection pressure in genomes and memomes for these actions. However as social learning kicks in, memetic selection against wasting time on these action takes over. Remember it is beneficial to spend as little time as possible on learning and so optimized memeplexes spend energy learning at only a single site. Many optimized memeplexes do not spend energy on learning at all. This is a detriment but if a memeplex is already highly optimized there is little benefit to learning. Further if a memeplex still spends time social learning an agent can still be adaptive.

It is also beneficial to not waste time on social learning. As above when social learning begins optimization places a negative selection pressure on social learning actions. However it is a memeplex's responsibility to spread itself. A memeplex that evolves to spend no time on social learning will never be spread (but may still be useful to an agent). Thus, we usually observe at least one site in a memeplex with social learning time and often more. This means that a memeplex usually has a particular site at which it spreads itself to other agents. This builds cultures of agents around meme spreading sites.

We notice for all breeding, learning and social learning actions as well as for optimization, cumulative cultural evolution causes a divergent effect from genetic evolution. In our simulations, this period of time occurred early on (in the first 500 days) and then social learning maintains an optimized culture through spreading evolved memeplexes to new agents.

To track cumulative cultural evolution we also assigned a generation to each memeplex. All original memeplexes created at birth are assigned generation zero. Whenever it is cloned in learning or social learning the clone is assigned generation one higher than its parent.
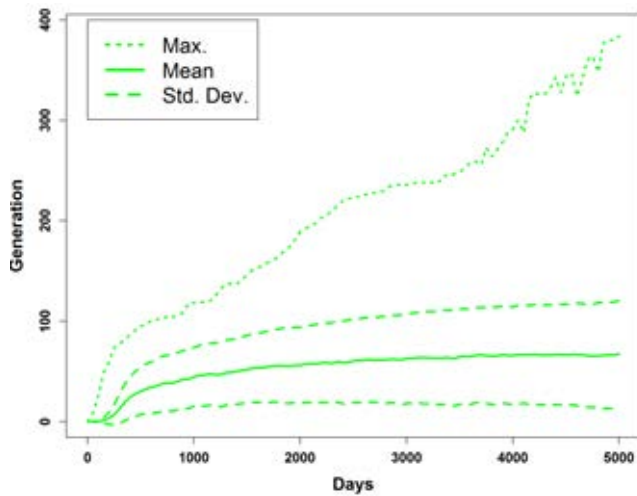
Figure 7: Average and maximum generation of active memeplexes over time with one standard deviation around the mean.

We can detect cumulative cultural evolution by detecting an increase in memeplex generation over time, especially from generation to generation (Figure 7). We can indeed confirm cumulative cultural evolution in our socializers by this method. Of course breeders can only have a memeplex generation of zero. Learners however can have a memeplex generation above zero if they learn better memeplexes. However these memeplexes can never leave their initial host and die with the host. Therefore, no cumulative culture can accrue (Kempe et al., 2014). We do see that learning agents can maintain a low non-zero memeplex generation but not one that increases over time.

Interestingly, in the socializers we do notice two stages in memeplex generation growth. The initial stage is more rapid and the second stage grows slowly at a fixed rate over time. We note that the period of rapid growth coincides with the period in which the memeplexes are being optimized prior to stagnation. The slower growth rate coincides with the stage where agents pass around optimized memeplexes.

Conclusions

Our implementation incorporated two information stores, one for genetic information and one for cultural information. It also had mechanisms of horizontal transfer of cultural information between agents of multiple generations. Our implementation is an example of our dual inheritance model of cultural evolution (Marriott and Chebib, 2016a). Our implementation demonstrates divergent cumulative cultural evolution under both conditions of cooperative and competitive selection pressures.

We can see that populations of agents that participate in the dual inheritance model can accelerate optimization relative to selection pressures that are cooperative between the genetic and memetic world. This is due to the potential for many cultural generations in a single biological generation. Thus, optimization can occur much faster over real time in cultural evolution than in biological evolution. Further, since the selection pressure on both the genome and the memome operate in the same direction there cannot be conflict between these pressures. The only divergence in this case is in terms of speed to convergence.

We also see cases where the selection pressure on the genome operates in an opposite direction for the memome. Genes and memes care only for spreading themselves. For genes, spreading occurs through reproductive events but for memes spreading occurs through social learning events. So it is not surprising that genes that increase the success of reproductive events are selected for by biological evolution and memes that increase the number and success of social learning events are selected for by cultural selection.

The contrary is not necessarily true. Genes that increase the number and success of social learning events may be selected for by biological evolution if social learning also helps improve survival or reproductive success. This is true in our simulation. Memes that increase the number and success of reproductive events have only a distant and indirect impact on the number and success of social learning actions. Since they also have a detrimental effect on the optimization of the memeplex there is a considerably stronger selection to avoid these actions.

Finally, we still see an interesting divergence in behavior of young (inexperienced) agents and old (experienced) agents. Young agents have had very little or no time to adapt their initial set of memeplexes either through individual optimization or through learning from others. Thus their behavior is still largely determined by their genome, which may also be the case in humans (Tomasello, 2016; Csibra and Gergely, 2011). This means they are more likely to breed, learn and social learn than old agents because all of these actions occur in much higher concentrations in the genome than in the memome.

The interesting impact of these trends is that young agents are more likely to be parents (i.e. before they learn better). They are more likely to learn from the environment more than older experienced agents. Finally, they are also more social. They are more likely to seek out social learning events than old agents. We find this conclusion interesting for two reasons. First, we see an emergent organization in our populations around age. Second this organization mimics the same organizations in other models and natural populations (Lehmann et al., 2013; Thornton and Malapert, 2009). Young humans are more likely to have children, more likely to attempt to improve themselves through learning, and more likely to seek out the knowledge of others than their older counterparts (Demps et al., 2012; Hewlett et al., 2011). Further older humans that engage in social learning are more often teachers than learners and this is also born

out by our experiment.

Finally we believe that the observed divergent cumulative cultural evolution is due to a critical component of the dual inheritance model. Specifically we think it is critical to keep genetic and cultural information separate from one another even if they store the same kinds of information (as in our implementation). Without separate information stores there is no environment for divergence to occur. Secondly it is important that cultural information can be transmitted between members of the same generation and between generations.

Acknowledgments

References

Baldwin, J. M. (1896). A new factor in evolution. *The american naturalist*, 30(354):441–451.

Boyd, R. and Richerson, P. J. (1996). Why culture is common, but cultural evolution is rare. In *Proceedings-British Academy*, volume 88, pages 77–94. OXFORD UNIVERSITY PRESS INC.

Csibra, G. and Gergely, G. (2011). Natural pedagogy as evolutionary adaptation. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 366(1567):1149–1157.

Dawkins, R. (1976). *The selfish gene*. Oxford University Press.

Dean, L. G., Vale, G. L., Laland, K. N., Flynn, E., and Kendal, R. L. (2014). Human cumulative culture: a comparative perspective. *Biological Reviews*, 89(2):284–301.

Demps, K., Zorondo-Rodríguez, F., García, C., and Reyes-García, V. (2012). Social learning across the life cycle: cultural knowledge acquisition for honey collection among the jenu kuruba, india. *Evolution and Human Behavior*, 33(5):460–470.

Dennett, D. C. (1995). Darwin's dangerous idea. *The Sciences*, 35(3):34–40.

Franklin, S. (1997). *Artificial minds*. MIT press.

Henrich, J. and McElreath, R. (2003). The evolution of cultural evolution. *Evolutionary Anthropology: Issues, News, and Reviews*, 12(3):123–135.

Hewlett, B. S., Fouts, H. N., Boyette, A. H., and Hewlett, B. L. (2011). Social learning among congo basin hunter–gatherers. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 366(1567):1168–1178.

Jackson, J. V. (1987). Idea for a mind. *ACM SIGART Bulletin*, (101):23–26.

Kempe, M., Lycett, S. J., and Mesoudi, A. (2014). From cultural traditions to cumulative culture: parameterizing the differences between human and nonhuman culture. *Journal of theoretical biology*, 359:29–36.

Lehmann, L., Wakano, J. Y., and Aoki, K. (2013). On optimal learning schedules and the marginal value of cumulative cultural evolution. *Evolution*, 67(5):1435–1445.

Lindblom, J. and Ziemke, T. (2003). Social situatedness of natural and artificial intelligence: Vygotsky and beyond. *Adaptive Behavior*, 11:79–96.

Marriott, C. and Chebib, J. (2014). The effect of social learning on individual learning and evolution. In *The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, pages 736–743. MIT Press.

Marriott, C. and Chebib, J. (2015a). Emergence-focused design in complex system simulation. In *European Conference on Artificial Life*. MIT Press.

Marriott, C. and Chebib, J. (2015b). Finding a mate with no social skills. In *Proceedings of the 2015 conference on Genetic and Evolutionary Computation*. ACM.

Marriott, C. and Chebib, J. (2016a). Finding a mate with eusocial skills. In *Proceedings of the 2016 Conference on the Synthesis and Simulation of Living Systems*. MIT Press.

Marriott, C. and Chebib, J. (2016b). Modelling the evolution of gene-culture divergence. In *Proceedings of the 2016 Conference on the Synthesis and Simulation of Living Systems*. MIT Press.

Marriott, C., Parker, J., and Denzinger, J. (2010). Imitation as a mechanism of cultural transmission. *Artificial Life*, 16:21–37.

Mesoudi, A., Whiten, A., and Laland, K. (2006). Towards a unified science of cultural evolution. *Behavioral and Brain Sciences*, 29:329–383.

Mouret, J.-B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*.

Paenke, I., Kawecki, T., and Sendhoff, B. (2006). On the influence of lifetime learning on selection pressure. In Rocha, L., Yaeger, L., Bedau, M., Floreano, D., Goldstone, R., and Vespignani, A., editors, *Artificial life X*. MIT Press.

Penrose, M. (2003). *Random geometric graphs*, volume 5. Oxford University Press Oxford.

Sznajder, B., Sabelis, M. W., and Egas, M. (2012). How adaptive learning affects evolution: Reviewing theory on the baldwin effect. *Evolutionary Biology*, 39:301–310.

Thornton, A. and Malapert, A. (2009). Experimental evidence for social transmission of food acquisition techniques in wild meerkats. *Animal Behaviour*, 78(2):255–264.

Tomasello, M. (2016). The ontogeny of cultural learning. *Current Opinion in Psychology*, 8:1–4.

Whiten, A., Hinde, R. A., Laland, K. N., and Stringer, C. B. (2011). Culture evolves. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 366(1567):938–948.

Computational Biology

"Shit Happens": The Spontaneous Self-Organisation of Communal Boundary Latrines via Stigmergy in a Null Model of the European Badger, *Meles meles*

Seth Bullock

Department of Computer Science, University of Bristol, UK
seth.bullock@bristol.ac.uk

Abstract

The ability of European badgers to establish communal latrines at their territory boundaries is a well-known but poorly understood example of group-level biological organisation. To what extent might we expect it to arise via self-organisation rather than as the result of specific adaptations? This paper replicates and extends a model of badger foraging and territoriality to include defecation, "fæcotaxis" and overmarking behaviours, and shows that communal boundary latrines arise spontaneously through stigmergy in both territorial *and* non-territorial badgers, with no need for specific cognitive or behavioural adaptations such as spatial memory, or individual recognition. The model suggests that fæcotaxis and overmarking behaviours are necessary for boundary latrine formation, that culling has little effect on the prevalence of fæcal sites (implicated in the spread of bovine tuberculosis in the UK), and that the spatial micro-structure of the environment is significant to the self-organisation process.

Introduction

One key Artificial Life research question is understanding the extent to which living systems result from self-organisation (Kauffman, 1993; Goodwin, 1994) or adaptation (Dawkins, 1986). Here we develop a simple, spatially extended model of species-environment self-organisation to better inform our expectations regarding the spatial patterns that analogous living systems are capable of.

Many animal and insect species establish and use communal toilets. From ants (Czaczkes et al., 2015) to lemurs (Irwin et al., 2004) and rhinoceroses (Freeman et al., 2014), there are many examples of organisms preferentially leaving fæces, urine and other waste in areas dedicated for this purpose. The function of these waste chambers, latrines and middens is not always limited to sanitation, but can involve defence (where latrines are used to mark territory boundaries) and communication (where scent marking at latrines is used to signal information about, e.g., mate quality).

The European badger, *Meles meles*, is the only British mammal that uses purpose-built toilets (other than humans). They establish communal latrines within their territories and at territory boundaries (Roper et al., 1993). Biologists believe that communal boundary latrines may play a communicative role, allowing each group of badgers to communicate with other nearby groups, sharing information on the status of each clan (e.g., Stewart et al., 2002).

While there have been studies of latrine use in badgers (e.g., Roper et al., 1986) and models of territoriality in badger populations (e.g., Stewart et al., 1997), there is little understanding of how communal latrines arise, how their spatial distribution is influenced by badger foraging behaviour and territoriality, the extent to which specific adaptations are required in order to establish and maintain them, and how their use and distribution might be affected by changes in population size. This last issue is particularly timely given recent attempts to control the spread of bovine tuberculosis from badger fæces to cattle in the UK through culling of the badger population (Indepedent Scientific Group on Cattle TB, 2007; Donnelly and Woodroffe, 2015).

The current paper presents a simple null model of communal latrine formation and uses it to establish how readily communal latrines arise in a population of simulated badgers, to identify which factors influence the spatial distribution of latrines, and to understand the effect of simulated culling. The model confirms that communal latrines can arise spontaneously when badgers exhibit a minimal form of fæcotaxis and overmarking behaviour, that they tend to self-organise along territory boundaries for territorial badgers, and points equidistant between setts for non-territorial, free-roaming badgers, and that culling is not, prima facie, an effective way to limit the distribution of badger fæces.

The paper goes on to explore the role of spatial embedding in facilitating communal latrine formation, demonstrating that shared boundary latrines can arise in a system that has no spatial structure within each territory. These results raise the possibility that the model might be used to shed light on territorial social media behaviour exhibited by communities of humans (Williams et al., 2015), where online activity at sites frequented by multiple communities may play a similar defensive or communicative function to that of deposits left by badgers and other mustelids, despite social media activity taking place in a non-Euclidean virtual space.

The model

The model of badger foraging and latrine formation that will be developed and presented here is an extension of a previous simulation model of badger foraging and territoriality (Stewart et al., 1997). This model did not include defecation or latrine formation, but was developed to explore the hypothesis that territoriality in *Meles meles* arises as a consequence of *Passive Range Exclusion* (PRE), i.e., badgers from adjacent clans tend to avoid each other's home range merely as a consequence of optimizing their foraging efficiency, since food gradients tend to peak at or around home range boundaries.

The PRE model represents the badgers' environment as an array of hexagonal *cells*, organised into seven close-packed, non-overlapping hexagonal *territories*. Each territory comprises a central cell containing a *sett* (the underground home of a clan of badgers), surrounded by r rings of cells (see Figure 1). (While the original PRE model did not allow badgers to move beyond the outer edges of the outer territories, in our replication we allow the environment to be a hexagonal torus in which the outer cells are adjacent to the appropriate cells on the opposite side of the environment.)

Each model run simulates n consecutive nocturnal foraging periods of T timesteps each. Initially, B badgers are allocated to each sett. All badgers in a sett have the same strategy. During each foraging period, each badger leaves its sett and moves around the environment consuming the food that it discovers. By the end of each foraging period, each clan of B badgers will have returned to their sett where they will remain inactive for T timesteps until the beginning of the next nocturnal foraging period. At the start of each foraging period food is redistributed throughout the environment. Every cell that is not a badger sett has a probability P_F of being allocated F units of food, and a probability $1 - P_F$ of being empty. At the end of each nocturnal foraging period any remaining food is removed from the environment.

During each foraging timestep, badgers that are not busy feeding move simultaneously to an adjacent cell (they may not enter the cell that they last occupied or any cell that contains a sett). When badgers move onto a cell containing food that is not already being eaten, they start to consume it, taking a handling time of h timesteps before they continue to forage in other cells. A badger moving onto a cell containing food that was already being eaten by another badger or badgers on the previous timestep ignores the food. If b badgers move simultaneously onto the same cell, and this cell contains food that is not already being eaten, the food is shared equally amongst the b badgers, with each badger receiving $\frac{F}{hb}$ units of food for each of h timesteps.

When the shortest distance from a badger to its home sett, d, is equal to the number of time steps remaining in the current nocturnal foraging period, it ceases foraging and proceeds to move in the direction of its home sett (even if this means leaving food uneaten or moving into a cell that it oc-
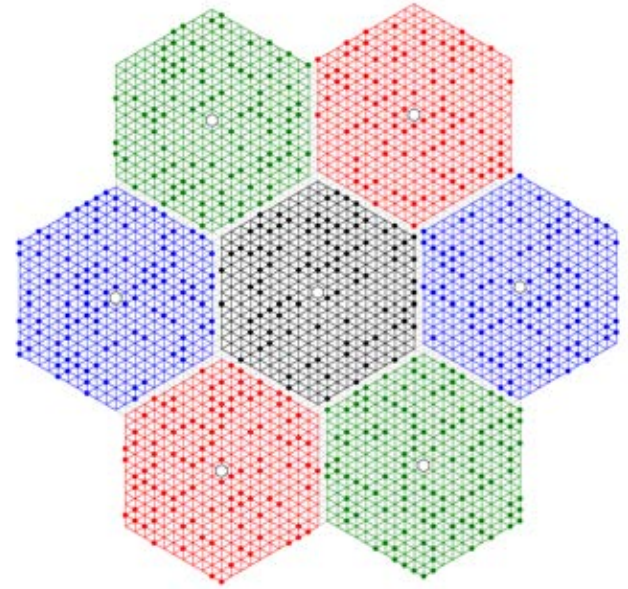

Figure 1: Seven badger territories, each comprising a central sett (open circle) surrounded by $r = 10$ rings of cells. Filled circles indicate cells containing food ($P_F = 0.3$). Grey edges represent transitions between territories.

cupied on the previous timestep).

The PRE model considered three badger strategies:

- **Free-Roamers** may move to any adjacent legal cell when foraging, ignoring considerations of territoriality.

- **Overlappers** must remain within a distance $d \leq r + w$ of their home sett, i.e., they may move up to w cells into an adjacent territory. Overlappers generalise the PRE model's **Territorial Badgers** where are Overlappers with $w = 0$.

- **Boundary-Walkers** are badgers that behave as Free-Roamers until they first reach a cell at the perimeter of their own territory (i.e., $d = r$), after which they must forage within a distance w of their territory's boundary (i.e., $r - w < d \leq r + w$) until they return home towards the end of the foraging period.

Extending PRE

Here, after replicating the key results of the PRE model (see Figure 2), we extend it in order to model latrine formation by including (i) *defecation*, (ii) a minimal form of "fæcotaxis" (taxis towards sites of recent defecation), and (iii) a simple form of *overmarking* behaviour (the tendency to defecate on top of previous fæcal deposits).

For each badger, we define the number of timesteps elapsed since the badger's last defecation, p. Badgers are initialised with $p = T$ to represent the period of inactivity

immediately before the first nocturnal foraging period, and the value increases by one during each foraging timestep, and by T during each period of daytime inactivity. Each badger's p counter is reset to zero when it defecates.

In order to implement a tendency for badgers to move towards nearby sites of defecation we implement a biased random walk, favouring adjacent locations with more fæcal deposits. We set the strength of this bias to increase linearly with p, the length of time since a badger's last defecation. Instead of moving a badger to a cell selected at *random* from the set of *all* legal adjacent cells, we select from a random *subset* of these cells the cell with most fæcal deposits from a ran, breaking any ties at random.

Where the set of legal cells adjacent to a badger is L, we consider a random subset of $\left\lfloor |L| \frac{\min(T,p)}{T+1} \right\rfloor + 1$ cells from L. Consequently, immediately after defecation a badger will move to a random legal adjacent cell (subset size is 1), whereas T or more time steps after defecation it will always move to the legal adjacent cell with the greatest number of fæcal deposits (subset size is $|L|$). Note that this fæcotaxis is minimal in that it only considers the cells immediately adjacent to a badger.

The probability, P_D, that during a particular timestep a badger will defecate is defined as a sigmoid function of p, the time since last defecation, and f, the number of fæcal deposits in the cell, parameterised by κ, the steepness of the sigmoid, θ, the position of the sigmoid mid-point, and α, a factor controlling the strength of the overmarking tendency, i.e., the influence of any fæcal deposits in the cell:

$$P_D = 1/(1 + e^{-kx}) \qquad \text{where } x = p - \theta + \alpha f$$

Simulations described here use: $\kappa = 0.25$, $\theta = 1 - (T/20)$, and $\alpha = T/20$ in order to achieve a relatively sharp transition from low to high probability near the end of the day.

At the start of each nocturnal foraging period, any fæcal deposits older than zT timesteps are removed from the environment. For the results reported here, $z = 5$, which is consistent with estimations of how long deposits can be considered "fresh" (Roper et al., 1993).

Results

Replication

The reimplementation confirmed that badger foraging behaviour tends to establish a food density gradient that peaks at or around the boundary between adjacent territories (Figure 2), ensuring that Boundary Walkers who spend most time in the boundary zone tend to forage more successfully and that this advantage increases with the initial density of food in the environment (Figure 3).

The food density gradient results from the fact there are only a small number of heavily foraged cells near to a sett compared to the larger number of cells at the periphery
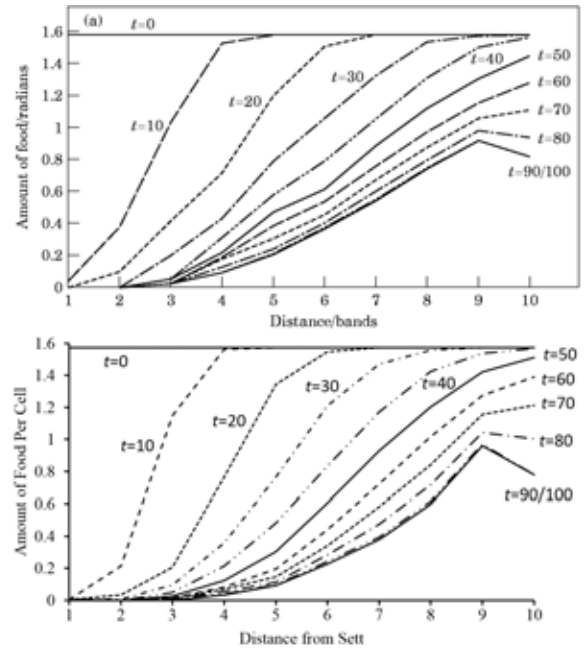

Figure 2: Results from the original PRE model (top) agree closely with those from the reimplementation (bottom). Both plots depict the establishment, by Boundary Walkers, of a food gradient in the central territory running from the central sett to the territory boundary. The dip in food density that arises at the boundary after 80 timesteps results from the boundary-restricted movement of Boundary Walkers. The peripheral territories contain Territorial badgers. Gradients are plotted at 10 timestep intervals with $r = 10$, $T = 100$, $P_F = 1.0$, $F = 1.575$, $B = 20$, $h = 4$, and $n = 10$ foraging periods. (While the meaning of original (top) y axis label is not clear, it is assumed to be a measure of food density along the lines of that used in the lower plot.)

which are foraged less readily. Since it takes time for badgers to reach the boundary zone, it tends to remain a richer source of food for longer during each foraging period.

However, Figure 4 demonstrates that overall foraging success is higher for Free-Roamers than for either Territorial badgers or Boundary Walkers. Badgers that have their Restricting movement tends to limit food intake because badgers more often re-enter cells they visited previously. A direct comparison of the efficiency of the different strategies was not reported by Stewart et al. (1997), but this result tends to undermine the passive range exclusion hypothesis which holds that badgers forage within a "territory" merely to achieve higher rates of foraging success.

Latrines

When the model is extended to include defecation, fæcotaxis and overmarking behaviours, badgers readily achieve communal latrines (defined here as those with > 10 fæcal
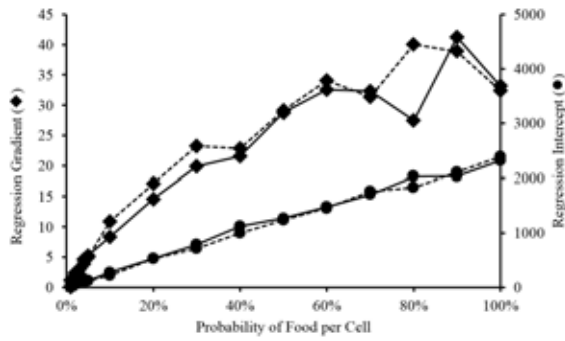
Figure 3: A comparison of the PRE model (dashed line) and the replication (solid line) in terms of the intercepts (circles) and gradients (diamonds) of linear models regressing the amount of food consumed by an Overlapper on the time it spent in the boundary zone for different values of P_F. Parameters as per Figure 2, save that $F = 200$ and $n = 25$.
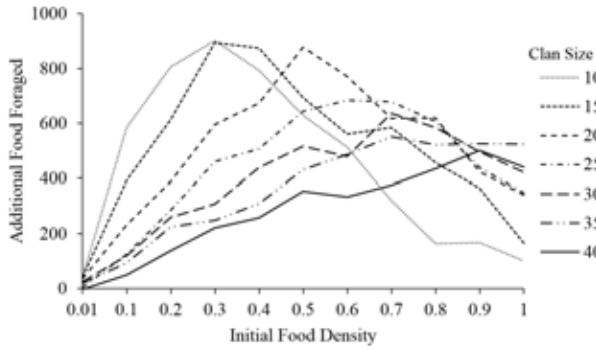

Figure 4: Free-Roamers collect more food than Boundary-Walkers regardless of both clan size and initial food density. Model parameters are: $r = 10$, $T = 100$, $F = 200$, $h = 4$, and $n = 25$ foraging periods. Standard errors (not shown) are ≈ 70 food units on average.

deposits at the end of the final foraging period). These latrines tend to appear at territory boundaries and also close to badger setts (Figure 5). Communal boundary latrines are achieved both by Overlappers (who may only forage up to w cells inside an adjacent territory) *and* by Free-Roamers (who are ignorant of any territory boundaries), although the communal latrines achieved by the latter strategy are not as tightly distributed at territory boundaries (Figure 6).

Communal latrines at the sett result from the tendency for badgers to defecate at the start of each foraging period, whereas communal boundary latrines result from a combination of (i) higher numbers of Overlapper or Free-Roamer badgers in the boundary zone, (ii) positive feedback resulting from fæcotaxis and overmarking, which encourages badgers to add to existing latrines rather than start new ones.

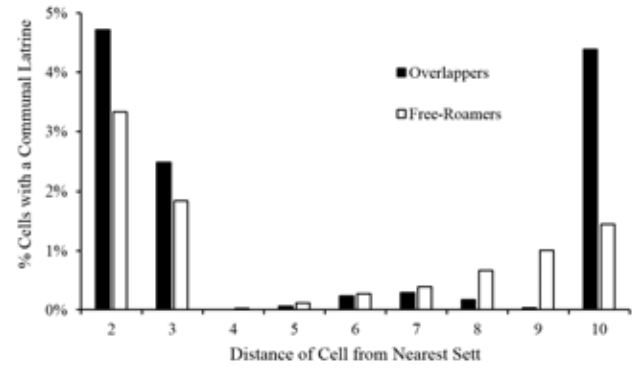Figure 7 demonstrates that cells foraged by two clans of

Figure 6: The proportion of cells containing communal latrines at differing distances from the nearest sett for Overlappers (filled bars) and Free-Roamers (open bars). Over 95% of cells immediately adjacent to setts contained communal latrines (not shown). Figures depict data from the final state of 25 runs with parameters as per Figure 5.
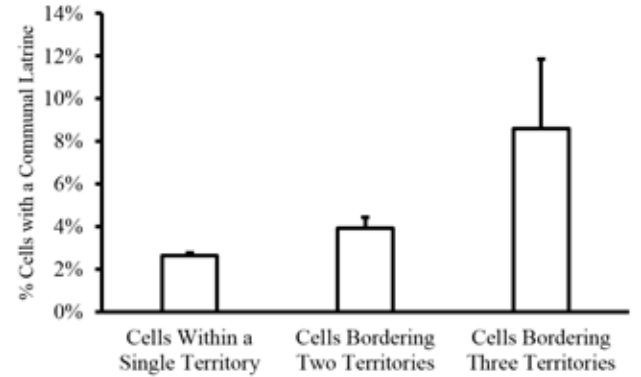

Figure 7: Simulation results from Figure 5a, replotted to depict the proportion of communal latrine cells within the hinterland of a single Overlapper territory, at the boundary between two territories, or at the intersection between three territories. Whiskers represent standard deviations.

Overlapper badgers (i.e., cells along a territory boundary) are more likely to be the site of a communal latrine than cells foraged by one clan (i.e., cells within a territory), and that cells at the intersection of three territories are the most likely to become communal latrines.

To what extent are communal boundary latrines dependent on the positive feedback brought about by fæcotaxis and overmarking behaviour? Might only one of these behaviours be necessary, or even neither of them? Figure 8 shows Overlapper latrine distributions that arise when each of these behaviours is suppressed. In neither case are communal boundary latrines achieved at a significant rate, indicating that both behaviours are crucial (Figure 9).
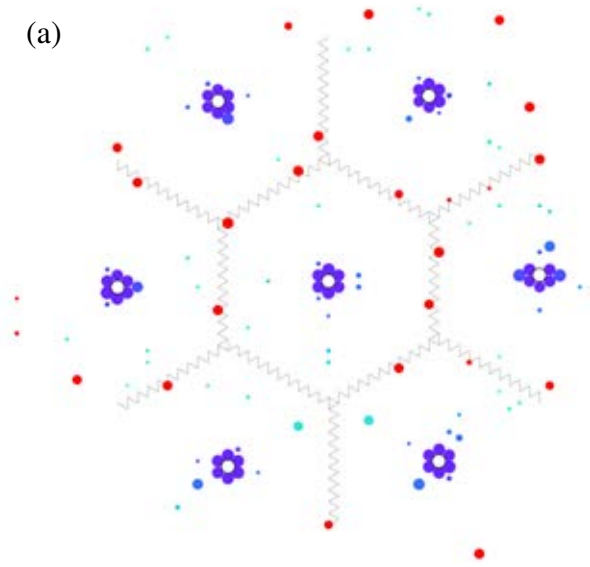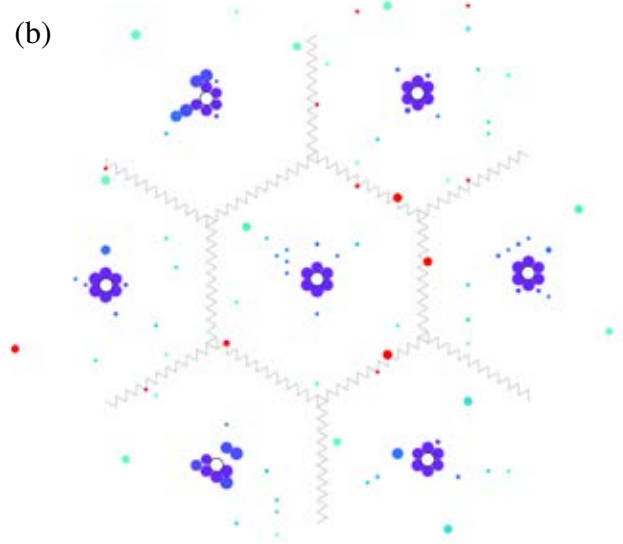
Figure 5: Representative latrine distributions after $n = 200$ nocturnal foraging periods for (a) Overlappers, and (b) Free-Roamers. Setts are represented by open circles. Latrines are represented by filled circles. Larger circles contain more fæcal deposits. Latrines closer to their nearest sett are represented by darker blue circles. Red circles represent latrines in the boundary zones between territories (which are toroidal at the periphery of the environment). Model parameters are: $r = 10$, $T = 100$, $F = 200$, $B = 20$, $h = 4$, $P_F = 0.3$ and $w = 1$.
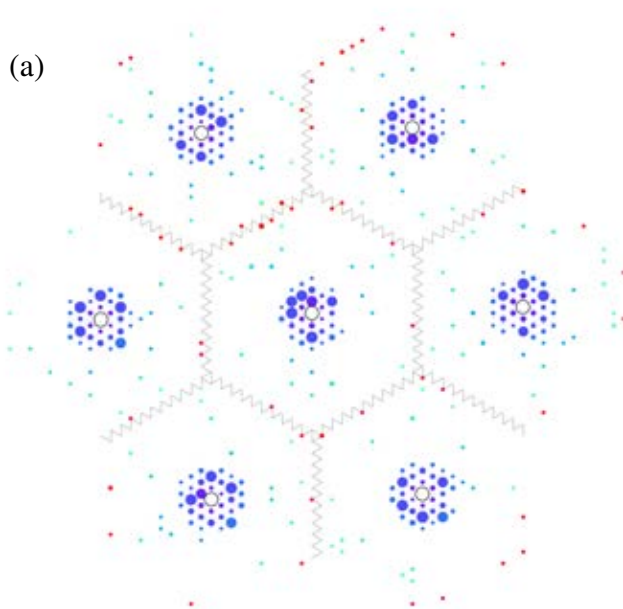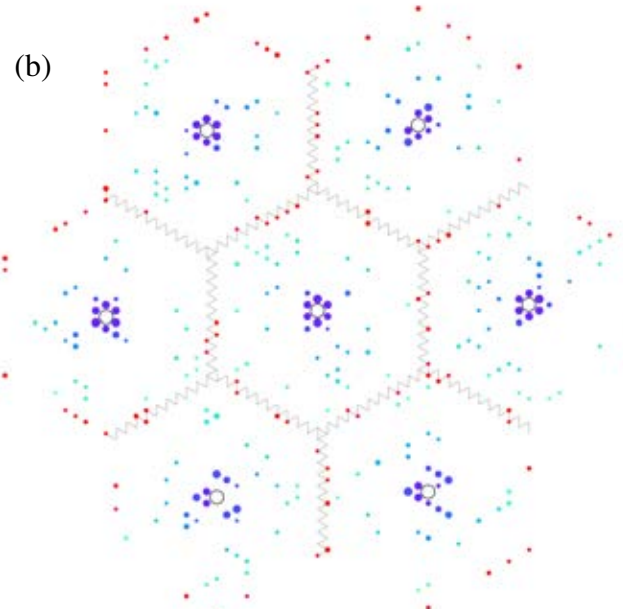

Figure 8: Representative latrine distributions after $n = 200$ nocturnal foraging periods for Overlappers with either (a) no fæcotaxis behaviour, or (b) no overmarking behaviour. Data visualization and model parameters are as per Figure 5.
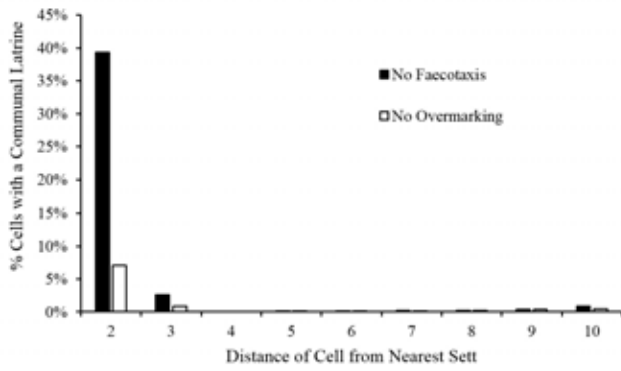
Figure 9: The proportion of cells containing communal latrines at differing distances from the nearest sett for Overlappers with either no fæcotaxis behaviour (filled bars) or no overmarking behaviour (open bars). Again over 95% of cells immediately adjacent to setts contained communal latrines (not shown). Figures depict data from the final state of 25 runs with parameters as per Figure 5.

Culling

One route by which badgers might cause bovine tuberculosis in cattle is through contact between cows and infected badger fæces. In the UK, there have been attempts to reduce bovine TB by culling badger populations (Indepedent Scientific Group on Cattle TB, 2007), although the policy has been controversial (Donnelly and Woodroffe, 2015). Figure 10 shows how the distribution of fæcal deposits is affected by changes in clan size. While reducing clan size reduces the average number of fæcal deposits at each site at which fæcal deposits are present, the number of such sites is relatively stable. This is because the stigmergic feedback that tends to encourage badgers to defecate in the same place becomes weaker as the number of badgers is decreased. Consequently, when badger populations are small, although the total volume of fæces present in the environment is reduced, it is distributed just as widely. Hence, in a simple model such as this one, culling is not an effective way to reduce the rate at which cattle encounter badger fæces.

Eroding Spatial Constraints

To what extent is the tendency for communal latrines to arise at territory boundaries dependent on the environment being spatially structured? In order to explore this question, we introduce a degree of "random rewiring" to the environment, eroding the spatial correlation structure within each territory and between each pair of adjacent territories. The rewiring scheme is as follows:

1. Start with a graph, G, representing the original environment structure in which every cell is represented by a node and every pair of adjacent cells is represented by an edge.
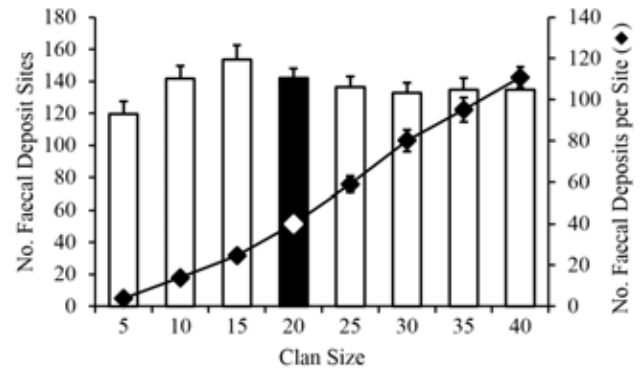

Figure 10: Influence of clan size on the distribution of fæcal deposit sites, i.e., cells containing at least one fæcal deposit at the end of a run. Bars represent the number of sites. Diamonds represent the mean number of deposits at each site. Data is averaged over 25 runs (whiskers represent standard deviations). Runs with the same parameters as Figure 5 are represented by the solid bar and empty diamond. Other bars/diamonds represent runs that differ only in terms of clan size. Reducing clan size lowers the average number of deposits at fæcal deposit sites, but reductions of as much as 75% fail to result in a significant reduction in the *number* of sites.

2. For each undirected edge, $ij \in G$, with probability P_R remove it from G and add i and j to a list L_{IJ} where I and J are the territories of i and j, respectively. (Note that a list may contain the same node more than once.)

3. Merge each pair of lists L_{IJ} and L_{JI}, where $I \neq J$.

4. For each list, until it is empty, repeatedly pick a random pair of nodes, i and j, from the list, ensuring that $i \neq j$ and $ij \notin G$, and add an undirected edge ij to G.

This rewiring scheme is constrained to ensure that, whilst it erodes the local spatial structure within and between territories, it does not change the overall gross structure of the environment and does not introduce new biases that favour particular sites. If an original edge linked two nodes within the same territory then so will the new randomly rewired edge. If an original edge linked nodes from two different territories, then the new randomly rewired edge will link nodes from the same pair of territories. Consequently, while the micro-structure within and between territories is eroded by rewiring, the gross territorial structure of the environment is maintained. This means that, amongst other things, the environment ceases to be a lattice with high clustering and long characteristic path lengths, and each sett is no longer equidistant from the boundary zones with adjacent territories. Moreover, the random rewiring process does not change the total number of neighbours that any node possesses (i.e., the network's degree distribution remains un-
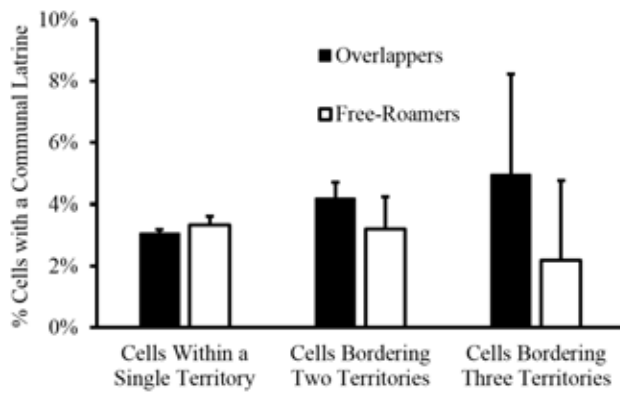
Figure 11: The proportion of communal latrine cells within a single territory's hinterland, at the boundary between two territories, or at the intersection between three territories for Overlappers and Free-Roamers in environments where spatial structure has been eroded by random rewiring ($P_R = 1.0$). Model parameters are as per Figure 5.

changed). This is important, since high-degree nodes will tend to experience higher traffic and will thus tend to have a higher chance of becoming latrine sites.

Figure 11 indicates that, when environment are fully rewired (i.e., $P_R = 1.0$), while Free-Roamers no longer tend to establish communal latrines at territory boundaries, Overlapper communal latrines continue to be over-represented in these boundary zones (although to a lesser extent than in unrewwired environments). Figure 12 shows example distributions of Overlapper and Free-Roamer latrines in these fully rewired environments.

Discussion

In reality, both the behaviour and environment of the European badger are, of course, far more complex than the idealised simulation presented here. They spend time on a range of activities other than foraging and defecating, and their world amounts to more than a regular array of setts surrounded by a uniform random distribution of food. In particular, there is evidence that badgers prefer to establish latrines along linear environmental features such as hedges or walls (Delahay et al., 2007), and that they commute directly to latrine locations (Roper et al., 1986). It is in this sense that the simulation presented here is a *null model*—one that establishes a set of basic expectations concerning the behaviour exhibited by a certain class of system, against which to evaluate the actual patterns discovered in empirical data.

The model demonstrates that communal latrines may arise as a result of foraging and defecation, but only if there are positive feedbacks (fæcotaxis and overmarking) that amplify existing defecation sites. The model suggests that we might expect communal latrines to arise disproportionately often at territory boundaries, as a result of the increased traffic from

multiple clans foraging in these areas. Moreover, communal latrines might arise in these same locations *before* territories and territorial behaviour are established. This raises the possibility that communal latrines might shape territoriality rather than vice versa.

The model suggests that reducing badger numbers will not tend to reduce the rate at which cattle encounter badger fæces. However, introducing more sophisticated badger behaviours to the model (such as a tendency to deliberately visit latrines, rather than stumble upon them) may reverse these findings. Conversely, the model does not consider how the transmission of infection might be related to latrine size, with small deposits potentially posing more risk to cattle than large latrines that can be easily avoided.

Finally, preliminary results suggest that there may be considerable potential for using a simulation model such as the one presented here to explore a wider set of stigmergic behaviours in environments that are not embedded in physical space, such as human behaviour on the Internet (Williams et al., 2015). Questions might include: how and when might online social media sites that are utilised by multiple groups with different competing belief systems become platforms for effective communication between them?

Conclusion

We have demonstrated that communal latrines can arise at territory boundaries disproportionately often as a result of simple stigmergic behaviours (defecation, fæcotaxis and overmarking) without the need for specific cognitive or behavioural adaptations such as memory, spatial knowledge, agonistic interactions, deliberate latrine visits, or recognition of in-group or out-group individuals or their fæcal deposits.

Significantly, communal latrines mid-way between badger setts are achieved by non-territorial badgers, suggesting that such latrines may precede territories (and even help to bring them about). While the establishment of boundary latrines is facilitated by the environment's spatial microstructure (i.e., correlation structure due to Euclidean geometry), they can be established by territorial agents even when this micro-structure is eroded completely.

A note on the title: The phrase "shit happens" is an idiom that means "some things occur without a specific reason", and thus seems apposite here despite its vulgarity.

References

Czaczkes, T. J., Heinze, J., and Ruther, J. (2015). Nest etiquette: Where ants go when nature calls. *PLoS ONE*, 10(2):e0118376. DOI: 10.1371/journal.pone.0118376.

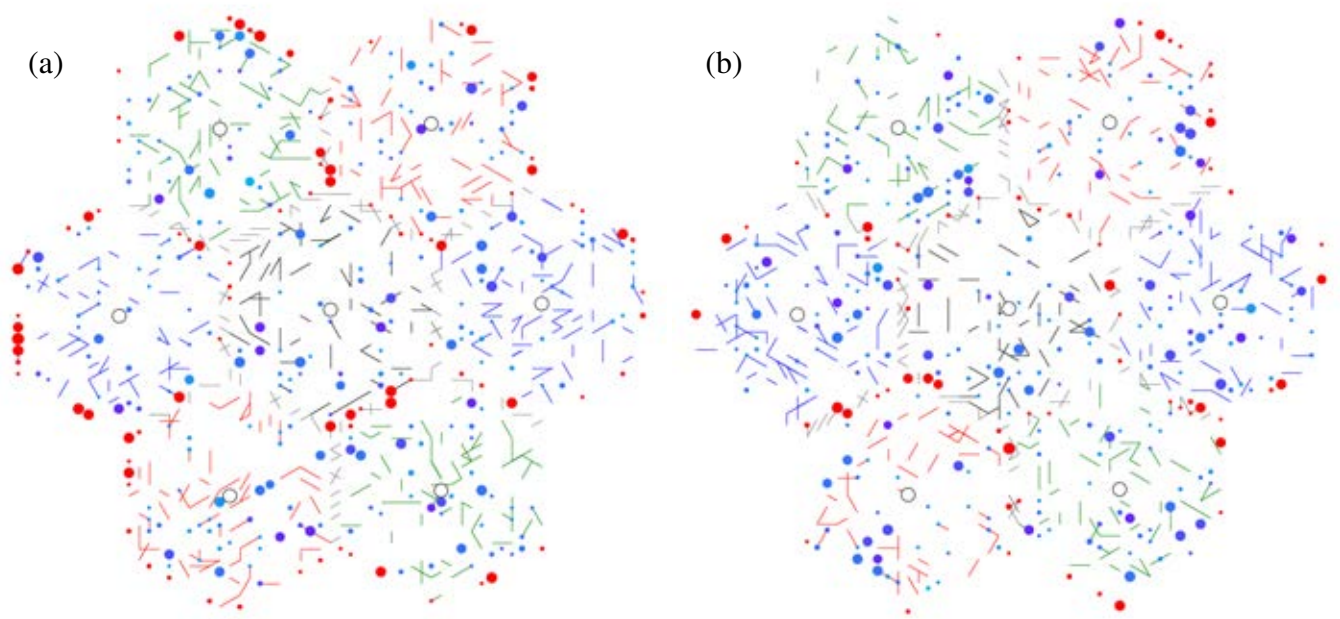Dawkins, R. (1986). *The Blind Watchmaker*. Longman.

Figure 12: Representative latrine distributions after $n = 200$ nocturnal foraging periods for (a) Overlappers and (b) Free-Roamers in environments where spatial structure has been eroded by random rewiring ($P_R = 1.0$). Line segments represent a fraction of the connections between neighbouring cells (those that link cells located within 3 spatial units of each other) to avoid visual clutter (coloured links: cells belong to the same territory; grey links: cells belong to different territories). Otherwise, data visualization and model parameters are as per Figure 5.

Delahay, R. J., Ward, A. I., Walker, N., Long, B., and Cheeseman, C. L. (2007). Distribution of badger latrines in a high-density population: Habitat selection and implications for the transmission of bovine tuberculosis to cattle. *Journal of Zoology*, 272(3):311–320.

Donnelly, C. A. and Woodroffe, R. (2015). Bovine tuberculosis: Badger-cull targets unlikely to reduce TB. *Nature*, 526(640):doi:10.1038/526640c.

Freeman, E. W., Meyer, J. M., Adendorff, J., Schulte, B. A., and Santymire, R. M. (2014). Scraping behavior of black rhinoceros is related to age and fecal gonadal metabolite concentrations. *Journal of Mammalogy*, 95(2):340–348. DOI: 10.1644/13–MAMM–A–059.

Goodwin, B. (1994). *How the Leopard Changed Its Spots*. Scribner.

Indepedent Scientific Group on Cattle TB (2007). Bovine TB: The scientific evidence. Report, UK Govt.

Irwin, M. T., Samonds, K. E., Raharison, J.-L., and Wright, P. C. (2004). Lemur latrines: Observations of latrine behavior in wild primates and possible ecological significance. *Journal of Mammalogy*, 85(3):402–427.

Kauffman, S. A. (1993). *The Origins of Order*. OUP.

Roper, T. J., Conradt, L., Butler, J., Christian, S. E., Ostler, J., and Schmid, T. K. (1993). Territorial marking with faeces in badgers (*Meles meles*): A comparison of boundary and hinterland latrine use. *Behaviour*, 127(3):289–307.

Roper, T. J., Shepherdson, D. J., and Davies, J. M. (1986). Scent marking with faeces and anal secretion in the european badger (*Meles meles*): Seasonal and spatial characteristics of latrine use in relation to territoriality. *Behaviour*, 97(1):94–117.

Stewart, P. D., Anderson, C., and Macdonald, D. W. (1997). A mechanism for passive range exclusion: Evidence from the European badger (*Meles meles*). *Journal of Theoretical Biology*, 184(3):279–289.

Stewart, P. D., MacDonald, D. W., Newman, C., and Tattersall, F. H. (2002). Behavioural mechanisms of information transmission and reception by badgers, *Meles meles*, at latrines. *Animal Behaviour*, 63(5):999–1007.

Williams, H. T. P., McMurray, J. R., Kurz, T., and Lambert, F. H. (2015). Network analysis reveals open forums and echo chambers in social media discussions of climate change. *Global Environmental Change*, 32:126–138.

Visual Navigation in Simulated Pigeons

Neil Vaughan[1]

[1]Bournemouth University, United Kingdom, Department of Computing and Informatics
nvaughan@bmth.ac.uk

Abstract

This research presents computational models to represent visual navigation mechanisms which guide pigeons (Columba Livia) during flight. A 3D graphics computer simulator was developed to model autonomous flight in virtual pigeons. The aim was to investigate the role of (i) visual landmarks (ii) flocking with other pigeons and (iii) image familiarity in pigeon navigation. A recursive processing algorithm enabled landmarks and other pigeons to be located, identified and counted. Image processing could form a feasible mechanism for autonomous visual navigation by identification of familiar route headings. This could be used in autonomous flying drones or flight simulators.

Introduction and History

Pigeons (Columba Livia) were the first domesticated bird, around 6000 years ago. This long and important history of interaction with humans has enhanced their capability to communicate. Modern pigeons can understand hand signals, voice commands and recognise individual humans even when wearing different clothing (Brett et al. 2015). During domestication, the fastest and most reliable message carrier pigeons have been favoured for breeding which has produced faster, stronger pigeons which can fly further with enhanced homing ability. Homing is defined as the process of a pigeon navigating back to its loft after being released remotely – either independently or in flocks. Pigeons have been known to successfully navigate home to their loft from 7200 miles which is almost one third of Earth's circumference. Racing pigeons commonly fly at speeds of up to 80 miles per hour, flying for 24 hours without stopping.

Navigation mechanism in pigeons. Pigeon navigation mechanisms involve (1) the position of the sun in the sky relative to the hour of day, known as solar clock (2) sensation of the earth's magnetic forces, known as magnetoception (3) visual recognition of landmarks especially for the region closest to the loft. The solar clock is preferred by the pigeon, however on cloudy or rainy days it is not possible to use the sun and pigeons then switch to navigation by magnetoception. This has been proven by releasing 'time shifted' pigeons (Biro et al., 2004). Their blacked-out loft had no sunlight and artificial lighting was used to offset sunrise and sunset by 2 hours from the actual sunset. When released, 'time shifted' pigeons set off at the wrong angle, taking a long detour with increased chance of becoming lost, showing that pigeons navigate by solar clock when the sun is visible. However, on cloudy days the 'time shifted' pigeons were unaffected, flying home in a straight line, proving that when the sun is not visible, pigeons navigate by magnetoception.

Breeders generally lose pigeons each year due to hazards including farmland shooting, colliding with high sided vehicles and power cables. Some losses are unavoidable but understanding navigation mechanisms could reduce losses in training caused by navigation issues or weather.

Pigeons are an example of expert visual navigators but it is not known how their vision is optimised for navigation or how their optical array is optimised for view matching strategies to enable orientation and navigation. Between the eyes of various aerial animals, there is enormous variation in the information provided by visual systems (Gaffin et al. 2015). Mechanisms of insect navigation have recently been investigated by computer simulation (Wystrach et al., 2015).

Video recording during pigeon flight

Small lightweight video camera devices have been mounted onto a pigeon during flight which revealed the bird's eye view during pigeon navigation (Fig. 1). Results from analysis of these videos and real-time GPS tracking data during pigeon flights show that pigeons use visual navigation to follow roads, rivers, or coastlines which lead towards their loft during flights within familiar territory or within the last leg of a longer journey (Biro et al., 2004). These findings suggest that navigation within one mile of the loft is led by visual navigation of recognised landmarks. The outline of landmarks must be visible for pigeons to recognise them correctly (Gibson et al., 2015). This suggests that navigation by solar clock and magnetoception may not be accurate enough for locating the exact loft, but guides to within the correct mile. This would explain why pigeons become lost when released without first seeing and becoming familiar with the loft's surroundings, which is a core part of training for homing pigeons. Within visual navigation (i) landmarks, (ii) flocking and (iii) familiar scenes play a part in guiding pigeons. This research aims to identify their roles in simulated navigation.

Fig. 1. Video device mounted on a pigeon during flight to monitor visual navigation [National Geographic, 2011].

Methods - Simulated Pigeon Visual Navigation

Virtual Reality habitats were generated using the GLUT framework to represent environments encountered by pigeons in their natural habitat (Fig. 2). For pigeons this consisted of a random assortment of plants and trees on hilly ground with scattered areas of water. Plants and tree objects were created using random configurations based on geometric patterns. The size of trees in the virtual environment was based on the scale of objects as perceived by the pigeons in their natural habitat at flight levels between 0-100m. Plants and trees were rendered as 3D objects. A flock of 100 pigeons was simulated, each navigating independently.

Pigeon's eye view. At each iteration of the simulator, a bird's eye view of each virtual pigeon in the sky was generated and stored as an image using perspective projection view from the position (latitude, longitude and altitude) within virtual world coordinates and, direction (elevation and azimuth) in degrees. The generated image proportions aim to match characteristics of pigeon eyes in terms of the degrees of panoramic view from the horizon. Various combinations of colour and resolution were investigated to represent pigeon's eye characteristics and identify the effect on ability to navigate. The pigeon's eye view images were used for (1) image processing to identify landmark features within the image, and (2) a neural network to assess familiarity of images.
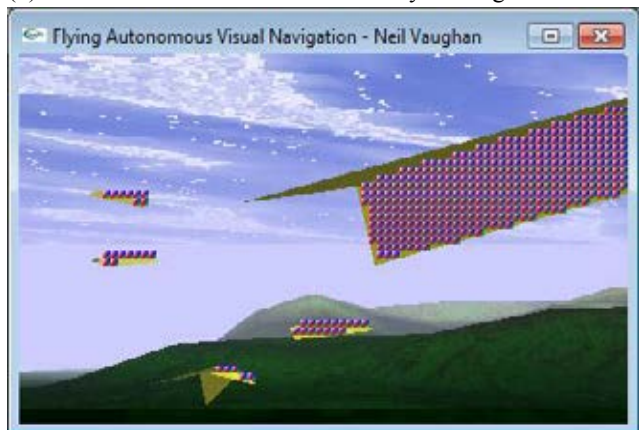

Fig. 2. Virtual Simulation of autonomous pigeon during flight.

Image processing: landmark recognition. Real-time Image processing algorithms were developed to process the pigeon's eye view images to identify (i) known landmarks and (ii) other pigeons to guide flocking behaviour. Recursive image processing was used to locate centroid of each object cluster (Fig. 3). A matrix represents all pixels within view. For pixels within the landmark RGB threshold, a recursive function tests the pixel values of four surrounding pixels (non-diagonal).

Neural Network Training. A neural network was used to assess the familiarity of images from the pigeon's eye view. The ANN was trained with stored images taken at regular intervals along a known training route through the habitat, from a familiar release site back to the loft (Wystrach et al. 2015). Distance between stored training images represents the pigeon's memory capability. The neural network required the familiar route to be clearly distinguishable from others.
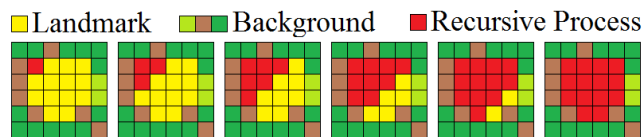

Fig. 3. Clustering recursion algorithm to identify landmarks.

Results and Performance Analysis of Recursion

The recursive image progressing algorithm from a pigeon's view allowed all pixels within landmarks to be identified. Size and centre of each cluster was calculated from sum of pixels. This enabled identification of the number, size and location of landmarks. This could enable navigation based on the location of known landmarks to recover accurate route headings from visual perception of landmark locations. The pigeons are required to learn a map of the landmark layout in relation to the loft. Learning the loft location relative to landmarks is an essential part of the training process. The number of other pigeons was identified by image processing, enabling flocking behaviour by responding to flight paths of other pigeons.

Conclusion

In 2016 there is increasing investment in autonomous navigation research. Beneficiaries include: driverless cars, flying delivery drones, autonomous robotics, game engines, auto-pilots and augmented smartphones.

This research used virtual models of autonomous agents within virtual environments. Image processing was used to develop visual navigation based on optical performance of pigeons. VR can provide a platform to develop and test algorithms for visual perception and navigation responding to sensory information in real-time.

The developed visual navigation algorithms from the simulator model could be transferred onto a drone with a microcontroller and camera to detect landmarks and other drones using image processing. This could enable a drone to use landmarks and scene familiarity for visual navigation and to flock or avoid collisions with other drones.

Recent research generated preliminary results validating feasibility of image processing for autonomous visual navigation (Vaughan, 2015). Familiarity of images could enable autonomous agents to choose the optimum orientation and navigation towards a trained location.

References

Biro, D. et al. (2004). Familiar route loyalty implies visual pilotage in the homing pigeon. *Proceedings of the National Academy of Sciences of the United States of America*, 101(50), 17440-17443.

Gibson, Brett M., Andrew B. Leber, and Max L. Mehlman. "Spatial context learning in pigeons (Columba livia)." Journal of *Experimental Psychology: Animal Learning and Cognition* 41, no. 4 (2015): 336.

Gaffin DD, Dewar A, Graham P, Philippides A (2015) Insect-Inspired Navigation Algorithm for an Aerial Agent Using Satellite Imagery. PLoS ONE 10(4): e0122077.

National Geographic, *Brilliant Beasts Pigeon Geniu*s, 2011.

Vaughan, N., 2015. Simulated Robotic Autonomous Agents with Motion Evolution. *European Conference on Artificial Life ECAL*.

Wystrach A, Dewar ADM, Philippides A, Graham P (2016) How do field of view and resolution affect the information content of panoramic scenes for visual navigation? A computational investigation. *Journal of Comparative Physiology* A, 202 (2). pp. 87-95.

A Level Set Approach to Simulating *Xenopus laevis* Tail Regeneration

Zachary Serlin[1], Jason Rife[1] and Michael Levin[2]

[1]Automated Systems and Robotics Laboratory, Tufts University, Medford, MA 02155
[2]Center for Regenerative and Developmental Biology, Tufts University, Medford, MA 02155
Zachary.Serlin@tufts.edu; Jason.Rife@tufts.edu; Michael.Levin@tufts.edu

Abstract

A framework for predictively linking cell-level signaling with larger scale patterning in regeneration and growth has yet to be created within the field of regenerative biology. If this could be achieved, regeneration (controlled cell growth), cancer (uncontrolled cell growth), and birth defects (mispatterning of cell growth) could be more easily understood and manipulated. This paper looks to create a key part of this preliminary framework by using level set methods and a cellular control scheme to predict macroscopic regenerative morphology. This simulation specifically looks at *Xenopus laevis* tail regeneration, and uses three control regimes to collectively mimic biological regeneration. The algorithm shows promise in creating an abstracted model to predict cell patterning on a macroscopic level.

Introduction

If the control of cell growth and tissue patterning can be better understood, cancer (uncontrolled cell growth), birth defects (mispatterning), and organ regeneration (cell growth harnessed toward the repair of complex organs) could be more easily manipulated. While the molecular mechanisms of cellular control are increasingly understood, the field lacks frameworks for predictively linking cell-level signals to large-scale pattern controls. This paper looks to leverage methods from continuum mechanics to provide new tools for modeling the control of cell growth and patterning. To do this, regeneration in tadpole tails is modeled as an iteration between two processes. The first process is a control scheme, which decides where and when tissue should grow or shrink. The second process is a growth model that describes changes in tissue morphology due to cell division, motion, and growth.
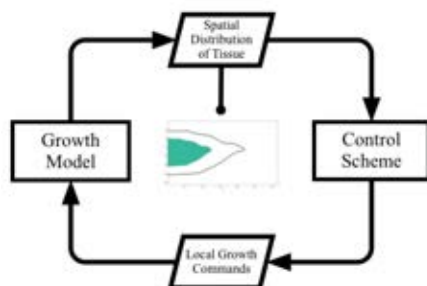

Figure 1: Two-process system model. The control scheme creates local growth commands, which the growth model uses to outputs the spatial distribution of tissue.

One challenge in modeling regeneration and tissue growth is relating the macro (organism) scale to the micro (cell) scale. Traditionally, as the scale of the organism increases, so does the computational expense of modeling its smallest features and interactions. The average human is composed of 37.2 trillion cells – an unrealistic number of cells and interactions to model (Bianconi et al. 2013). Our proposed method is advantageous because it treats tissue as a continuum, blurring the boundaries between individual cells. This approach avoids the problem of managing cells as individual agents, which like marbles on a Chinese checkers board, would need to be shuffled to open spaces to make room for new marbles.

What we describe is the difference between tracking individual agents and tracking the motion of bulk material through a fixed volume of space. Two types of mathematical thinking exist to distinguish these types of phenomena. The first approach is Lagrangian, meaning cells are tracked on an individual basis. Such a method allows cells to operate on their own growth rules and is commonly used in biological growth modeling (Walker et al. 2004; Rejniak and Anderson 2011). Lagrange models often suffer morphologically from internal voids because the individual "cells" cannot directly organize in a manner that preserves contact without overlapping. The second approach is Eulerian, meaning it focuses on the space through which particles move. Eulerian approaches are classically employed in modeling fluid flow and heat transfer (J. A. Sethian 1985; Osher and Sethian 1988). Such methods have not been widely used to model tissue growth and patterning; however, they offer great promise to capture the effects of microscopic phenomena interacting across macroscopic domains.

Our proposed model uses an Eulerian approach based on the level-set method. Level sets use a modified mass balance to describe a moving boundary, such as the interface between an organism and the surrounding medium. Level set methods are used to model crystal growth and combustion, as well as for computer vision and microchip fabrication (Osher and Sethian 1988; J. A. Sethian 1985; J. Sethian 1984; J. A. Sethian, n.d.). Level set methods have also seen some use in biological modeling (C.S. Hogea, Murray, and Sethian, n.d.) although not to our knowledge in a closed-loop feedback scheme for patterned growth as illustrated in Figure 1.

Level set methods use a scalar field to describe a moving boundary. The boundary is at the zero values of the scalar field and motion of the boundary is determined by assigning a speed at each point. The speed function is ultimately what controls the development of the boundary. For this biology-motivated

application, we propose a speed function consisting of three main components: isometric control, patterning control, and smoothing control.

This paper uses level sets to simulate the regeneration of the amputated tail of a *Xenopus laevis* tadpole. *Xenopus* is a simple vertebrate that regenerates its tail until early in its life cycle, through stage 52 or 53 (Suzuki et al. 2006). This makes *Xenopus* ideal for modeling patterning growth, and regeneration in particular, across a macroscopic scale. In this work, we conceptualize regeneration as growth that restores animal morphology back to a "reference" shape. Our particular approach will assume that a global reference map is available and that control laws act by setting growth rate based on the distance of the organism boundary from the reference. In fact, it is not critical as to whether an actual reference "map" might exist in an animal system (Friston et al. 2015) or whether the "map" is an emergent pattern that results from local control decisions (Zhang and Levin 2009; Chernet, Fields, and Levin 2015). On the simulation scale, both mechanisms are functionally equivalent.

In the following sections, this paper will detail methods used to implement the growth model and the control scheme. The paper will then go into detail on the simulation used to assess these models. Finally, we will present results, conclusions, and future work.

Methods

Level Set Analogy

Level set methods were originally created to model combustion and two-phase flow (J. Sethian 1984). A level set can be conceived as a geographic contour map, where each level set is an elevation contour that consists of the set of points at a particular elevation. These contours may change over time if geography changes. By following a particular contour in time, it is possible to model the motion of an interface (e.g. a flame front in combustion or a liquid-gas interface in two-phase flow). The level set equation is directly related to the scalar transport equation. Instead of transporting material however, the level set method transports a scalar distance from the boundary of interest.

To make the geographic analogy more concrete, consider a particular case – a volcanic island rising out of the ocean (Figure 2). In this example, we will track *sea level* over time, as this elevation marks the interface between the island and the ocean. As a volcanic eruption takes place and adds material over the entire island, the island's elevation map will evolve both on the land side (topography) and on the ocean side (bathymetry). The addition of new material will cause the island to grow, such that the sea-level elevation contour pushes outward as shown in Figure 2. Erosion would garner the opposite effect – sharp features would be worn down, shrinking the boundary. In Figure 2, the elevation is denoted by the scalar ϕ. The interface between land and water is denoted by the sea-level contour, with $\phi = 0$.

This geographic concept can be extended to biological modeling. In this paper, we consider a two-dimensional model of the *Xenopus* tail. In our model, the x coordinate corresponds to the anterior-posterior direction, the y coordinate to the dorsal-ventral direction, and the lateral direction is not modeled. The level set field $\phi(x,y)$ is now used to represent distance away from the outer surface of the organism. The

contour that represents the outer surface of the organism is labeled Γ and represents the set of all points where $\phi = 0$. Moving inside the organism, the distance-from-contour is measured as positive and so ϕ is set positive inside the organism. By contrast, ϕ is set negative outside the organism. This mathematical approach lets us track the motion of the outer surface of the organism through space as the organism grows in time. An illustration of this concept is shown in Figure 3, which depicts a section of the *Xenopus* tail. A photo of the *Xenpous* tadpole, in Figure 3(a), shows the tail prior to amputation. The outer tail surface can be identified and used to
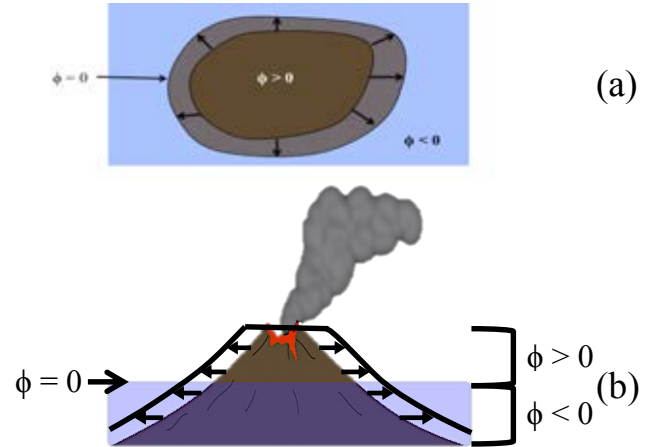
Figure 2: (a) Top view of the volcanic island erupting – arrows indicate boundary movement. (b) Front view of the volcanic island describing the same boundary movement. Elevation above sea level is indicated by the variable ϕ. Addition of volcanic material increases elevation and translates directly to change in island circumference.
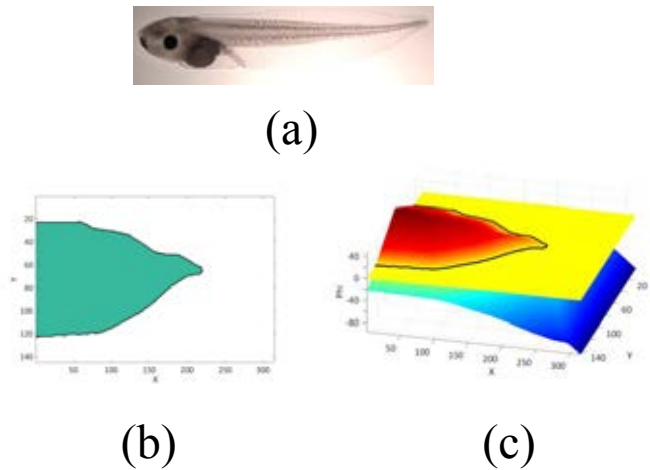
Figure 3: Level Set Scalar Field Derivation (a) *Xenopus laevis* at stage 42. (b) A 2D view of the tail representation. This represents the morphology of the tail and is the zero level set contour of the scalar field. (c) A 3D representation of the level set field. The flat plane is the zero level set. Above that plane represents material inside the body; while, the section below that plane is outside the body. The black line represents the current zero contour.

generate a binary image, as shown in Figure 3(b), where the green region indicates the interior of the organism and the light region indicates the exterior. This tail region can also be viewed as a level set field, shown in three-dimensions in Figure 3(c). The height of the field indicates the distance of each (x,y) point from the outer surface of the organism.

Growth Model

This section describes the level set equations we used to model organism growth. Begin by identifying the outer surface of the organism in a Cartesian space described by coordinates x and y. In two-dimensions, the outer surface of the animal is a contour, which we label Γ as shown in Figure 3(b). Now construct a scalar field $\phi(x,y)$ that represents distance from the contour Γ, with values increasing interior to the organism ($\phi > 0$) and decreasing exterior to the organism ($\phi < 0$). In order to represent distance, note that the gradient of this scalar field must be one at all points where the slope of the field is continuous. Slope discontinuities appear only at the center of the field, where points are equidistant from multiple sections of the Γ contour, as illustrated by the ridge that appears along the midline of the tail, as illustrated in Figure 3(c).

To model growth, we evolve the scalar field ϕ over time. As time advances, the level set is propagated using a velocity field $\mathbf{v}(x,y)$, where the velocity vector is specified at every point in the field. The magnitude F of the velocity vector will be set by the control scheme, as described below. As the control scheme transform the scalar field, the outer surface of the organism Γ moves in time, representing organism growth.

The following equation governs the time dynamics of the scalar field $\phi(x, y, t)$.

$$\frac{D}{Dt}\phi = S(x,y,t) \tag{1}$$

Here the full derivative of the scalar ϕ is related to a source term S. The source term allows for the production of new material (or the destruction of old material) at every point in the field. Where the source term is zero, there is no change in the total amount of material present; in other words, ϕ is conserved in the absence of a source term. Equation (1) is a classical conservation law from continuum mechanics, as might be used to model the conservation of mass, momentum, or energy (Kundu, Cohen, and David R. Dowling Ph.D. 2011).

The full derivative is linked to velocity $\mathbf{v}$ through the following equation.

$$\frac{D}{Dt}\phi(x,y,t) = \phi_t + \nabla\phi \cdot \mathbf{v} \tag{2}$$

This equation, obtained from standard calculus using the chain rule, uses the notation ϕ_t to identify the partial derivative of ϕ with respect to time and the notation $\nabla\phi$ to identify its spatial gradient. In this work, ϕ contours are assumed always to move outward in the direction normal to each existing contour. The local unit normal to each contour is defined by the gradient:

$$\mathbf{n} = \frac{\nabla\phi}{\|\nabla\phi\|} \tag{3}$$

The velocity vector can be written in terms of a magnitude term F multiplied by this local normal.

$$\mathbf{v} = -F\mathbf{n} \tag{4}$$

The negative sign is introduced here, so that a positive speed F corresponds to organism growth (toward lower values of ϕ).
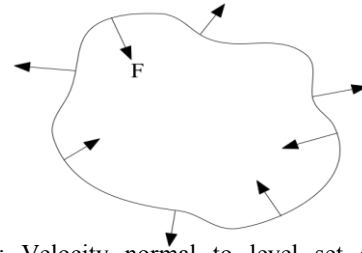

Figure 4: Velocity normal to level set contour. All movement is normal to each point on the contour. Only the magnitude of F determines contour movement.

A field in which velocity is locally normal to the contour is shown in Figure 4.

Substituting equation (4) into equation (2) gives

$$\frac{D}{Dt}\phi(x,y,t) = \phi_t - F\|\nabla\phi\| \tag{5}$$

Recalling that the gradient is equal to one at all points where it is defined, the full derivative becomes

$$\frac{D}{Dt}\phi(x,y,t) = \phi_t - F \tag{6}$$

where the gradient $\nabla\phi$ is continuous. To avoid issues with the gradient being undefined at some locations (at cusps and ridges in the ϕ field), the velocity magnitude F is restricted to be zero at these locations. Thus, by combining equations (1) and (6), we obtain the following equation to describe the change in the level set field ϕ at each point and at each moment in time.

$$\phi_t = \begin{cases} F + S, & \nabla\phi\ defined \\ S, & elsewhere \end{cases} \tag{7}$$

Because this equation for propagating the ϕ field behaves differently in regions where the gradient is either continuous or not, it is natural to decompose our solution approach into two parts. In the first part of the solution, we update the field at each time step assuming that the source term is negligible. For this step we use a first-order discretization of equation (7).

$$\phi(t + \Delta t) = \begin{cases} \phi(t) + F\Delta t, & \nabla\phi\ defined \\ \phi(t), & elsewhere \end{cases} \tag{8}$$

Assuming a negligible source term is reasonable over short time periods; however, over longer periods neglecting the source term rapidly degrades the assumption that the gradient is unity magnitude (where defined), since a source should exist at peaks and ridges (as in the Volcano example of Figure 2). As such, the source term must be taken into account somehow.

To account for the source term, we use a process called reinitialization (J. A. Sethian 1985; Osher and Sethian 1988; Brakke 2015; Evans and Spruck 1991). Reinitialization serves two purposes; it forces the non-boundary region to have a gradient of one, and it implicitly adds material to the whole field to maintain the field's shape. In particular, we use a process called a narrowband reinitialization (J. A. Sethian, n.d.). The narrowband solution assumes that the location of the zero-contour Γ is predicted accurately. The solution domain is then divided into two regions: a region near the zero-contour (the *interface* region) and the region farther from the zero-contour (the *far field*). Values of ϕ in the interface region are

preserved; values in the far field are replaced by computing the distance of each location from the zero contour. Although this process does not correct the gradient inside the interface region, the far field values effectively introduce a boundary condition that drives the slope in the interface region back toward its correct magnitude (of one). This narrowband approach is numerically robust and has been used extensively in other applications of the level set method (J. A. Sethian, n.d.).

A practical issue is that the approximations introduced by narrow banding can affect the accuracy of the prediction of the zero-contour Γ. A balance must be struck however between giving the zero-contour freedom of movement and constraining the field to a gradient of one. For this reason, reinitialization is not necessarily performed at every time update. In our method, we used a boundary width of 2 pixels on either side of the Γ contour and performed reinitialization at a rate of once per 20 time steps.

In summary, the key idea of the growth model is that a velocity field can be assigned to every point in space, allowing the organism surface Γ to be grown without knowing the precise location of that surface. This property is in turn useful because it permits the simulation of a smooth, continuous boundary using a relatively coarse grid.

Control Scheme

In this section we describe a control law that can be used to in conjunction with the level set methodology to define organism shape during regeneration. The control law defines a velocity field at every point in the simulation domain. As described by equation (4), the velocity is locally normal to contours of constant ϕ. The velocity magnitude F is set by the control described in this section. Specifically, F is a summation of three terms, which are assumed to act independent of each other. These terms include patterning control P, isometric control I, and smoothing control K. Each term models a distinct aspect of biological growth. It is important to note, however, that the models are phenomenological in nature and are not derived directly from detailed data sets. Taken together, the three terms sum to give F.

$$F = P + I + K \tag{9}$$

All terms in this equation are functions of 2D space and time.

Patterning Control: Patterning control, P, is the key term for this paper as it shapes morphology by enabling local growth. The idea is that local cell-level actions may trigger tissue deterioration or growth in a small region (as at the regeneration site or *blastema* in an amputated *Xenopus* tail (King and Newmark 2012)). These local actions are responsible for regeneration and also for shape changes that occur during normal growth. Furthermore, this local activity counteracts disturbances, constantly adding or removing tissue to maintain an appropriate organism shape under varying environmental conditions. In principle, a failure of local patterning might result in uncontrolled growth (i.e. cancer).

In our simulation, we assume that patterning growth is active for cells that are near the organism surface Γ but that are not at a desired location. For simplicity, we use a global reference map ϕ_{ref}, which is scaled to the current width of the simulated tail. From this map, we derive an error e at each point in the simulation domain.

$$e(x, y, t) = \phi_{ref}(x, y, t) - \phi(x, y, t) \tag{10}$$

The error for a given element of tissue represents the difference between its desired and actual distance from the organism surface. The error term is limited to a maximum value, e_{max}, to reflect a threshold where the cells are so far from their target that they grow at a maximum rate. In the region near the organism surface, the patterning speed is set to be proportional to the error, modulated by a patterning control gain labeled C_P.

This proportionality is capped, however, to reflect a maximum cellular growth rate, which is slightly faster than nominal, isometric growth. The maximum speed F_{max} is related to the maximum cellular growth rate $C_{GC,max}$.

$$F_{max} = \frac{V_{pat}}{SA} C_{GC,max} \tag{11}$$

Here the variable SA represents the surface area (which in 2D is the length of the contour where $\phi = 0$). The variable V_{pat} represents the volume of tissue that is active in patterning and is proportional to V_{tot}. The result is that patterning growth is nonzero only in the active region; in this region patterning growth is nominally proportional to error, subject to saturation if the growth rate becomes too large or too small.

$$P = \begin{cases} F_{max} \frac{e}{e_{max}} & |e| < e_{max} \text{ and } 0 < \phi \leq d \\ F_{max} \text{sign}(e) & |e| \geq e_{max} \text{ and } 0 < \phi \leq d \\ 0 & otherwise \end{cases} \tag{12}$$

Isometric Control: Our simulation uses an isometric control term I to allow for growth that is organism wide (as compared to patterning growth which is local). We assume that this growth occurs at the same rate throughout the entire organism, such that the organism maintains its shape when only I is active. Isometric growth is, in fact, a nominal behavior for some organisms such as the flatworm. When food resources are plentiful, the flatworm grows uniformly in all directions; when the flatworm is starved it shrinks uniformly in all directions (Lobo, Beane, and Levin 2012). In *Xenopus* tadpoles, by contrast, some changes in shape occur as the organism grows (Love et al. 2013; Suzuki et al. 2006; Chernet, Fields, and Levin 2015), and so nominal growth combines some aspects of isometric control I with patterning control P.

The isometric control term has a uniform value of C_V everywhere in the simulation domain when the term is active.

$$I = F_V \tag{13}$$

The isometric model represents constant cellular growth with time, meaning the boundary velocity must increase in time (as the volume to surface area ratio increases). To account for this, the growth speed F_v is computed as

$$F_V = \frac{V_{tot}}{SA} C_{GC,nom} \tag{14}$$

Here V_{tot} represents the total volume of the organism (or in this 2D simulation, the area of the tail). The rate $C_{GC,nom}$ represents the rate of cellular growth (mitosis), which is modeled to be uniform in space . For our simulations we assume that sufficient resources are available to the organism to maintain a nominal growth rate $C_{GC,nom}$ that is constant in time.

Smoothing Control: The final term of the speed function is a smoothing control term K designed to eliminate sharp features (e.g., corners created by amputation) or to eliminate tissue

filaments that might (by random chance) begin to develop as extensions of the organism surface. This term essentially regularizes the organism surface to maintain smoothness. The specific mechanism for performing smoothing is to introduce a perturbation to the growth rate that is proportional to the local curvature κ of the ϕ contours. The constant of proportionality is C_κ.

$$K(x,y,t) = C_\kappa \kappa(x,y,t) \qquad (15)$$

This concept for smoothing has been employed in other applications of level set methods, as described by (J. A. Sethian, n.d.; Brakke 2015; Evans and Spruck 1991; Osher and Sethian 1988). In our biological application, the curvature term not only eliminates spurious features; it is meant to prevent the formation of holes and discontinuities that are not represented in the reference map.

By definition, curvature in a level set is the spatial derivative of normal vectors along a contour. The more rapidly the contour changes direction, the higher will be its curvature. Mathematically, curvature can be written

$$\kappa = \nabla \cdot \mathbf{n} \qquad (16)$$

where the normal vector $\mathbf{n}$ is defined by equation (3) above.

Each control regimes can be linked directly to micro scale cell behavior that impacts macro scale morphology. Patterning control begins by considering a maximum growth rate of individual regeneration, multiplies that growth by the number of patterning cells, and then distributes that growth along the growing boundary. Similarly, isometric control begins by considering the natural growth rate of cell division, multiples that growth by the cells in the tail, and distributes this growth over the body boundary by dividing by the surface area of the tail. Smoothing control is not as simply related to cell growth, but instead represents cohesiveness in the cellular matrix by minimizing areas of high curvature or irregularity.

Implementation

Each time step is represented by a single iteration of the main loop shown in Figure 5. In general, the simulation is allowed to run for 15,000 time steps, but this number must be adapted to grid resolution and control coefficients.

The level set ϕ is stored as a two-dimensional array on a Cartesian x-y grid. In this simulation the size of the grid was 301 pixels (anterior-posterior) and 135 pixels (dorsal-ventral). Two extra cells pad the field on each edge to simplify gradient and curvature calculations.

Computing $\|\nabla\phi\|$ (labeled Magnitude of Gradient of Phi block in Figure 5) introduces a potential source of numerical error, as derivative operations amplify numerical errors. Therefore, a weak Gaussian filter was introduced in this block to smooth gradient values. The filter uses a two-dimensional Gaussian kernel with a standard deviation of 0.25.

At each step, the surface of the organism ($\phi = 0$) may lie between cells. No attempt is made to interpolate the actual surface (e.g. red contour shown in Figure 6). Rather, figures in this paper report all cells with $\phi > 0$ as being part of the organism and all cells with $\phi < 0$ as being exterior to the organism.
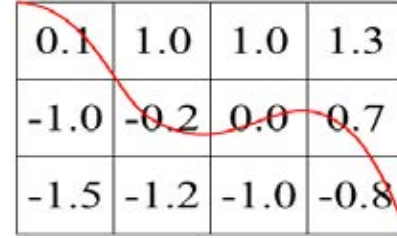
Figure 6: The boundary (red curve) may lie between cell centers, as shown on this grid.

Results of Simulation Verification

A suite of four test cases were simulated in order to verify the basic functionality of the algorithm. The four test cases all considered a simulation domain modeling a stereotypical *Xenopus* tadpole tail. The final tail morphology is derived from Reid et. al. (Reid, Song, and Zhao 2009) and The Normal Table of Xenopus laevis (Faber and Nieuwkoop 1967). The four tests include (a) no growth, (b) patterning-based regeneration following amputation, (c) nominal isometric growth, and (d) nominal isometric growth and simultaneous regeneration following amputation. These test cases were selected respectively to examine algorithm stability, performance of patterning control (in isolation), performance of isometric control (in isolation), and performance of combined control terms.

Patterning and isometric control parameters are derived from experimental regeneration data. All four test cases have (unless otherwise noted) $C_{Gc,nom} = 0.00075$, $C_\kappa = 0.001$, $C_p = 0.02$, $d = 3$, $e_{max} = 10$, $C_{Gc,max} = 0.1$, 1 time step = 12 minutes, and are run for 15,000 time steps. The patterning control coefficient (C_p), was found using the Reid et. al. image sequence and analyzing its length growth rate. Isometric control coefficient

Figure 5: Implementation of primary simulation loop.

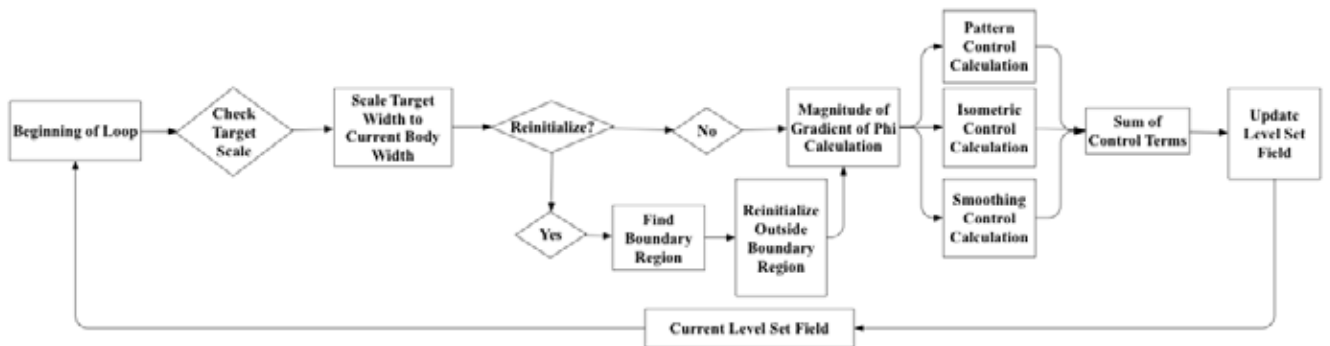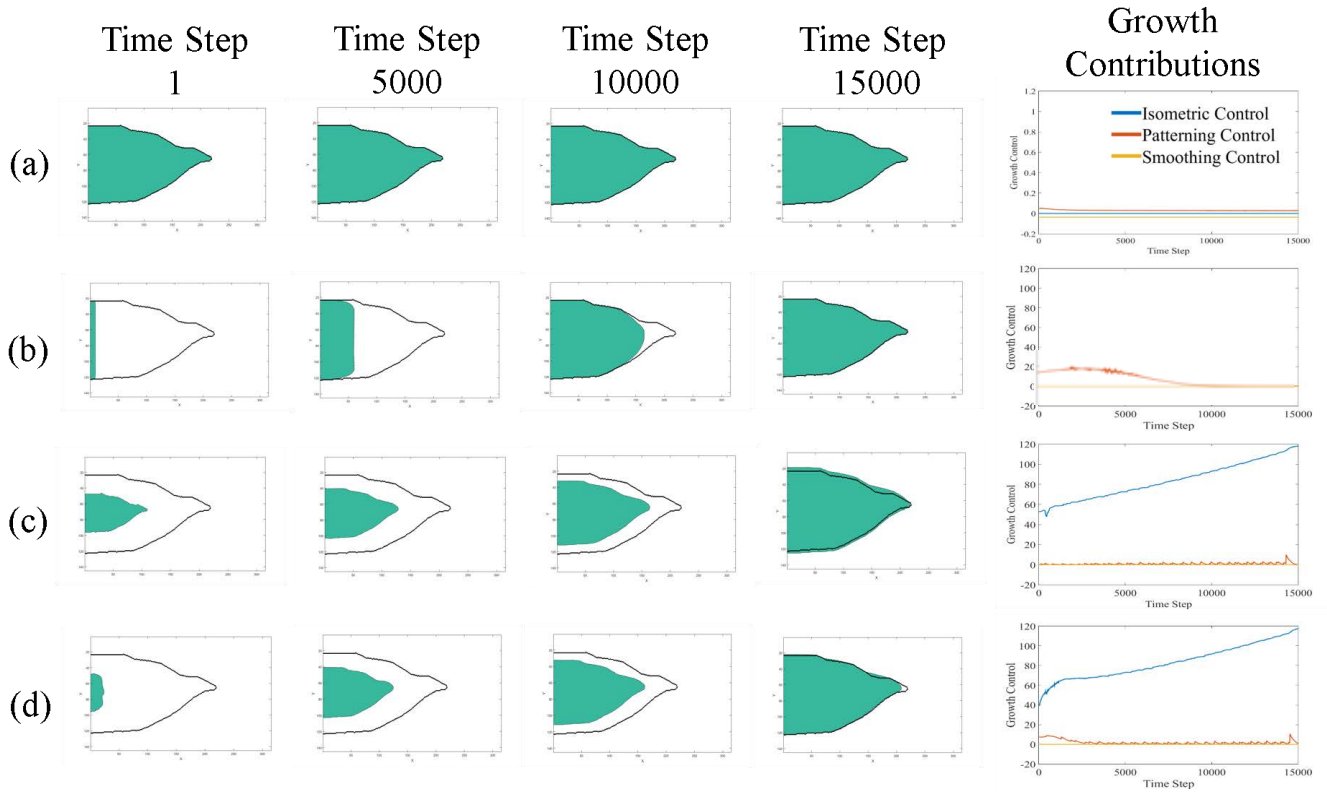Figure 7: Verification tests: (a) No Growth, (b) Regeneration Following Amputation (c) Nominal Growth (d) Nominal Growth and Simultaneous Regeneration Following Amputation. The right most graph shows contributions of individual control terms to F, integrated over the simulation domain.

($C_{Gc,nom}$) was found using the normal table of *Xenopus laevis* stage series (Faber and Nieuwkoop 1967). The pixel area of the organism was evaluated from experimental images, between stages 40 and 52. Data were plotted and the value of the growth rate was determined by a linear fit of the data.

In analyzing the four test cases, it is useful to consider image sequences that illustrate the growth process. Figure 7 shows four image sequences, one for each test cases. At the end of each row, an additional plot shows growth contributions per time step for each term (P, I, and K). Each image sequence starts from the left; the green body is the current body shape, and the black outline is the final target reference. In the growth contributions plot (far right) the horizontal axis indicates time and the vertical axis indicates the growth contribution (for P, I, and K) integrated over the entire organism for each time step in units. Growth is measured in terms of grid cells, or pixels, that the organism fills. Hence the units of the vertical axis can be considered to be pixels per unit time. The area under each of the growth contribution curves can loosely be viewed as the overall contribution of material due to each growth contribution (P, I, and K) from the start of the simulation.

Case A: No Growth

The first test case looks at the stability of the patterning algorithm by considers a fully-grown tail where the surface of the organism is already at its reference location and where the nominal growth term is shut off ($C_{GC,nom} = 0$). One would expect there should be no change in body shape, and little to no contribution from any of the active control regimes, since the

initial organism shape matches the reference contour. Indeed, the image sequence in Figure 7(a) shows qualitatively that the tail remains stationary. Growth contributions are tiny but nonzero (smaller than 0.1 pixel/time step, as shown in the growth contributions plot at the end of the row). Though nonzero, the growth contributions from patterning (positive) and smoothing (negative) are essentially balanced. The implication is that the smoothing term is continually active at a low level and that the patterning term compensates, such that the two remain in static equilibrium. The fact that the system reaches equilibrium indicates that the algorithm is in fact stable.

Case B: Regeneration Following Amputation

The second test case considers a simplified model of regeneration following amputation, with no nominal organism growth. The purpose of the test case is to examine the performance of the patterning control term. The amputation is performed digitally, with the tail being "cut" at the initial time to leave tissue only in the 10 leftmost grid cells of the image. For this simulation, nominal growth is again disabled ($C_{Gc,nom} = 0$). Under these conditions, we would expect the tail to return its nominal shape (pre-amputation), with tissue filling the reference map much as material might flow into a mold.

Figure 7(b) shows that by time step 15,000, the tail has in fact regenerated to its nominal shape and size. Note that, for Case B only, the patterning growth rate was reduced relative to its nominal value (to $C_p = 0.005$), in order to better visualize the growth process. As shown in the initial sequence, the corners created in the amputation persist through time step 5,000.

Together the smoothing term and the reference image (which inhibits patterning growth when the organism boundary reaches the reference boundary) introduce more curvature into the regenerating tail, as seen by time step 10,000. The smoothing term becomes more active as the tail becomes sharper, but eventually the patterning term overcomes the smoothing term to fill the pointed tip of the tail, as seen in time step 15,000.

The growth contribution plot shows that total patterning growth ramps up slowly to a peak around time step 4,000 before tapering toward zero approaching time step 10,000. The explanation is that initially the amputation boundary grows nearly straight (toward the right), such that the total patterning growth (which is proportional to the size of the amputation surface) is nearly constant with some slight rise due to the curvature appearing at the top and bottom corners of the amputated face. As the tail narrows, the surface area of the amputation face grows narrower and the amount of patterning growth falls quickly toward zero. Note that the scale of the growth contribution plot for Case B is zoomed out by 100 times relative to that of Case A, indicating much, much higher growth rates in Case B (as would be expected).

Case C: Nominal Growth

The third test case mimics nominal growth with no amputation or regeneration. In this example, the initial organism is set to the same shape as the initial organism in Case A, but scaled down in size by 50%. In concept, one would expect only isometric control would be necessary to grow the tail, even with patterning control active.

The simulation indicates that the organism shape is preserved during growth, as shown in Figure 7(c). The growth contributions plot for this row shows that the majority of all growth is generated by the isometric growth term. Though the patterning growth term is active, it remains essentially zero throughout the simulation, as expected. (The exception is a small spike near time step 15,000, triggered by a combination of the reinitialization process and by erosion due to smoothing). The final image of the sequence shows the simulated organism overgrows the reference contour slightly, as can be observed near the tip of the tail. This occurs largely because of the action of the smoothing term, which rounded the tail during otherwise isometric growth.

As a final note, it is worth observing that the organism growth rate increases slightly over time, as is evident from the growth contributions plot at the end of row (c) in Figure 7. The acceleration of growth over time matches the intent of equation (14), which was designed to keep the rate of cellular division constant, with the implication that the organisms total growth (pixels added per time step) should become faster as the organism becomes larger.

Case D: Nominal Growth and Regeneration

A final test case combines Cases B and C to provide a more realistic model for amputation, one in which patterning growth occurs in parallel with nominal growth. For this case, all three control terms are active simultaneously (with control parameters set to their nominal values).

The Case D simulation confirms that the patterning and isometric growth terms complement each other when they are both active, allowing the organism to change size and shape simultaneously. It is perhaps surprising to observe that the combined growth (Case D) image sequence much more closely resembles nominal growth (Case C) than regeneration (Case B). In fact, as early as time step 5000 of the image sequence, Case D and Case C appear qualitatively the same, even though the initial conditions (at time step 1) are entirely different. The similarity of the image sequences can be explained by examining the growth contributions plot, which shows that patterning growth term is most active early, approximately through time step 1000. In fact, the shape of the patterning growth curve for Case D is nearly identical to what was observed for Case B, but with a smaller peak amplitude and scaled to a shorter time scale (about five times faster completion of patterning as compared to Case B). The shortened duration of patterning growth is related to the choice of patterning growth coefficient C_p (which was reduced in Case B) and to the initial condition of Case D (which has half the width of the initial condition for Case B, such that the velocity-to-length ratio is increased in Case D).

At first glance, it appears that the growth contribution plot for Case D suggests some interaction between the isometric and patterning growth terms, since the isometric growth rate in Case D appears to dip for the first 1000 time steps as compared to Case C. This difference in the initial isometric growth can more simply be explained as a size effect, however, rather than an interaction. Since the total amount of isometric growth scales with the amount of material in the organism, and since the amount of simulated material is very low post amputation (in Case D), it should not be surprising that the volumetric growth rate is initially much lower in Case D than Case C.

Discussion

The verification tests described in the prior section, and in particular the Case D test, suggest that this simulation can provide a relevant model for simulation of *Xenopus* regeneration. In the Case D simulation, morphology is regenerated while the tail grows in size, a behavior seen in *Xenopus* regeneration. Importantly, the verification tests of the prior section also demonstrate that the simulation is stable and qualitatively well behaved. The patterning control switches off when the organism shape approaches the reference map (Case A). The isometric growth and volumetric terms perform as expected when active individually (Case B and Case C) and when active simultaneously (Case D). The smoothing control term was active in all cases, providing small adjustments to regularize shape (as visible in Case B in particular), but always resulting in a very small contribution to the overall growth of the simulated organism, so small that the contribution was only visible when magnifying the scale of the growth contribution plot (as in Case A).

The simulation does have limitations. First, the reinitialization process introduces slight irregularities, since reinitialization occurs only periodically (once every 20 time steps). The result is that the growth contribution plots can appear slightly choppy (as is visible in the saw tooth pattern for patterning growth in Case C and Case D). Reinitialization is also somewhat computationally intensive. As such, an alternative to reinitialization may be pursued in the future. Second, smoothing control effects make it difficult to generate sharp corners. Some modification to the smoothing control may be necessary in the future to allow sharp features to develop when desired (as in the tip of the simulated tail). Third, at small tail sizes, the discrete nature of the patterning control reference

map can introduce dithering when the reference map is rescaled. To mitigate this effect, we will consider alternate representations of the reference map in the future. Since, the current reference map is binary (with a one indicating a grid point inside the organism and a zero indicating a grid point outside), a floating point representation of the reference map would likely be helpful to reduce dithering that occurs when the reference map is scaled.

The intended application of our simulation tools is to examine control policies and sensing modalities that might be used during regenerative growth. Future studies will validate our simulation tools through direct comparison to biological studies of *Xenopus* tail regeneration. Also, we will augment our current simulations with new models of control and sensing with the goal of explaining biological observations about the impact of external factors (electrical, chemical, damage, etc.) on regenerative growth.

Conclusion

This algorithm creates a simplified abstraction of cell regeneration morphology, using level set methods and control regimes, that is a base module for a future framework to predictively link cell-level signaling to macroscopic patterning. This ultimate framework may provide insight into regeneration, cancer, and even birth defects. This algorithm reduces complex cellular interactions into body boundary movement using three control regimes – patterning control, isometric control, and smoothing control. Patterning control mimics regeneration at wound sites and acts on the body boundary. Isometric control mimics bulk growth of the organism with time, and smoothing control regularizes growth by reducing high curvature regions. Looking specifically at *Xenopus laevis* tail regeneration, this algorithm shows promise in predicting cell patterning on the macroscopic scale. Although this paper specifically discusses simulation of a *Xenopus* tail in two dimensions, the methodology is general enough to be applied to arbitrary morphologies in both two and three dimensions.

References

Bianconi, Eva, Allison Piovesan, Federica Facchin, Alina Beraudi, Raffaella Casadei, Flavia Frabetti, Lorenza Vitale, et al. 2013. "An Estimation of the Number of Cells in the Human Body." *Annals of Human Biology* 40 (6): 463–71. doi:10.3109/03014460.2013.807878.

Brakke, Kenneth A. 2015. *The Motion of a Surface by Its Mean Curvature. (MN-20)*. Princeton University Press.

Chernet, Brook T., Chris Fields, and Michael Levin. 2015. "Long-Range Gap Junctional Signaling Controls Oncogene-Mediated Tumorigenesis in Xenopus Laevis Embryos." *Biophysics* 5: 519. doi:10.3389/fphys.2014.00519.

C.S. Hogea, B.T. Murray, and J. A. Sethian. n.d. "Simulating Complex Tumor Dynamics from Avascular to Vascular Growth Using a General Level Set Method." *Journal of Mathmatical Biology*.

Evans, Lawrence C., and Joel Spruck. 1991. "Motion of Level Sets by Mean Curvature. I." *J. Differential Geometry* 33 (3): 635–81.

Faber, Jacob, and Pieter D. Nieuwkoop. 1967. *Normal Table of Xenopus Laevis (Daudin) : A Systematical and Chronological Survey of the Development from the Fertilized Egg till the End of Metamorphosis*. 2nd ed. Amsterdam : North-Holland Pub. Co.

Friston, Karl, Michael Levin, Biswa Sengupta, and Giovanni Pezzulo. 2015. "Knowing One's Place: A Free-Energy Approach to Pattern Regulation." *Journal of The Royal Society Interface* 12 (105): 20141383. doi:10.1098/rsif.2014.1383.

King, Ryan S., and Phillip A. Newmark. 2012. "The Cell Biology of Regeneration." *The Journal of Cell Biology* 196 (5): 553–62. doi:10.1083/jcb.201105099.

Kundu, Pijush K., Ira M. Cohen, and David R. Dowling Ph.D. 2011. *Fluid Mechanics*. 5th Edition. Academic Press.

Lobo, Daniel, Wendy S. Beane, and Michael Levin. 2012. "Modeling Planarian Regeneration: A Primer for Reverse-Engineering the Worm." *PLoS Comput Biol* 8 (4): e1002481. doi:10.1371/journal.pcbi.1002481.

Love, Nick R., Yaoyao Chen, Shoko Ishibashi, Paraskevi Kritsiligkou, Robert Lea, Yvette Koh, Jennifer L. Gallop, Karel Dorey, and Enrique Amaya. 2013. "Amputation-Induced Reactive Oxygen Species Are Required for Successful Xenopus Tadpole Tail Regeneration." *Nature Cell Biology* 15 (2): 222–28. doi:10.1038/ncb2659.

Osher, Stanley, and James A Sethian. 1988. "Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations." *Journal of Computational Physics* 79 (1): 12–49. doi:10.1016/0021-9991(88)90002-2.

Reid, Brian, Bing Song, and Min Zhao. 2009. "Electric Currents in Xenopus Tadpole Tail Regeneration." *Developmental Biology* 335 (1): 198–207. doi:10.1016/j.ydbio.2009.08.028.

Rejniak, Katarzyna A., and Alexander R. A. Anderson. 2011. "Hybrid Models of Tumor Growth." *Wiley Interdisciplinary Reviews: Systems Biology and Medicine* 3 (1): 115–25. doi:10.1002/wsbm.102.

Sethian, J. A. 1985. "Curvature and the Evolution of Fronts." *Communications in Mathematical Physics* 101 (4): 487–99. doi:10.1007/BF01210742.

Sethian, J.A. n.d. *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge Monographs on Applied and Computational Mathematics.

Sethian, James. 1984. "Turbulent Combustion in Open and Closed Vessels." *Journal of Computational Physics* 54 (3): 425–56. doi:10.1016/0021-9991(84)90126-8.

Suzuki, Makoto, Nayuta Yakushiji, Yasuaki Nakada, Akira Satoh, Hiroyuki Ide, and Koji Tamura. 2006. "Limb Regeneration in *Xenopus Laevis* Froglet." *The Scientific World Journal* 6: 26–37. doi:10.1100/tsw.2006.325.

Walker, D.C., G. Hill, S.M. Wood, R.H. Smallwood, and J. Southgate. 2004. "Agent-Based Computational Modeling of Wounded Epithelial Cell Monolayers." *IEEE Transactions on NanoBioscience* 3 (3): 153–63. doi:10.1109/TNB.2004.833680.

Zhang, Ying, and Michael Levin. 2009. "Particle Tracking Model of Electrophoretic Morphogen Movement Reveals Stochastic Dynamics of Embryonic Gradient." *Developmental Dynamics*, no. 238: 1923–35.

Learning Cassin's Vireo (*Vireo cassinii*) syntax through grammatical inference

Julio G. Arriaga[1], Richard Hedley[2], Edgar E. Vallejo[1] and Charles E. Taylor[2]

[1]Computer Science Dept., Tecnológico de Monterrey, Campus Estado de México
Atizapán de Zaragoza, Estado de México, 52926, México
[2]Dept. of Ecology and Evolutionary Biology, University of California, Los Angeles
Los Angeles, CA, 90095-1606, USA
elbuenjulius@gmail.com

Abstract

Birdsong may be regarded as a complex adaptive system. In this paper we study the relationship between complexity and consistency of an evolving model of Cassin's Vireo syntax, using a genetic algorithm to approximate a Minimal Consistent Deterministic Finite-state Automata (MCDFA) capable of accepting vocal sequences produced by birds of the study species. Our results imply that, despite the complex vocal behaviour of this species, the complexity of the model can be reduced considerably to encompass all of the positive samples while retaining the ability to exclude similar negative samples. These results suggest the existence of important regularities in the song sequences of this species.

Introduction

The study of birdsong as a science dates back to the 1950s when the first sound spectrographs became available, enabling researchers to objectively study and specify its structure with an unprecedent level of detail. Since then, much research have focused on deciphering its purpose, learning mechanisms, and meaning behind them.

There have been considerable efforts in the past towards understanding the structural rules that govern birdsong production (Honda and Okanoya, 1999; Berwick et al., 2011). However, due to the vast diversity of songbird species and the apparent structural differences in the songs they produce there is still no consensus on the limits of their complexity.

Similarly, birdsong have proven to be an excellent platform for exploring a variety of topics that are relevant to artificial life research, such as complexity, coevolution and sexual selection, among others (Taylor and Cody, 2015; Sasahara and Ikegami, 2003, 2004).

Our research is currently focused on the acoustic monitoring of different species of birds in several areas of the US and Mexico where they are abundant (Arriaga et al., 2013). Our long term goal is to understand the structure and function of birdsong. Particularly, the research described here aims at analysing the trade-off between complexity and consistency when learning a model representation of the syntactic rules governing birdsong structure of Cassin's Vireo (*Vireo cassinii*), a songbird possessing a complex vocal behaviour.

More precisely, we explore how the complexity of the model describing the syntax of CaVi song could be reduced while avoiding overgeneralisation. Toward that goal, we use a genetic algorithm to approximate a Minimal Consistent Deterministic Finite-state Automata (MCDFA) capable of accepting sequences produced by birds of the Cassin's Vireo species and rejecting a collection of artificially generated negative sequences.

The problem of finding a MCDFA from a set of examples S is $NP - Complete$ (Gold, 1978), further even finding an approximately small consistent DFA has also been proven to be not solvable in polynomial time (Pitt and Warmuth, 1989). Given these constraints, approaching the problem as an optimisation task using evolutionary computation is a natural approach as exhaustive methods are not tractable.

Previous research on the area by (Kakishita et al., 2009) has already tackled the problem of finding a minimised automata representation from observed song sequences, however this approach assumes the target representation is a k-reversible language. Although this assumption is backed by some biological plausibility, we consider exploring methods not bound by these a priori assumptions an interesting line of research.

Materials and Methods

Study Species

Cassin's Vireo (*CaVi*) is a small migratory songbird that breeds throughout western North America during the April-July period. Individuals are territorial, establishing and defending their breeding grounds from other males. Singing is exclusive to males and is primarily used during the breeding season (Goguen and Curson, 2002). In our studied population each male possesses a repertoire comprised of 51 ± 5 (mean $\pm$ standard deviation) highly stereotyped phrase-types, each lasting between 0.20 and 0.66 seconds (Hedley, 2016). Individuals sing persistently throughout the day, delivering phrases at varying rates: from more than one phrase per second during periods of high vocalisation production, to single phrases delivered between silence intervals of several seconds long (Hedley, 2016).

Field Recordings

All recordings were collected on private land five kilometres north of the town of Volcano in Amador County, California (10 S 706584 4262742) by Richard Hedley throughout the breeding season, from April 25 to June 28, 2013 and from May 5 to June 25, 2014. This period was deliberately chosen to account for possible seasonal variations in singing behaviour and to increase the likelihood of observing the majority of the temporal variability of the songs, as song output in the species is concentrated in the breeding season.

Recordings were made using a Marantz PMD 661 solid-state digital recording unit and a Sennheiser MKH20-P48 microphone with a Telinga parabolic reflector. Each recording session began when a bird was heard singing, and ended either when the bird stopped for a significant amount of time or flew away, becoming inaudible. Recordings were thus opportunistic in nature, without clear beginnings or endings in some instances.

Recording Annotation

Using the linguistics program Praat (Boersma and Weenink, 2014), each recording was subsequently annotated, identifying and categorising distinct phrase-types through visual inspection of their spectrograms. A unique two letter code (*aa*, *ab*, *ac*, etc.) was assigned to each phrase-type and a spectrogram image of the phrase-type was added to a reference catalogue for future phrase identification. Figure 1 shows spectrogram representations of three distinct phrase-types, as recorded from two different individuals in our study population.
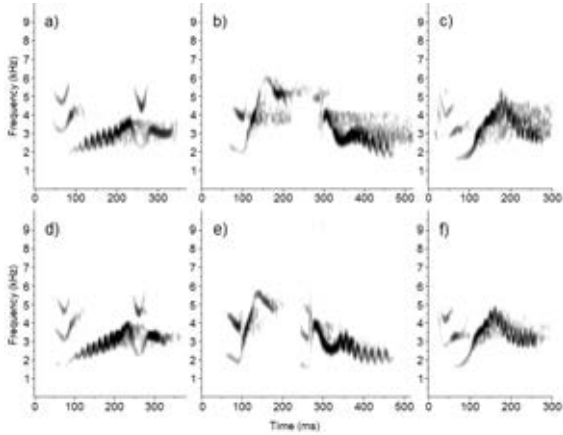

Figure 1: Spectrogram images of three distinct phrase-types, recorded from two different individuals. Panels a) and d) illustrate phrase-type *ai*; panels b) and e) illustrate phrase-type *bj*; and panels c) and f) illustrate phrase-type *br*. The first three exemplars (a-c) were recorded from the *Meadow* individual, while the last three (d-f) were recorded from the *Sign* individual.

Phrases belonging to each phrase-type were remarkably stereotyped, such that each phrase was readily assigned a phrase-type label by visually inspecting a spectrogram of the signal. We determined that this method of phrase identification was objective by subsequent classification with a variety of machine learning methods (Tan et al., 2015; Kantapon et al., 2015); these agreed almost perfectly with the human identification. This method has been found to be more than 99% accurate for annotation of phrase-types for the purpose of syntactic analysis (Hedley, 2016). We identified a total of 128 distinct phrase-types among the 15 different individuals in our study. This group of 15 birds represents the entire male population of *CaVi* individuals living within the 1-square kilometre valley of our study site.

Recordings were also tagged with the designated name of the individual bird which produced it. This was done by the recordist through means of visual identification and territory analysis, and was later verified through the use of an ensemble of Machine Learning Methods (Arriaga et al., 2016). Table 1 shows an example of some recordings as annotated phrase sequences.

Table 1: Example of three recordings as sequences of phrases after being manually annotated. Each two-letter code corresponds to a distinct phrase-type identified within the bird species.

Individual	Annotated Recording
agbk	ah, ai, ah, aj
	ah, ai, en, aj
	fg, em, cg, cr, fq
	ai, en, ai, aj, en, en, ak
	fg, em, ck, fg, ca, fg, em

Data Preparation

Recordings were divided into phrase sequences by grouping phrases sung with no more than ten second pauses between them. All phrase sequences were then grouped by the individual which produced them. Since all our data consists of observed sequences, all sets consist entirely of positive samples (S_+). Negative samples are fundamental to avoid excessive generalisation of the model. For this purpose we artificially generated a negative sample S_{-_1} of phrase sequences by randomly sampling phrases from a collection of all observed phrase-types for each individual. The sequences produced are thus composed of uniformly distributed phrase-types, and phrase-type combinations or *n-grams*. Previous research on the composition of observed sequences has shown non-random patterns of sequential distribution of phrase-types and *n-grams* (Hedley, 2016). Although we cannot guarantee all of these simulated phrase sequences are impossible to produce by the birds' syntax, their statistical composition make them highly unlikely. Ad-

ditionally, a second set of negative examples S_{-_2} was generated by randomly sampling phrases from a collection of all observed phrase-types for each individual maintaining the same phrase-type frequency distribution as the observed data. Figure 2 (a) and (d) show the total number of occurrences of distinct phrase-types and bigrams ordered by rank for the observed data of individual *agbk* compared to those in the simulated data S_{-_1} (b) and (e), and S_{-_2} (c) and (f).

Minimal Consistent Deterministic Finite-State Automata

A Deterministic Finite-State Automata (DFA) is a quintuple $A = \langle \Sigma, Q, q_\lambda, \mathbb{F}, \delta_N \rangle$ where Σ is an alphabet, Q is a finite set of states, $q_\lambda \in Q$ is the initial state, $\mathbb{F}$ denotes the set of final states (both accepting $\mathbb{F}_\mathbb{A}$ and rejecting $\mathbb{F}_\mathbb{R}$), and $\delta_N : Q \times (\Sigma) \to Q$ is a transition function.

The minimal DFA consistent with a given sample S is simply a DFA with at most *opt* states that accepts all S_+ positive examples and rejects all S_- negative samples. Where *opt* is the least number of states possible for the given sample. Given the hardness of the problem, an exhaustive search of all possible solutions is not practical for most cases.

Genetic Algorithm

A genetic algorithm was used to explore possible models for the observed data for each individual. All experiments were performed in the Python programming language (van Rossum and Drake, 2001), using the NetworkX (Hagberg et al., 2008) and DEAP (Fortin et al., 2012) packages. Positive samples (S_+) were used to build a Maximal Canonical Automata (MCA), a star shaped Non-Deterministic Finite-state Automata (NFA) with one branch for each sequence in S_+.

In other words, a MCA is an automata that accepts exclusively the sequences observed in the data. See Figure 3.

A MCA is also structurally complete with respect to the positive sample, as every transition and acceptor state is used at least once when parsing the observed strings. We thus defined our search space as the subset of all automata consistent with S which are structurally complete. Since we are interested in finding a DFA, non-deterministic transitions were removed from the MCA. The resulting automata is equivalent to a Prefix Tree Acceptor (PTA). See Figure 4.

Population

Individuals were defined as partitions over the PTA in order to explore the search space. A partition Π of a DFA A is defined as $A/\Pi = \left(\Sigma, \bar{Q}, \bar{q_\lambda}, \bar{\mathbb{F}}, \bar{\delta_N} \right)$

- $\bar{Q} = Q/\Pi$ is the set of equivalence classes defined by the partition Π.

- $\bar{\delta_N}$ is a function $\bar{Q} \times \Sigma \to Q$ such that $\forall \bar{q}, \bar{q}' \in \bar{Q}, \ \forall a \in \Sigma, \bar{q}' \in \bar{\delta_N} (\bar{q}, a) \ \ if_{def} \ \exists q \in \bar{q} \ \exists q' \in \bar{q}' \ : \ q' \in \delta_N (q, s)$.
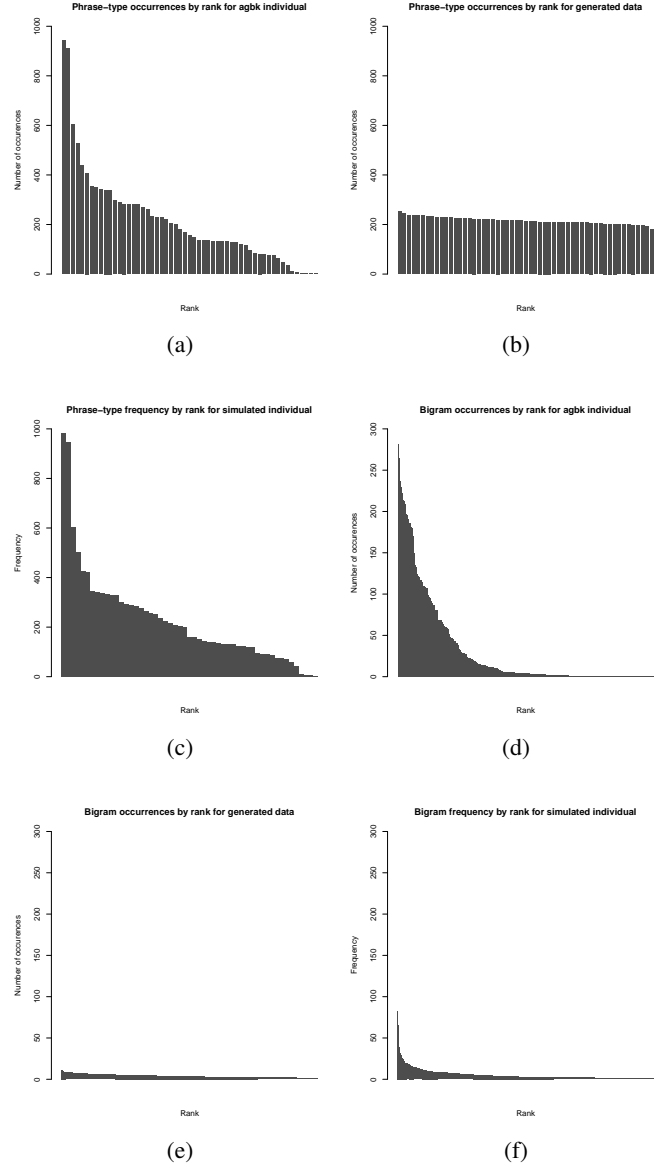
Figure 2: Phrase-type and bigram occurrences sorted by rank out of sequences of 11838 phrases. (a) Observed phrase-types in *agbk* samples; (b) Generated phrase-types in negative samples from uniform distribution (S_{-_1}); (c) Generated phrase-types in negative samples from same distribution as observed samples S_{-_2}. (d) Observed bigrams in *agbk* samples; (e) Generated bigrams in negative samples from uniform distribution (S_{-_1}). (f) Generated bigrams in negative samples from same distribution as observed samples (S_{-_2}).

Figure 3: Maximal Canonical Automata derived from sample sequences shown in Table 1. Nodes with a thicker line denote acceptor states.
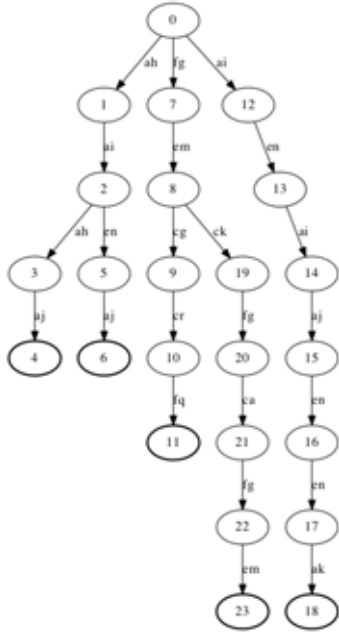
Figure 4: Prefix Tree Acceptor derived from sample sequences shown in Table 1. Nodes with a thicker line denote acceptor states.

- $\bar{q}_\lambda$ is the initial state.

- $\bar{\mathbb{F}}$ is the set of final states.

In short, given a partition $\Pi = \langle P_0, P_1, ..., P_k \rangle$, in which $P_i, i \leq k$ represents a group of states from Q, the automata a/Π is the result of mer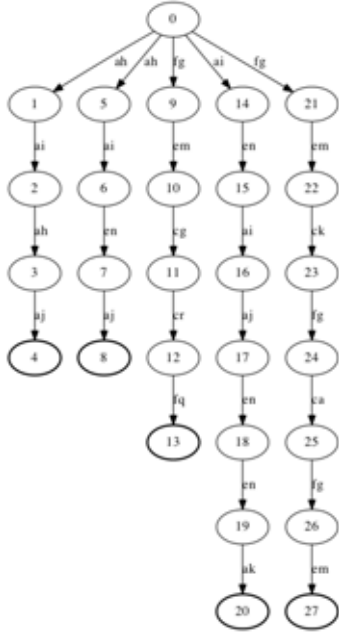ging together all of the states in each P_i. Individuals were encoded as a string $\langle x_0, x_1, ..., x_n \rangle$, where x_i represents the *group* to which the corresponding node in the PTA belongs and n is equal to the total number of nodes in the PTA. For example, the partition [[2, 21], [20], [7], [0], [14, 17], [15], [19], [6], [8], [13], [9, 10, 11, 18], [5, 16], [3], [12], [4], [1]] is encoded by the string $\langle 3, 22, 0, 16, 19, 14, 8, 2, 9, 13, 13, 13, 17, 12, 4, 6, 14, 4, 13, 7, 1, 0 \rangle$. See Figure 5.

Figure 5: Automata derived from applying partition $\langle 3, 22, 0, 16, 19, 14, 8, 2, 9, 13, 13, 13, 17, 12, 4, 6, 14, 4, 13, 7, 1, 0 \rangle$ to the automata shown in Figure 4. Nodes with a thicker line denote acceptor states.

Thus, the initial population was generated as a collection of 100 random partitions over the PTA.

Genetic Operators

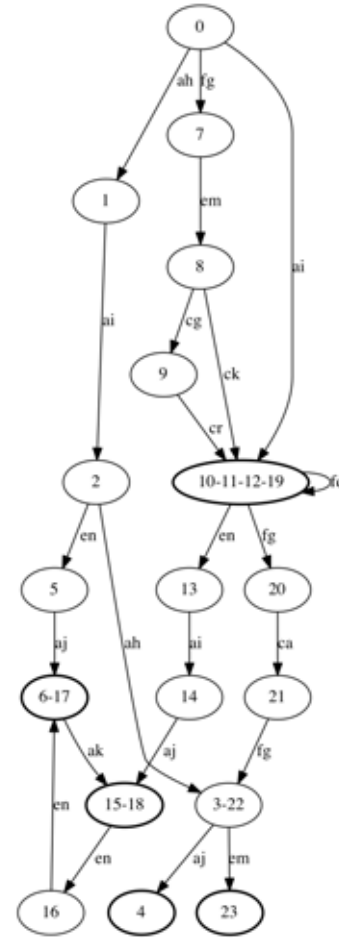Crossover: each generation, pairs of individuals from the population are chosen through tournament selection with a size of three and a probability of 95% and reproduced by swapping their chromosomes at a randomly selected crossover point. Mutation: individuals in the population change the partition group to which one of their nodes belongs with a probability of 1%.

Both of these genetic operators are guaranteed to produce valid individuals since all partitions of the PTA represent a valid solution for the formulated problem.

Fitness Function

Individuals are evaluated by a combined measure of the underlying DFA they represent, given by its complexity and inconsistency. The goal of the GA is to minimise this combined score.

The complexity of a DFA was measured following the Minimum Description Length (MDL) principle. The MDL principle states that the best solution for a given set of data is that which minimises the encoding of the data (Rissanen, 1978); when applied to grammatical inference, this means that the best hypothesis for some observed data S_+ is that which minimises the encoding of the grammar and its parsing of the data (De la Higuera, 2010). Specifically, we defined our complexity function for a DFA A as described in Equations 1 and 2.

$$complexity = |Q| \times |\Sigma| + \sum^{s} d(q_i) \qquad (1)$$

$$d(q) = \begin{cases} \log\left(|\{a \in \Sigma : \delta(q, a) \text{is defined}\}|\right) & \text{if } q \in \mathbb{F}_R \\ \log\left(1 + |\{a \in \Sigma : \delta(q, a) \text{is defined}\}|\right) & \text{if } q \in \mathbb{F}_A \end{cases} \qquad (2)$$

In Equation 1, $|Q|$ denotes the number of states of A; $|\Sigma|$ denotes the size of the alphabet, and $\sum^{s} d(q_i)$ the sum of the respective complexities $d(q_i)$ of each state q_i visited while parsing a sample s. Equation 2 represents the number of possible decisions presented in each state q as signified by the number of exiting edges for the state, plus one when the state is an acceptor, as ending parsing (and thus accepting the string s) represents an extra option. Note that states with only one option (i.e. only one exiting edge or an acceptor state with no exiting edges) will have a $d(q) = log(1) = 0$.

The inconsistency of a DFA is measured as the proportion of negative samples it accepts.

$$inconsistency = \frac{|B_{acc}|}{|B_-|} \qquad (3)$$

In Equation 3, $|B_{acc}|$ denotes the number of negative samples accepted and $|B_-|$ the total number of negative samples presented.

Two fixed-length random subsamples from S_+ and S_- are thus used each generation to evaluate the complexity and inconsistency respectively of the individuals in the population.

Results

Our approach proved to be successful at continuously decreasing the complexity of the target solution while also lowering the proportion of negative samples wrongly accepted. Figures 6 and 7 (a) and (b) show the box plot distributions of the complexity and inconsistency scores for the possible solutions population at each generation. For both the *gay* and *meadow* individuals a steady decline in inconsistency can be appreciated, however, the decrease in complexity is relatively low from generation to generation, making it necessary to either significantly increase the size of the initial population or the total number of generations the algorithm is allowed to run. Each of these options imply a significant overhead in computational cost due to the required manipulation of automata with a high number of nodes.

To overcome this we instead opted for an alternative iterative approach. Starting with an initial population of size $p = 100$, after $n = 10$ generations the best found solution was used as the starting point for generating a new population of a subsequent run, or *round*, of the GA, with an initial population of $p = p + 50$. After $n = n + 5$ generations, the best found solution is once again used as the starting point for another round of the GA. These values were determined both empirically and to match the available computational resources.

Figures 6 and 7 (c) and (d) show the box plot distributions of the complexity and inconsistency scores for each generation of the second round. Both scores follow a similar tendency as the previous round, with complexity scores improving slightly at each generation, while inconsistency scores show a speedier improvement. Figures 6 and 7 (e) and (f) show the results for a third round. Although the overall best possible solutions belonged to populations in this round, several individuals in those populations suffered over generalisation problems, as demonstrated by the increased ratio of incorrectly accepted negative samples and significantly low complexities.

A drawback to this approach is that it limits the search space on each iteration, from the lattice over the PTA to the lattices over the partial solutions, in doing so, better solutions might become completely inaccessible to the search procedure. However, it also offers the advantage of arriving at better solutions faster, expending less computational resources.

Figures 8 (a) and (b) show how the most significant decreases in the number of states for the best found solution were a direct result of the start of a new round (denoted by the dotted lines). Likewise, Figures 8 (c) and (d) show the same behaviour for complexity scores. The same cannot be said for inconsistency scores shown in Figures 8 (e) and (f),

(a)   (b)   (a)   (b)

(c)   (d)   (c)   (d)

(e)   (f)   (e)   (f)

Figure 6: Box plot distribution of complexity and inconsistency scores across all individuals in population throughout each generation after one (a and b), two (c and d), and three (e and f) rounds *ayo* individual using negative sample S_{-1}. The best solution found for each generation is marked with an x.
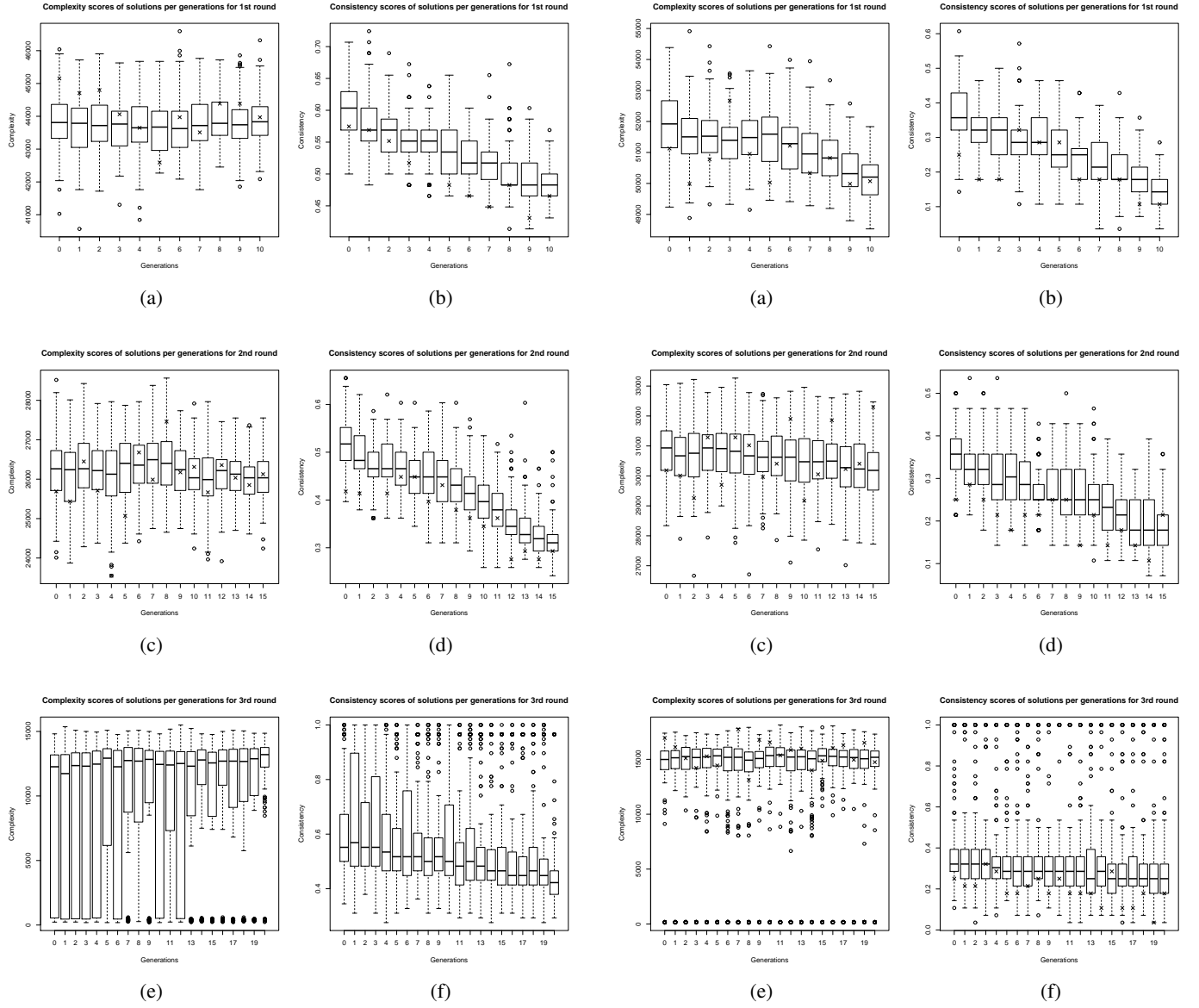
Figure 7: Box plot distribution of complexity and inconsistency scores across all individuals in population throughout each generation after one (a and b), two (c and d), and three (e and f) rounds *meadow* individual using negative sample S_{-2}. The best solution found for each generation is marked with an x.

where rounds one and two follow a more continuous decline until round three, where the over generalization of the potential solutions causes a sudden increase.

Discussion and Future Work

In this work, we explored the capabilities of a genetic algorithm coupled with the minimum description length principle to model an approximate representation of the syntactic structure governing *CaVi's* songs. Moreover, we analysed the results in terms of complexity and inconsistency of the evolved models.

Regarding the experiments comprising random negative examples, by sampling both phrases and their position in the sequence from uniform distributions, we observed a considerable reduction in complexity as a results of the evolutionary process. The inconsistency of the model also reduced considerably, often maintaining this tendency even with models of reduced complexity.

However, the syntactic models that were evolved using negative examples where phrases were sampled from the phrase distribution of the positive examples at random positions, began suffering from overgeneralisation as the complexity decayed.

The choice of using a DFA as a target representation was based on its simplicity, as there is no definite knowledge of the underlying complexity of the CaVi's syntax. The inability of DFAs to improve consistency when there is some degree of similarity between the positive and negative examples, suggests that more complex models such as probabilistic DFAs, pushdown automata or linear bounded automata, should be considered in the future. Similarly, this method uses exclusively a genetic algorithm for the search of possible solutions, however other machine learning techniques, such as tabu search, could be used instead. In the future we will experiment with different combinations of target representations and search mechanisms and compare their strengths and weaknesses among themselves and other existing methods.

Several obstacles hinder the study of this problem. Data availability remains a major concern, as the process required for gathering and preprocessing is extremely time consuming. This lower availability of data complicates the usage of already established techniques for grammatical inference as the field is more oriented towards human languages, for which bigger data sets are easier to come by.
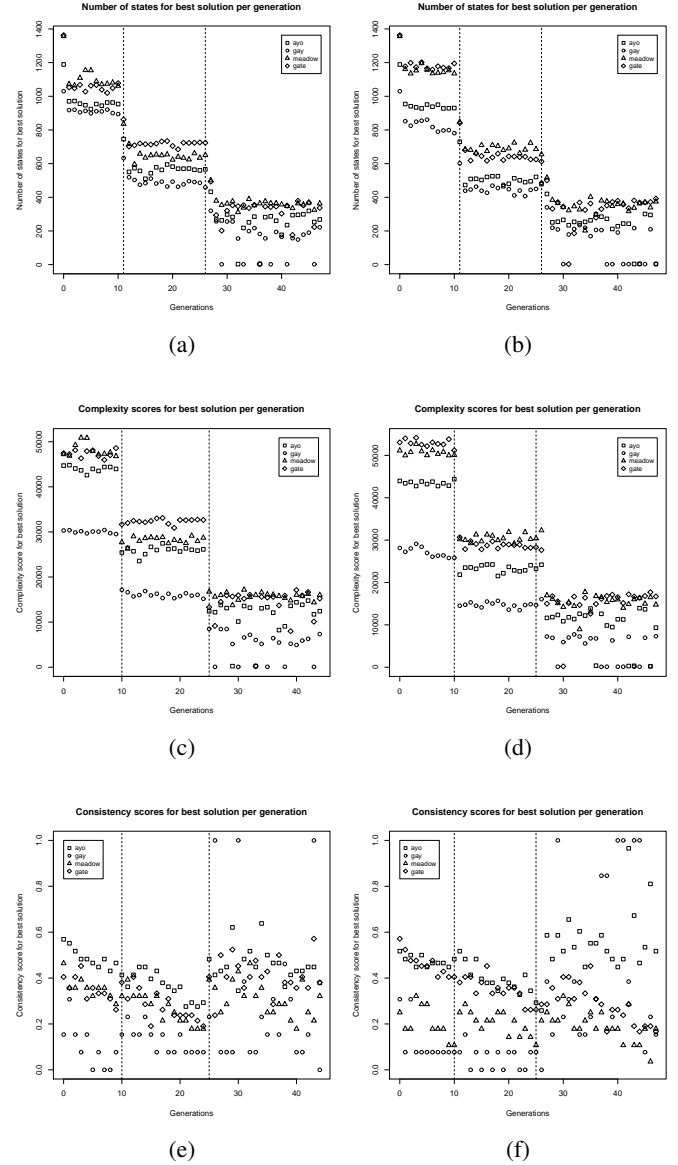
Acknowledgments

Figure 8: Number of states in best solution per generation for four different individual: *ayo, gay, meadow* and *gate*. Dotted lines along *y* axis represent different rounds of the GA. (a), (c), and (e) used negative samples S_{-1}; (b), (d), and (f) used negative samples S_{-2}.

References

Arriaga, J. G., Kossan, G., Cody, M. L., Vallejo, E. E., and Taylor, C. E. (2013). Acoustic sensor arrays for understanding bird communication. identifying cassin's vireos using svms and hmms. In *Advances in Artificial Life, ECAL*, volume 12, pages 827–828.

Arriaga, J. G., Sanchez, H., Vallejo, E. E., Hedley, R., and Taylor, C. E. (2016). Identification of cassin? s vireo (vireo cassinii) individuals from their acoustic sequences using an ensemble of learners. *Neurocomputing*, 175:966–979.

Berwick, R. C., Okanoya, K., Beckers, G. J., and Bolhuis, J. J. (2011). Songs to syntax: the linguistics of birdsong. *Trends in cognitive sciences*, 15(3):113–121.

Boersma, P. and Weenink, D. (2014). Praat: doing phonetics by computer. Computer program.

De la Higuera, C. (2010). *Grammatical inference: learning automata and grammars*. Cambridge University Press.

Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A., Parizeau, M., and Gagné, C. (2012). DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175.

Goguen, C. B. and Curson, D. R. (2002). *Cassin's vireo: Vireo cassinii*. American Ornithologists' Union.

Gold, E. M. (1978). Complexity of automaton identification from given data. *Information and control*, 37(3):302–320.

Hagberg, A. A., Schult, D. A., and Swart, P. J. (2008). Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy2008)*, pages 11–15, Pasadena, CA USA.

Hedley, R. W. (2016). Composition and sequential organization of song repertoires in cassin's vireo (vireo cassinii). *Journal of Ornithology*, 157(1):13–22.

Honda, E. and Okanoya, K. (1999). Acoustical and syntactical comparisons between songs of the white-backed munia (lonchura striata) and its domesticated strain, the bengalese finch (lonchura striata var. domestica). *Zoological Science*, 16(2):319–326.

Kakishita, Y., Sasahara, K., Nishino, T., Takahasi, M., and Okanoya, K. (2009). Ethological data mining: an automata-based approach to extract behavioral units and rules. *Data Mining and Knowledge Discovery*, 18(3):446–471.

Kantapon, K., Tan, L. N., Alwan, A., and Taylor, C. E. (2015). A robust automatic phrase classifier using dynamic time-warping with prominent region identification. *Proceedings of ICASSP*, In press.

Pitt, L. and Warmuth, M. K. (1989). The minimum consistent dfa problem cannot be approximated within and polynomial. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 421–432. ACM.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, 14(5):465–471.

Sasahara, K. and Ikegami, T. (2003). Coevolution of birdsong grammar without imitation. In *Advances in artificial life*, pages 482–490. Springer.

Sasahara, K. and Ikegami, T. (2004). Song grammars as complex sexual displays. In *Artificial Life IX: Proceedings of the 9th International Conference on the Simulation and Synthesis of Living Systems*, pages 194–199.

Tan, L. N., Alwan, A., Kossan, G., Cody, M., and Taylor, C. E. (2015). Dynamic time warping and sparse representation classification for birdsong phrase classification using limited training data. *Journal of the Acoustic Society of America*, In press.

Taylor, C. E. and Cody, M. L. (2015). Bird song: a model complex adaptive system. *Artificial Life and Robotics*, 20(4):285–290.

van Rossum, G. and Drake, F. (2001). Python reference manual. Technical report, PythonLabs, Virgina, USA.

Propagation of rhythmic dorsoventral wave in a neuromechanical model of locomotion in *Caernohabditis elegans*

Eduardo J. Izquierdo and Randall D. Beer

School of Informatics and Computing
Cognitive Science Program, Indiana University Bloomington
edizquie@indiana.edu

With 302 neurons and a fully reconstructed connectome, *Caernohabditis elegans* is an ideal candidate organism to study how behavior is grounded in the interaction between an organism's brain, its body, and its environment. Since nearly its entire behavioral repertoire is expressed through movement, understanding the neuromechanical basis of locomotion is especially critical as a foundation upon which analyses of all other behaviors must build. In this extended abstract, we report on the evolution and analysis of an integrated neuromechanical model of forward locomotion.

C. elegans locomotes in an undulatory fashion, generating thrust by propagating dorso-ventral bends along its body. How the rhythmic patterns are generated and propagated is not yet understood. We focus here on the propagation of the dorsoventral body bend along the body.

To date there have been a handful of models of forward locomotion (see reviews by Gjorgjieva et al. (2014) and by Cohen and Sanders (2014)). However, recent experimental analysis of the structure of ventral cord circuitry (Haspel and O'Donovan, 2011) and the effect of local body curvature on nearby motor neurons (Wen et al., 2012) undermine some of the assumptions of these models. Furthermore, all current models have assumed specific answers to how the rhythmic movement is propagated, with little systematic exploration of the possibilities.

First, we reconstructed a biomechanical model of the worm's body and musculature from published descriptions (Boyle et al., 2012). The complete physical model consists of a set of 147 stiff, highly nonlinear differential-algebraic equations (Fig. 1A). Second, we developed a neural model of the ventral nerve cord subcircuit associated with forward locomotion, comprising four main classes of motor neurons: 12 neurons of class VB and VD, and 6 neurons of class DB and DD (Fig. 1B), separated into 6 repeating neural units derived from a statistical analysis of the connectome (Haspel and O'Donovan, 2011). Following previous work, neurons were modeled as passive, isopotential components, and the model includes chemical and electrical synapses (Izquierdo and Beer, 2013). Third, we incorporated stretch-receptors innervating B-motoneurons from

anterior body segments (Fig. 1B), based on recent findings (Wen et al., 2012). The generation of the rhythmic wave was modeled as originating in the head.
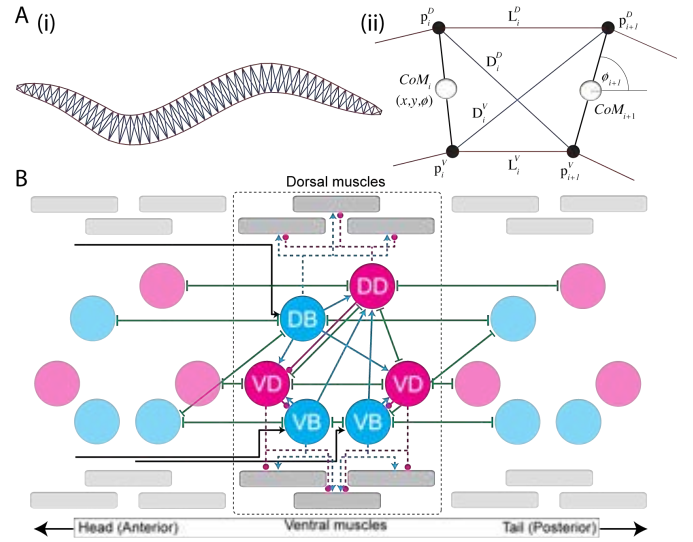
Figure 1: Neuromechanical model. (A) Complete physical model (i) and one of 49 individual segments (ii) adapted from (Boyle et al., 2012). (B) One of 6 repeating neuromuscular units, derived from a statistical analysis of the connectome (Haspel and O'Donovan, 2011). Each unit includes one dorsal and two ventral B- (cholinergic, blue) and D-class (GABAergic, magenta) motor neurons that connect to muscles (gray) on each side. The model includes all chemical synapses (blue excitatory and magenta inhibitory), gap junctions (green), and neuromuscular junctions (dashed). Additionally, B-class neurons receive stretch-receptor input from anterior muscles (black) (Wen et al., 2012).

Altogether, the model included 21 unknown electrophysiological parameters. An evolutionary algorithm was used to determine values of the unknown parameters that optimized behavioral performance. Solutions were evaluated on how closely they matched the speed of the worm on agar. We ran 100 evolutionary runs and consistently found elec-

trophysiological configurations that produced realistic control of movement when coupled to the biomechanical model of the body, and situated in a simulated agar environment.

Each successful search produced a distinct set of parameter values, leading to an ensemble of models that are consistent with the known biological constraints. The focus of our analysis was first to identify different possible classes of solutions through the exploration of electrophysiological configurations that produce realistic control of movement. The second part of the analysis was to understand the operation of the highest-performing exemplars of each class. We use this insight to propose experiments on the organism that test the hypotheses generated by the different classes.

In all evolved solutions, forward movement is produced with each body region alternating between positive and negative curvature, and bands of curvature propagating from head to tail as shown in a kymogram (Fig 2A). Emergent properties of the evolved networks reproduced key experimental observations that they were not designed to fit, including the curvature profile of the body's movement (i.e., curvatures near the head larger than curvatures near the tail) and the wavelength of the propagating wave. This suggests that the model may be operating according to principles similar to those of the biological network.

We analyzed the properties of the entire ensemble of solutions using a number of different techniques, including neuron recordings (Fig. 2B), neural and behavioral manipulations, and lesion studies. An examination of the ensemble revealed two broad classes of solutions: some where the D-class motoneuron did not play a role in forward locomotion and some where it did. All of the evolved solutions relied primarily on stretch-reception. Further analysis of the operation of these networks reveals the roles that the individual neural, synaptic and proprioceptive components of this system play in propagating and coordinating rhythmic undulatory waves from head to tail during locomotion.

C. elegans offers an unique opportunity to obtain a complete systems-level understanding of a locomotory circuit. As we better understand the operation of the ensemble of integrated neuromechanical model of locomotion, the insights will be used to propose novel experiments on the living organism that test the hypotheses generated by the different classes of solutions. The results of such neurobiological experiments can be used to constrain subsequent iterated optimizations, ultimately improving our understanding of the biological system, and more generally the generation of behavior in a coupled brain-body-environment system.

References

Boyle, J., Berri, S., and Cohen, N. (2012). Gait modulation in *C. elegans*: an integrated neuromechanical model. *Frontiers in Computational Neuroscience*, 6(10).
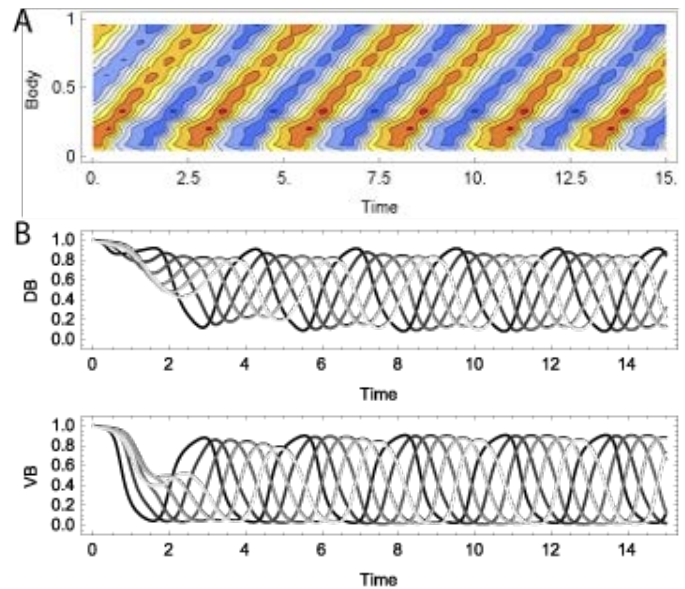

Figure 2: Characterizing evolved solutions. (A) Kymogram of time-varying curvature illustrating retrograde bending waves along the simulated worm (head = 0; tail = 1) responsible for forward movement. Wavelength and frequency in the model is similar to what has been observed in worm. (B) Neural activity in the Dorsal (top) and Ventral (bottom) B-class motoneuron for each of the six different neural units along the body (lighter shades of gray represent neurons in units closer to tail). Neural traces in the evolved circuits illustrate: (i) rhythmic patterns that are propagated anteriorly through the neural units with a phase lag, and (ii) anti-phase patterns of activity in dorsal and ventral units.

Cohen, N. and Sanders, T. (2014). Nematode locomotion: dissecting the neuronalenvironmental loop. *Current Opinion in Neurobiology*, 25(4):99–106.

Gjorgjieva, J., Biron, D., and Haspel, G. (2014). Neurobiology of *Caenorhabditis elegans* locomotion: where do we stand? *BioScience*, 64(6):476–486.

Haspel, G. and O'Donovan, M. (2011). A perimotor framework reveals functional segmentation in the motoneuronal network controlling locomotion in *Caenorhabditis elegans*. *Journal of Neuroscience*, 31(41):14611–23.

Izquierdo, E. J. and Beer, R. D. (2013). Connecting a Connectome to Behavior: An Ensemble of Neuroanatomical Models of *C. elegans* Klinotaxis. *PLoS Computational Biology*, 9(2).

Wen, Q., Po, M., Hulme, E., Chen, S., Liu, X., Kwok, S. W., Gershow, M., Leifer, A. M., Butler, V., Fang-Yen, C., et al. (2012). Proprioceptive coupling within motor neurons drives *C. elegans* forward locomotion. *Neuron*, 76(4):750–761.

A 3D Multiscale Model of Chemotaxis in Bacteria

Andrew Wu[1], Timothy Davison[1], and Christian Jacob[1,2]

[1]Department of Computer Science, University of Calgary, Calgary, AB, Canada T2N 1N4
[2]Department of Biochemistry & Molecular Biology, University of Calgary, Calgary, AB, Canada T2N 4N1
cjacob@ucalgary.ca

Abstract

We present an interactive, agent-based, multi-scale 3D model of a colony of *E. coli* bacteria. We simulate chemical diffusion on an agar plate which is inhabited by a colony of bacterial cells. The cells interact with a discrete grid that models diffusion of attractants and repellents, to which the cells react. For each bacterium, we simulate its chemotactic behaviour, making a cell either follow a gradient or tumble. Cell propulsion is determined by the spinning direction of the motors that drive its flagella.

In an agent-based model, we have implemented the molecular elements that comprise the two key chemotactic pathways of excitation and adaptation, which, in turn, regulate the motors and influence a cell's movement through the agar medium. We show four interconnected model layers that capture the biological processes from the colony layer down to the level of interacting molecules.

Introduction

We have implemented a model of a colony of bacterial cells, which we can visualize and interact with at four distinct, yet computationally interconnected levels (Fig. 1). At the "naked eye" level we model the gradient of a diffusing chemical signal, similar to what one observes in a laboratory by looking at an agar plate inhabited by a bacterial colony. Once we zoom closer into the plate, the colony of bacteria becomes visible, which reflects the simulated behaviours of cell clusters. Picking one of the cells transitions to a close-up view of an individual bacterium, with its flagella propelling it through the medium. As the last model level, we can navigate into a bacterium's cytoplasm, where we have implemented the molecular signalling pathways that drive chemotaxis.

E. coli and Chemotaxis

The prokaryotic cell we have modelled is known as *Escherichia coli*. Most strains of *E. coli*, as it is known for short, are harmless. We have billions of these bacteria naturally residing within our intestinal tracts (Zimmer, 2009). *E. coli* has been at the centre of many biological discoveries due

Eye view: agar plate	Colony view: cell cluster
Cell view: *E. coli*	Molecular view: cytoplasm
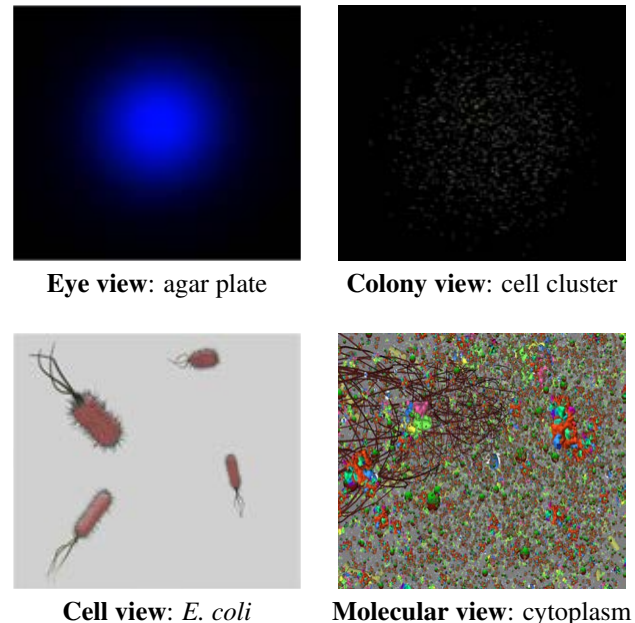
Figure 1: Snapshots of the four layers of resolution within our multi-scale *E. coli* model.

to its ease of growth and adaptability to different conditions and manipulation of its genome (Berg, 2004).

Chemotaxis is a universal attribute of motile cells and organisms. It is the mechanism that dictates their movement in the presence of a stimulus (Wadhams and Armitage, 2004). The stimulus—often chemical (hence "chemo")—is either an attractant or a repellent (Berg, 2004). A cell like *E. coli* moves in the direction of the higher gradient towards an attractant source (Fig. 8A), thereby exhibiting a positive chemotactic response. Correspondingly, chemo-repellents cause the organism to turn away from the stimulus, thereby exhibiting a negative chemotactic response (Fig. 8B). Starting from a center point, a typical *E. coli* colony expands in an elliptical pattern known as chemotactic rings (Fig. 2).

Figure 2: A) A tryptone soft agar plate in which motile cells can swim through water-filled tunnels in the agar. Two chemotactic colonies are shown. As the cells grow, they establish attractant gradients as they consume energy sources saturated in the agar. Printed with permission from Dr. John S. Parkinson Lab. B) Simulated chemotactic gradient in the agar layer. C) Closeup of the chemotactic ring formed by the simulated bacteria colony.
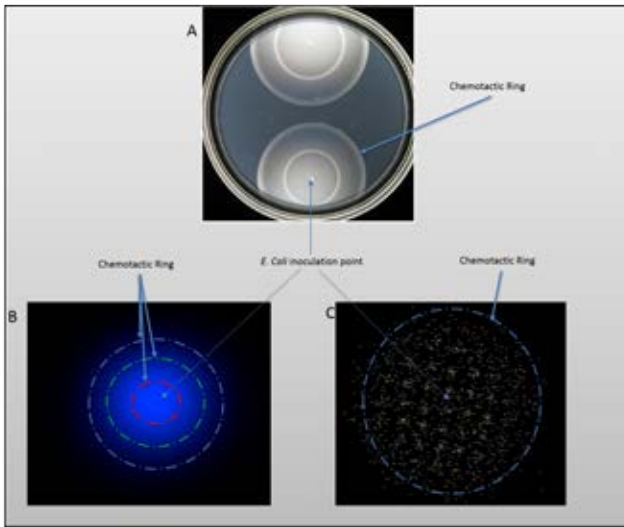
Chemotaxis Receptors

Bacteria use specific receptors to recognize chemical stimulants in their environment (Adler, 1966a,b, 1969, 1973; Adler et al., 1973). Five different receptors known as the methyl-accepting chemotaxis proteins (MCPs) play a key role in the signalling pathways (Berg, 2004): Tsr, Tar, Tap, Trg, and Aer. Each MCP detects a different chemical. MCPs are usually bundled into clusters at the poles of a bacterium (Adler, 1969; Sourjik, 2004). For simplicity, in our model we include one *generic receptor* which subsumes all MCP properties.

Chemotaxis Pathways

E. coli's chemotactic pathway is comprised of two distinct networks (Hauri and Ross, 1995): (1) the Signal Transduction Cascade, which leads to an *excitation* reaction, and (2) the Methylation Response, which results in *adaptation* (Berg, 2004; Hauri and Ross, 1995; Wadhams and Armitage, 2004). This chemotactic behaviour is a series of runs and tumbles the bacterium performs during its life span.

A bacterium is propelled by flagella filaments (Fig. 3) that extend from its cellular membrane (Alon, 2007). Each flagellum is controlled at its base by a motor inside the cell membrane. The direction in which the motors rotate determines whether the bacterium "runs" or "tumbles". A *run* is defined by the bacterium swimming in a forward mo-
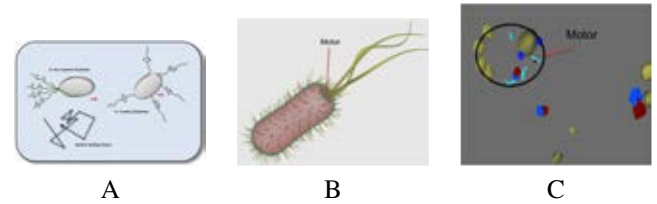

Figure 3: A) When the flagella filaments rotate counterclockwise, they bundle together and propel the bacterium forward. A clockwise motor rotation makes the bacterium tumble, as the flagella bundles break up. B) Location of the motor unit in our *E. coli* model. C) The motor complex inside the modelled cytoplasm.

Protein	Role
CheA	Phosphorylation protein
CheB	De-methylation protein
CheY	Signal transmission protein
CheZ	De-phosphorylation protein
CheR	Methylation protein
CheW	Binds with CheA to form CheAW complex

Table 1: Proteins and their roles in the chemotaxis pathway

tion. When the motor rotates counter-clockwise, the flagella bundle together to propel the bacterium forward (Berg, 2004). During the *tumble* phase, the motor rotates clockwise (Berg, 2004), which causes the flagella to break from the bundle, resulting in a random change of direction. In the absence of any stimulus, runs and tumbles alternate, which leads to a random walk for the bacterium.

In the presence of a stimulus—whether it be an attractant or repellent—the rotation bias of the flagella is affected by the stimuli. Molecules of a stimulus are picked up by the receptors protruding from the cell membrane. Such a stimulus then triggers the excitation and adaptation response. *E. coli* is constantly comparing the concentration of its current location to its previous location, thus implementing a short-term "memory" that compares present and past information (Segall et al., 1986).

Table 1 summarizes the proteins involved in the chemotaxis signalling pathways in *E. coli*'s cytoplasm. Each protein plays a specific role in one of the two response networks.

Excitation Response: Signal Transduction Cascade
The excitation response directly affects the motion of the bacterium, where a series of signals is transferred downstream from the receptor to a flagellum motor (Fig. 4A). To keep our model simple, we assume that receptors can only be active or inactive. CheAW, a complex formed by CheA and CheW, is bound to the receptor end inside the cell membrane. With an increase in attractant concentration, the receptor becomes inactive, thus increasing CheY concen-
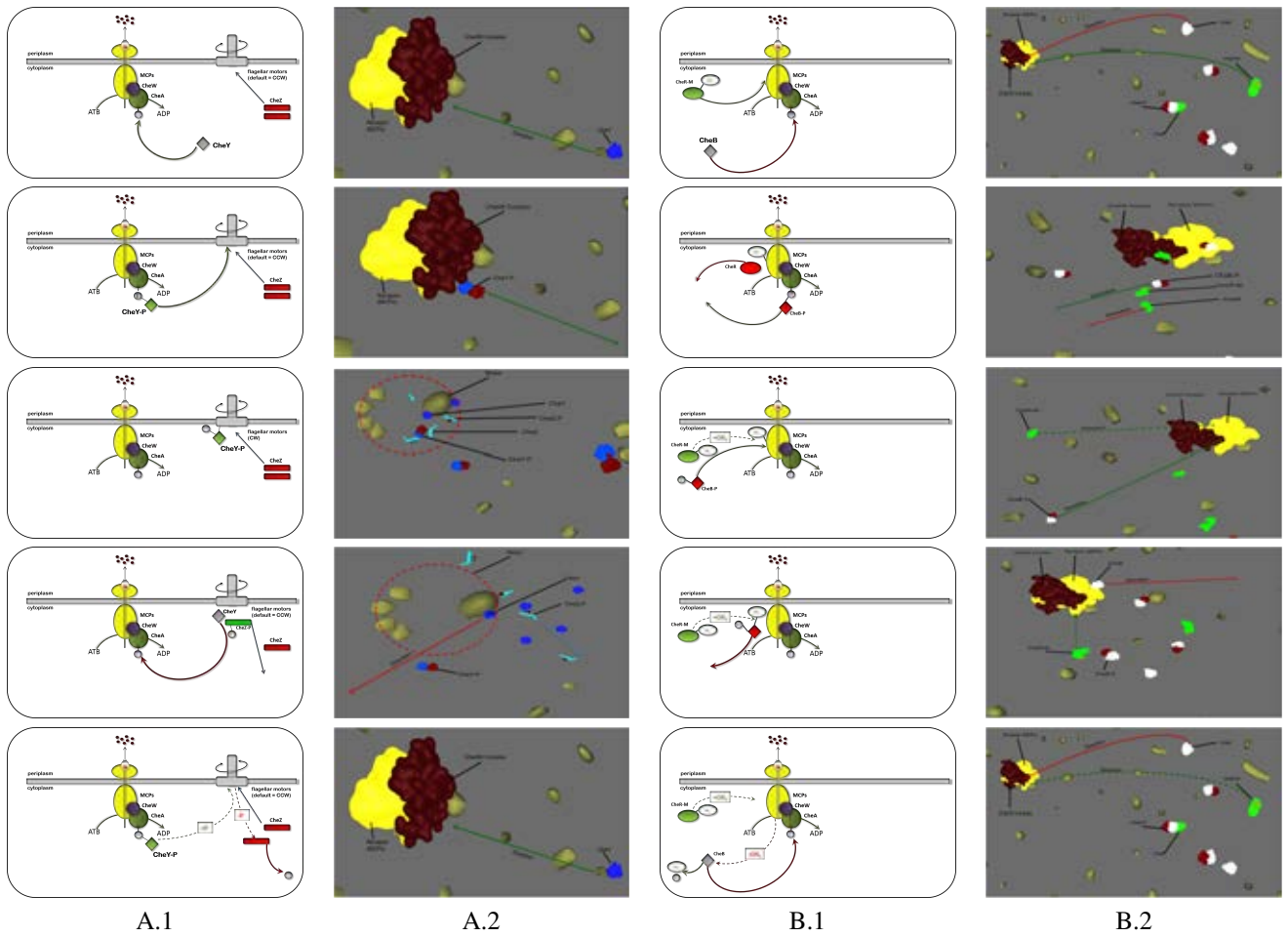
Figure 4: Chemotactic pathways: schematics and translations into our agent-based *E. coli* model. A.1) Excitation; A.2) agent interactions for excitation; B.1) Adaptation; B.2) agent dynamics during adaptation response.

tration, making the motor turn counter-clockwise, and suppressing the bacterium's tumble motion (Miller et al., 2010).

Active CheAW phosphorylates both CheB and CheY. CheB plays a role in adaptation (see below). CheZ's role is to de-phosphorylate CheY-P. The rotation of the flagella is dependent on the concentrations of CheY and CheY-P. A counter-clockwise rotation results from a higher CheY concentration. Likewise, motor rotation occurs in a clockwise fashion when CheY-P is present at a higher concentration than (unphosphorylated) CheY. Increasing attractant or decreasing the repellent concentration will bias the bacterium into swimming in smooth arcs by suppressing CheY phosphorylation, and thereby increasing the concentration of CheY. Similarly, an increase in repellent (or decrease in attractant) results in an increase of CheY-P, which leads to more frequent tumbling.

Adaptation Response: Methylation Adaptation is the process of the cell returning to its normal state of behavior (Fig. 4B). Without methylation, the cell would continuously travel in a straight line regardless of the conditions. Methylation ensures that there is a recovery condition in the cell so that it may continue to query its vicinity for a more favourable location to travel towards.

Methylation involves the CheA-CheW complex again (CheAW), as well as CheR and CheB. CheB is phosphorylated by CheAW into CheB-P, which subsequently removes a methyl group from the receptor (Berg, 2004; Adler et al., 1973; Patnaik, 2007). This process is called de-methylation, after which CheB-P returns to its unphosphorylated state CheB. Regardless of conditions inside or outside of the cell, CheR keeps methylating the receptor, which reactivates it. Consequently, CheB-P and CheR alternate in deactivating and reactivating the receptors.

Models of Bacterial Chemotaxis

There are two key methods for modelling biological systems: mathematical or agent-based approaches. The method of choice depends on the system or process being modelled. In recent years, a new method known as hybrid modelling is

being utilized, which is precisely what we are using to create our chemotaxis multi-scale model. This method combines both mathematical and agent-based models and automatically switches between these two techniques as required.

Following a hybrid approach for our *E. coli* model, we use mathematical equations to track diffusion of attractants and repellents as well as the concentration of cells in an agar plate environment. In order to simulate single-cell behaviours and bio-molecular interactions inside a cell, we use an agent-based model, where we track movements and collisions of elements in 3-dimensional cytoplasmic space. In the following sections, we give a brief overview of related modelling work for *E. coli* chemotaxis.

Mathematical Models of Chemotaxis

Many models of chemotaxis in *E. coli* have been developed over the last few decades, which differ in their comprehensiveness and coverage of pathways (Fernando, 2005). Some models capture general principles, while others focus on particular pathway parts. In fine-tuned models, biochemical parameters are accurately replicated, whereas more robust models replicate a wide range of parameters (Alon, 2007).

In the 1970s, Keller and Segel (1971) proposed a mathematical model that was originally developed to analyze the movement of slime molds. Over many years, their model has served as the foundation for modelling chemotaxis at the population level. Knox et al. (1986) proposed a theoretical model for the adaptation process. This model has formed the basis for later theoretical work on adaptation response and is often replicated in fine-tuned models (Alon, 2007).

Bray et al. developed a computational chemotaxis model (Bray et al., 1993), the Bacterial Chemotaxis Program (BCT), which initially modelled only the excitation response. BCT later incorporated a more biologically accurate representation of the receptor complexes (Bray and Bourret, 1995), binding affinities were optimized by an evolutionary algorithm (Fernando, 2005), and incorporation of receptor clustering and sensitivity (Bray et al., 1998). Bray et al. also created *E. solo* (Bray and Lipkow, 2007), using ordinary differential equations to replicate the signaling reactions in the pathway. E. solo provides a graphical display of bacterial movement in a 2D environment.

Barkai and Leibler (1997), presented a model for the adaptation response in chemotaxis, which takes a wide range of possible values for biochemical parameters into account. Their model includes several methylation sites, and reproduces many observations on the dynamical chemotactic behaviour of cells (Barkai and Leibler, 1997). Using a three-component model they showed that the adaptation process is robust rather than fine-tuned (Alon, 2007).

In 1999, the BCT team and Morton-Firth et al. developed *StochSim*, the first stochastic simulation model of bacterial chemotaxis (Morton-Firth et al., 1999). StochSim incorporates both excitation and adaptation responses. *Smoldyn*, an extension of StochSim, simulates cell-scale biochemical reactions to capture natural stochasticity data. The program was developed to provide a more realistic way to simulate the diffusion of signaling molecules through the cytoplasm (Andrews and Bray, 2004). StochSim was further expanded by Emonet et al. (2005) to develop *AgentCell*, which utilizes agent-based modelling to represent chemotactic responses at the population and single cell level, simulated independently. AgentCell accurately reproduces validated results under both stimulated and unstimulated conditions.

A more recent framework to specify and simulate microcolony growth and molecular signaling for synthetic biology applications was developed by Jang et al. (2012).

Agent Based Models (ABM)

In agent-based approaches one simulates the interactions of elements ("agents") with other elements and their environment. These interactions often give rise to complex patterns, referred to as emergence (Macal and North, 2005; Bonabeau, 2002). Emergent properties are often not identifiable by looking at the individual agents, but evolve from interactions between agents, as described by Ginovart et al. (2002) for discrete simulations of bacterial cultures .

Most modelers consider any independent component (software, object, model, etc.) with some sort of defined (programmed) behaviour rules to be an agent. The behaviour can range from primitive reactive protocols to adaptive intelligence programs (Mellouli et al., 2003). Berry (1997) proposed that an agent should contain both base-level rules for its behaviour and higher-level protocols to "change the rules" (adaptive intelligence). The base-level rules provide a reaction to the environment, whereas the higher-level protocols provide adaptation to the environment. This is the agent definition we have followed in our *E. coli* model.

Hybrid Chemotaxis Models

The use of hybrid models, which combine mathematical modelling with ABM techniques, is becoming widespread especially with the growth in computational power. This allows for more complex systems to be modelled and simulated such as biological systems and their cellular processes. Hybrid modelling has been applied for tumor growth (Patel et al., 2001) and forest dynamics (Landsberg and Waring, 1997). Hybrid models have been applied for chemotaxis in slime molds (Dallon and Othmer, 1997) and bacteria (Fernando, 2005).

A Hybrid, Multi-scale Model of Chemotaxis

As an extension of *Prokaryo*, a hybrid model of prokaryotic gene regulation (Esmaeili et al., 2015), we have developed a generalized model of *E. coli* chemotaxis. The model captures the key attributes and characteristics of the chemotaxis pathways which control the locomotion of bacteria in a simulated agar. The mathematical model handles all of the
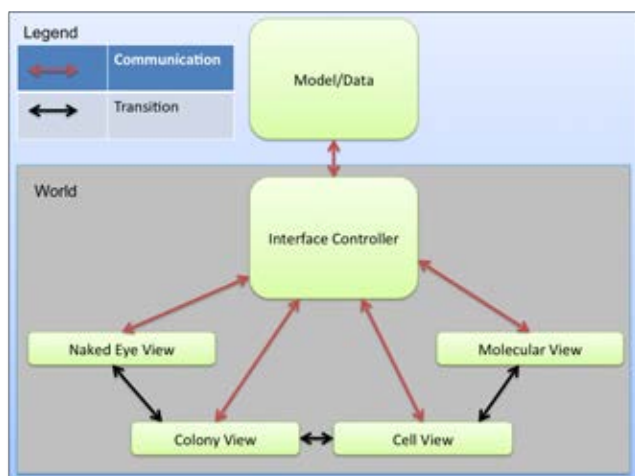
Figure 5: Software architecture and communication hierarchy between the components in the *E. coli* model.

calculations related to intra-celluar and extra-cellular signals as well as concentrations of molecules. The ABM handles interactions among cells (as individual agents on the agar) and among molecules in the cell's cytoplasm. By zooming in and out of the 3D scenario, the scales for the model and its visualization automatically transition between four layers (Fig. 1), which includes switching from ABM calculations to the mathematical model (and back). The mathematical model is always executed in the background; this ensures that information is shared between the layers.

Model Architecture

We have used the latest version of our *LINDSAY Composer 2.0* agent simulation software (Jacob et al., 2012) to implement the multi-scale *E. coli* model. LINDSAY Composer provides 3D simulation, including physics and graphics engines, camera navigation, interactive parameter manipulation, scene hierarchies, and an object-oriented programming environment. The architecture of our model consists of the following components (Fig. 5):

Model Data: Simulation data is stored and shared among the layers through this module, which is only accessible through the Interface Controller.

Naked Eye Layer: In this top-level view, concentrations of chemicals (attractants or repellents) as well as colony distribution is visualized by color gradients, which are drawn onto the grid surface that represents the agar environment. Algorithm 1 is used to update the grid.

Colony Layer: In this layer, *E. coli* cells and stimuli are represented through particle systems, which are used to generate the illusion of thousands of cells and molecules in the environment. Movement of the particles is governed by Algorithm 1.

Cell Layer: In order to reach this level, a single cell has been selected on the Colony Layer. An *E. coli* cell is represented as an agent with its own distinct properties. The cell moves based on the other agents and the stimuli in the simulation at each time step.

Molecular Layer: The lowest layer of our simulation captures the molecular details and interactions of the chemotaxis pathways. Molecules and proteins are modelled as individual agents that interact with other agents and the environment.

Simulation Engine: The Interface Controller and model layers are embedded in the simulation environment. Transitions only occur between connected layers.

Interface Controller: The interface controller relays interactions from the user (gestures, mouse clicks and movements, parameter manipulations) to the model layers and the data module.

Model data: The model data is situated in a separate module and shared among the layers by communicating through the Interface Controller.

Algorithm 1 Update of Petri Dish and Colony Layer

<u>CREATE:</u>
Initialize 2D grid cells
$attractant := 0$; $repellent := 0$; $ecoli := 0$

<u>ITERATE:</u>
for all $gcell \in Grid$ **do**
    for all $e_{gcell} \in \{attractant, repellent, ecoli\}$ **do**
        if $|e_{gcell}| \neq 0$ **then**   ▷ Cell contains element(s)
            Apply Gaussian algorithm   ▷ Diffusion
            Update $gcell$
        end if
    end for
    if $layer == PetriDish$ **then**   ▷ Petri dish layer
        $isColourCell := true$
        Apply colour visuals to $gcell$
    else   ▷ Colony layer
        $isParticlesCell := true$
        Generate particles for $gcell$
    end if
end for

Molecular Agents

All protein models in our simulation have been extracted from online protein databases. We have listed the 3D shapes and the PDB IDs of our bio-molecular agents in Figure 7. Recall the role of each agent from Table 1. In this paper, we only have space to illustrate two agents and how we have implemented their behaviour rules. More information is available on our project website (LindsayVirtualHuman.org).

CheY protein is phosphorylated by the CheAW complex (Algorithm 2). CheY-P will dock onto the motor unit. Upon
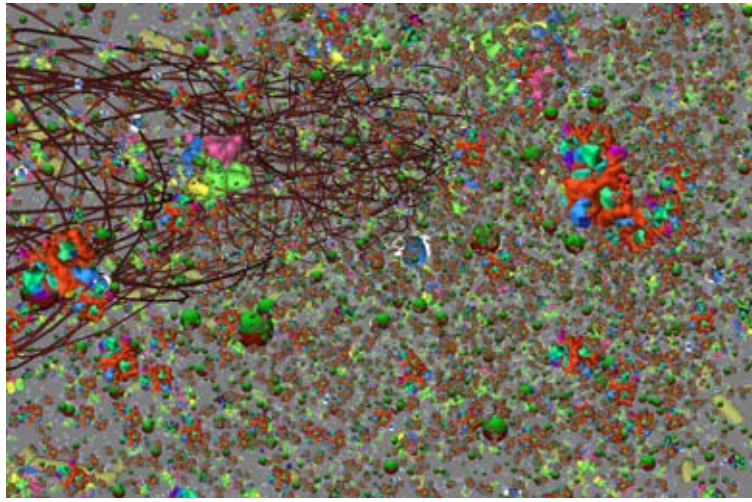
Figure 6: A glimpse of the full-scale simulation inside the *E. coli* cell's cytoplasm.

Receptor   CheAW

CheY   CheY+P   CheB   CheB+P
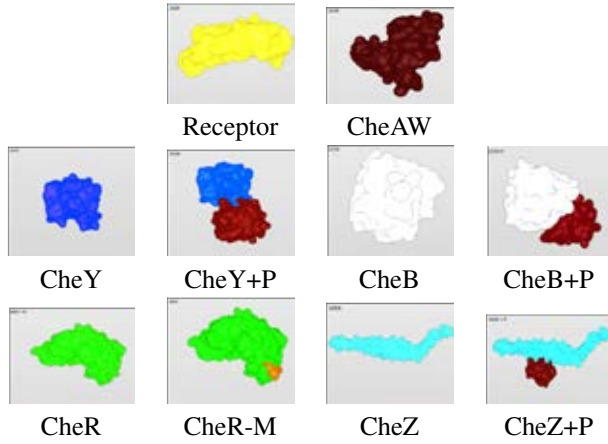
CheR   CheR-M   CheZ   CheZ+P

Figure 7: 3D structures of the protein agents in our chemotaxis model as used in the cytoplasm layer (Fig. 6). The meshes were imported using the associated PDB Ids from the Protein Databank (www.wwpdb.org).

collision with CheZ, CheY-P gets de-phosphorilized and follows an attractive force towards the CheAW complex, which will start the interaction loop again.

CheZ is set to be attracted to the motor complex (Algorithm 3). CheZ performs a random walk in the vicinity of the motor complex, checking for collisions with phosphorylated CheY. Upon collision with a CheY-P agent, CheZ removes the phosphate group from CheY, and subsequently enters a short period of inactivity.

An impression of what the simulation looks like—with all molecular agents, including water and lactose, interacting inside the cytoplasmic space—is depicted in Figure 6.

Algorithm 2 CheY

1: CREATE:
2: set state to $RandomWalk$

3: ITERATE:
4: **if** CheY is $Active$ **then**
5:    **if** CheY is not phosphorylated **then**
6:      **if** collided with CheAW complex **then**
7:        set state to phosphorylated
8:        set agent conformation to CheY-P mesh
9:        set attraction to Motor
10:      **else**
11:        set state to $RandomWalk$
12:      **end if**
13:    **else**   ▷ CheY is phosphorylated
14:      **if** collided with Motor **then**
15:        set state to $BoundToMotor$
16:        clear movement velocities
17:        **if** collided with CheZ **then**
18:          set state to not phosphorylated
19:          set agent conformation to CheY mesh
20:          set attraction towards CheAW Complex
21:        **end if**
22:      **end if**
23:    **end if**
24: **else**
25:    set state to $RandomWalk$
26: **end if**

Simulation Results

We have been using results from our simulations throughout the illustrations in this paper. Starting with our Eye and Colony view, we can see that in an environment such as

Algorithm 3 CheZ

1: CREATE:
2: set state to $RandomWalk$, $Active$
3: set attraction towards Motor
4: set timer to 100   ▷ Start timer

5: ITERATE:
6: **if** CheZ is $Active$ **then**
7:     **if** CheZ collided with CheY-P **then**
8:        remove P from CheY-P
9:        set CheZ to $\neg Active$
10:    **end if**
11: **else**
12:    timer := timer -1
13:    **if** timer == 0 **then**
14:       set CheZ to $Active$
15:       timer := 100   ▷ Restart timer
16:    **end if**
17: **end if**
---

an agar plate, with no external stimulus, chemotactic rings are formed (Fig. 2B,C), similar to a wet-lab experiment of *E. coli* growing in a tryptone soft agar plate.

Placing an attractant stimulus in the environment, our simulated *E. coli* cells grow and move towards the origin of attraction (Fig. 8A). Similarly, with a repellent stimulus the colony moves away from the repelling source (Fig. 8B).

In single cell view, we see an *E. coli* bacterium following a gradient or performing a random walk in the absence of a stimulus (Fig. 9).

On the molecular interaction level, we have replicated the different interaction phases of the chemotaxis pathway. Figure 4 illustrates this with a side-by-side comparison of the pathway diagrams and their replication in our agent-based model, where the agent behaviours are driven by short code scripts, such as the examples of Algorithms 2 and 3.

Conclusion

We have introduced a multi-scale, hybrid model that replicates and illustrates chemotaxis of *E. coli* bacteria. We have implemented abstractions of chemotaxis on four levels of detail: from the naked eye and colony level down to single cells and their cytoplasm. Our model system is interactive, provides 3D visuals, and can serve as a tool to learn about and explore behaviours in biological systems arising from the interactions of many constituents across a range of scales of resolution. More information about this system and related simulations can be found on the LINDSAY Virtual Human web site.

References

Adler, J. (1966a). Chemotaxis in bacteria. *Science*, 153(3737):708–715.

A.1: gradient at t and $t + \Delta t$ A.2: colony at t and $t + \Delta t$

B.1: gradient at t and $t + \Delta t$ B.2: colony at t and $t + \Delta t$
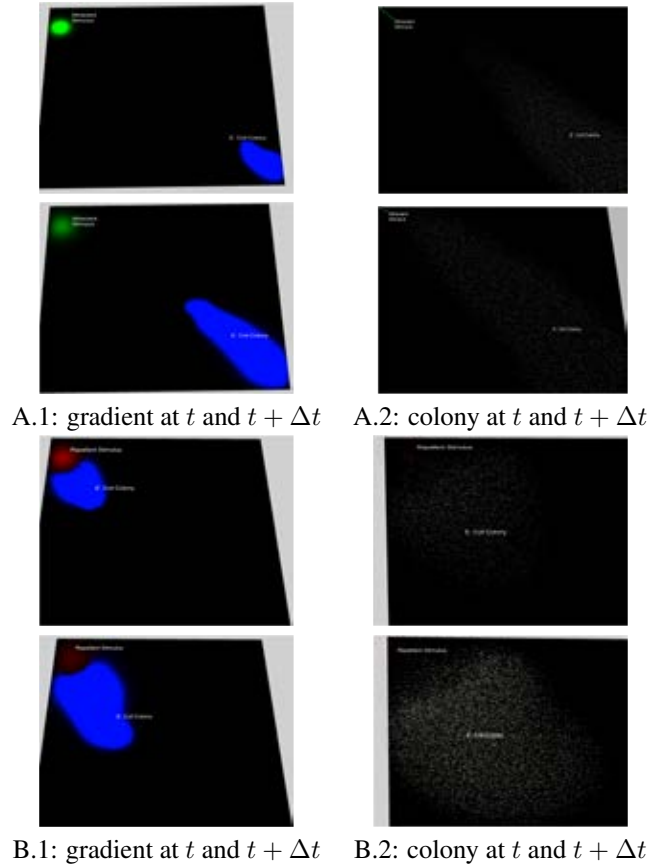
Figure 8: Modeled chemotaxis: Movement toward an attractant (A) and away from a repellent stimulus (B). The left column shows colony movement (blue), repellent (red) and attractant (green); bacterial colony closeups on the right.
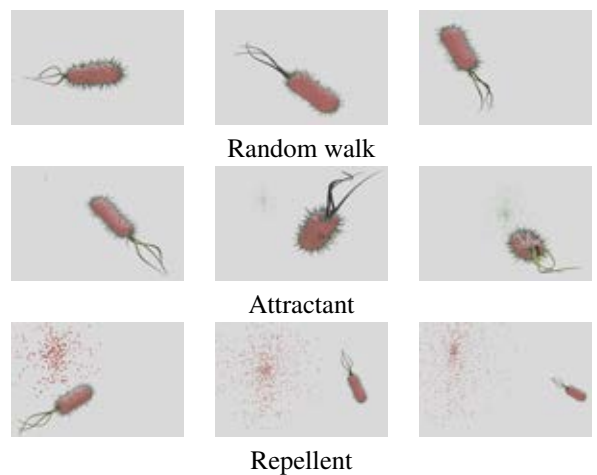

Random walk

Attractant

Repellent

Figure 9: *E. coli* as a single agent in agar environment.

Adler, J. (1966b). Effect of amino acids and oxygen on chemotaxis in Escherichia coli. *Journal of bacteriology*, 92(1):121–129.

Adler, J. (1969). Chemoreceptors in bacteria. *Science*, 166(3913):1588–1597.

Adler, J. (1973). A method for measuring chemotaxis and use of the method to determine optimum conditions for chemotaxis by Escherichia coli. *Journal of General Microbiology*, 74:77–91.

Adler, J., Hazelbauer, G. L., and Dahl, M. M. (1973). Chemotaxis toward sugars in Escherichia coli. *Journal of Bacteriology*, 115(3):824.

Alon, U. (2007). *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman and Hall/CRC.

Andrews, S. S. and Bray, D. (2004). Stochastic simulation of chemical reactions with spatial resolution and single molecule detail. *Physical Biology*, 1(3):137–151.

Barkai, N. and Leibler, S. (1997). Robustness in simple biochemical networks. *Nature*, 387(6636):913–917.

Berg, H. C. (2004). *E. coli in Motion*. Springer.

Berry, M. (1997). Would-be worlds: How simulation is changing the frontiers of science - Casti,JL. *Nature*, 385(6611):33.

Bonabeau, E. (2002). Agent-based modeling: methods and techniques for simulating human systems. *PNAS*, 99 Suppl 3(90003):7280–7287.

Bray, D. and Bourret, R. B. (1995). Computer analysis of the binding reactions leading to a transmembrane receptor-linked multiprotein complex involved in bacterial chemotaxis. *Molecular biology of the cell*, 6(10):1367–1380.

Bray, D., Bourret, R. B., and Simon, M. I. (1993). Computer simulation of the phosphorylation cascade controlling bacterial chemotaxis. *Molecular Biology of the Cell*, 4(5):469–482.

Bray, D., Levin, M. D., and Morton-Firth, C. J. (1998). Receptor clustering as a cellular mechanism to control sensitivity. *Nature*, 393(6680):85–88.

Bray, D Levin, M. D. and Lipkow, K. (2007). The chemotactic behavior of computer-based surrogate bacteria. *Current Biology*, 17:12–19.

Dallon, J. C. and Othmer, H. G. (1997). A discrete cell model with adaptive signalling for aggregation of Dictyostelium discoideum. *Philosophical Transactions of the Royal Society B-Biological Sciences*, 352(1351):391–417.

Emonet, T., Macal, C. M., North, M. J., and Wickersham, C. E. (2005). AgentCell: a digital single-cell assay for bacterial chemotaxis. *Bioinformatics*, 21(11):2714–2721.

Esmaeili, A., Davison, T., Wu, A., Alcantara, J., and Jacob, C. (2015). Prokaryo: an illustrative and interactive computational model of the lactose operon in the bacterium escherichia coli. *BMC Bioinformatics*, 16(1):1–23.

Fernando, C. (2005). Computer Models of bacterial Chemotaxis.

Ginovart, M., Lopez, D., and Valls, J. (2002). Indisim, an individual-based discrete simulation model to study bacterial cultures. *Journal of theoretical biology*, 214(2):305–319.

Hauri, D. C. and Ross, J. (1995). A model of excitation and adaptation in bacterial chemotaxis. *Biophysical journal*, 68(2):708–722.

Jacob, C., von Mammen, S., Davison, T., Sarraf-Shirazi, A., Sarpe, V., Esmaeili, A., Phillips, D., Yazdanbod, I., Novakowski, S., Steil, S., Gingras, C., Jamniczky, H., Hallgrimsson, B., and Wright, B. (2012). *LINDSAY Virtual Human: Multiscale, Agent-based, and Interactive*, chapter 14, pages 327–349. Springer.

Jang, S. S., Oishi, K. T., Egbert, R. G., and Klavins, E. (2012). Specification and simulation of synthetic multicelled behaviors. *ACS synthetic biology*, 1(8):365–374.

Keller, E. F. and Segel, L. A. (1971). Traveling Bands of Chemotactic Bacteria - Theoretical Analysis. *Journal of theoretical biology*, 30(2):235–248.

Knox, B. E., Devreotes, P. N., Goldbeter, A., and Segel, L. A. (1986). A molecular mechanism for sensory adaptation based on ligand-induced receptor modification. *PNAS*, 83(8):2345–2349.

Landsberg, J. J. and Waring, R. H. (1997). A generalised model of forest productivity using simplified concepts of radiation-use efficiency, carbon balance and partitioning. *Forest Ecology and Management*, 95(3):209–228.

Macal, C. M. and North, M. J. (2005). Tutorial on agent-based modeling and simulation. In *Simulation Conference, 2005 Proceedings of the Winter*, pages 2–15. IEEE.

Mellouli, S., Moulin, B., and Mineau, G. (2003). Laying down the foundations of an agent modelling methodology for fault-tolerant multi-agent systems. *Engineering Societies in the Agents World Iv*, 3071:275–293.

Miller, J., Parker, M., Bourret, R. B., and Giddings, M. C. (2010). An agent-based model of signal transduction in bacterial chemotaxis. *PloS one*, 5(5):e9454.

Morton-Firth, C. J., Shimizu, T. S., and Bray, D. (1999). A free-energy-based stochastic simulation of the Tar receptor complex. *Journal of molecular biology*, 286(4):1059–1074.

Patel, A. A., Gawlinski, E. T., Lemieux, S. K., and Gatenby, R. A. (2001). A cellular automaton model of early tumor growth and invasion: The effects of native tissue vascularity and increased anaerobic tumor metabolism. *Journal of theoretical biology*, 213(3):315–331.

Patnaik, P. R. (2007). Robustness analysis of the E. coli chemosensory system to perturbations in chemoattractant concentrations. *Bioinformatics*, 23(7):875–881.

Segall, J. E., Block, S. M., and Berg, H. C. (1986). Temporal comparisons in bacterial chemotaxis. *Proceedings of the National Academy of Sciences*, 83(23):8987–8991.

Sourjik, V. (2004). Receptor clustering and signal processing in E. coli chemotaxis. *Trends in microbiology*, 12(12):569–576.

Wadhams, G. H. and Armitage, J. P. (2004). Making sense of it all: bacterial chemotaxis. *Nature Reviews Molecular Cell Biology*, 5(12):1024–1037.

Zimmer, C. (2009). *Microcosm: E. coli and the new science of life*. Vintage Books.

553

Flies as Ship Captains?
Digital Evolution Unravels Selective Pressures to Avoid Collision in Drosophila

Ali Tehrani-Saleh[1,3], Randal S. Olson[1,3], and Christoph Adami[2,3]

[1]Dept. of Computer Science and Engineering
[2]Depts. of Physics and Astronomy & Microbiology and Molecular Genetics
[3]BEACON Center for the Study of Evolution in Action
Michigan State University, East Lansing, MI 48824
adami@msu.edu

Abstract

Flies that walk in a covered planar arena on straight paths avoid colliding with each other, but which of the two flies stops is not random. High-throughput video observations, coupled with dedicated experiments with controlled robot flies have revealed that flies utilize the type of optic flow on their retina as a determinant of who should stop, a strategy also used by ship captains to determine which of two ships on a collision course should throw engines in reverse. We use digital evolution to test whether this strategy evolves when collision avoidance is the sole selective pressure. We find that the strategy does indeed evolve in a narrow range of cost/benefit ratios, for experiments in which the "regressive motion" cue is error free. We speculate that these stringent conditions may not be sufficient to evolve the strategy in real flies, pointing perhaps to auxiliary costs and benefits not modeled in our study.

Introduction

How animals make decisions has always been an interesting, yet controversial, question to scientists (McFarland, 1977) and philosophers alike. Animals obtain various types of sensory information from the environment and then process these information streams so as to take actions that benefit them in survival and reproduction. The visual system plays an important role in providing animals information about their environment, for example when foraging for food, detecting predators or prey, and when searching for potential mates. One of the primary components of visual information is motion detection. Motion is a fundamental perceptual dimension of visual systems (Borst and Egelhaaf, 1989) and is a key component in decision making in most animals. Here, we study a very particular type of motion detection and concomitant behavior (collision avoidance) in *Drosophila melanogaster* (the common fruit fly), and attempt to unravel the selective (i.e., evolutionary) pressures that might have given rise to this behavior.

D. melanogaster shows a striking difference in behavior when exposed to two different types of optical flow. Branson et al. (2009) recorded the interaction of groups of fruit flies in a planar covered arena (so that they could only walk, not fly) and used computer vision algorithms to analyze the walking trajectories in order to study fly behavior. Their analysis revealed that female fruit flies stop walking when they perceive another fly's motion from back-to-front in their visual field (an optical flow referred to as "regressive motion") whereas they keep walking when perceiving conspecifics moving from front-to-back in their visual field (referred to as "progressive motion," see Figure 1). Zabala et al. (2012) further investigated this behavior and tested the "regressive motion saliency" hypothesis, suggesting that flies stop walking when perceiving regressive motion. They used a programmable fly-sized robot interacting with a real fly to exclude other sensory cues such as image expansion ("looming," see Schiff et al. 1962) and pheromones. Their results provide rigorous support for the regressive motion saliency hypothesis.

Subsequently, Chalupka et al. (2015) coined the term "generalized regressive motion" for optic flows in which images move clockwise on the left eye and conversely, counterclockwise on the right eye (see Figure 1). They presented a geometric analysis for two flies moving on straight, intersecting trajectories with constant velocities and showed that the fly that reaches the intersection first always perceives progressive motion on its retina, whereas the one that reaches the intersection later perceives regressive motion at all times before the other fly reaches the intersection. They went on to suggest that this behavior is a strategy to avoid collisions during locomotion similar to the rules that ship captains use when moving on intersecting paths (see, e.g., Maloney 1989).

As intriguing as this hypothesis may seem, it is not clear *a priori* which selective pressures or environmental circumstances could give rise to this behavior. For example, it is unclear whether collision avoidance provides a significant enough fitness benefit. As a consequence, it is possible that the behavior has its origin in a completely different cognitive constraint that is fundamentally unrelated to collision avoidance, or to the rules that ship captains use to navigate the seas. While such questions are difficult to answer using traditional behavioral biology methods, Artificial Life offers unique opportunities to test these hypotheses directly.
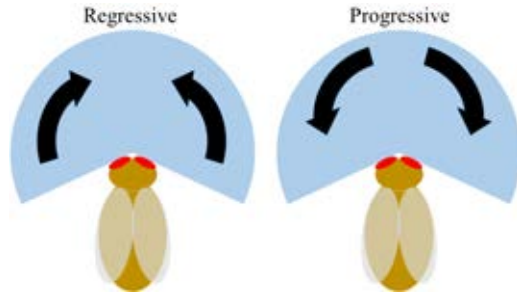
Figure 1: An illustration of regressive (back-to-front, left) and progressive (front-to-back, right) optic flows in a fly's retina.

In this study, we tested whether collision avoidance can be a sufficient selective pressure for the described behavior to evolve. We also investigated the environmental conditions under which this behavior could have evolved, in terms of the varying costs and benefits involved. By using an agent-based computational model (described in more detail below), we studied how the interplay (and trade-offs) between the necessity to move and the avoidance of collisions can result in the evolution of regressive motion saliency in digital flies.

Digital evolution is currently the only technique that can study hypotheses concerning the selective pressures necessary (or even sufficient) for the emergence of animal behaviors, as experimental evolution with animal lines of thousands of generations is impractical. In digital evolution, we can study the interplay between multiple factors such as selective pressures, environmental conditions, population size and structure, etc. For example, Olson et al. (2013) used digital evolution to show that predator confusion is a sufficient condition to evolve swarming behavior, but they also found that collective vigilance can give rise to gregarious foraging behavior in group-living organisms (Olson et al., 2015). In principle, any one hypothesis favoring the emergence of behavior can be tested in isolation, or in conjunction (Olson et al., 2015).

Methods

Markov Networks

We use an agent-based model to simulate the interaction of walking flies with moving objects (here, potentially conspecifics) in a two-dimensional virtual world. Agents have sensors to perceive their surrounding world (details below) and have actuators that enable them to move in the environment. Agent brains in our experiment have altogether twelve sensors, three internal processing nodes, and one output node (the actuator). The brain controlling the agent is a "Markov network brain" (MNB), which is a probabilistic controller that makes decisions based on sensory inputs and internal nodes (Edlund et al., 2011). Each node in the
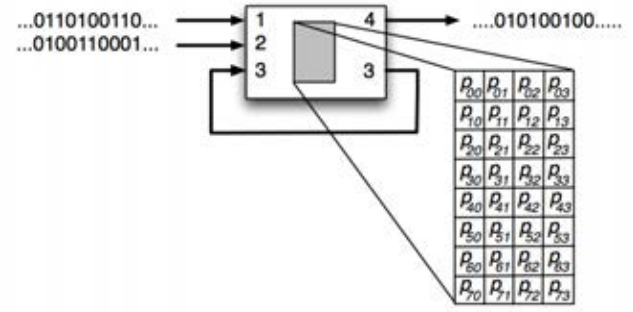

Figure 2: Probabilistic logic gates in Markov network brains with three inputs and two outputs. One of the outputs writes into one of the inputs of this gate, so its output is "hidden." Because after firing all Markov neurons automatically return to the quiescent state, values can only be kept in memory by actively maintaining them. Probability table shows the probability of each output given input values.

network (i.e., sensors, internal nodes, and actuators) can be thought of as a digital (binary) neuron that either fires (value=1), or is quiescent (value=0). Nodes of the network are connected via Hidden Markov Gates (HMGs) that function as probabilistic logic gates. Each HMG is specified by its inputs, outputs, and a state transition table that specifies the probability of each output state based on input states (Figure 2). For example, in the transition table of Figure 2 (a three-input, two-output gate), the probability p_{73} controls the likelihood that the output state is 3 (the decimal equivalent of the binary pattern 11, that is, both output neurons fire) given that the input happened to be state 7 (the decimal translation of 111, i.e., all inputs are active). MNBs can consist of any number of HMGs with any possible connection arrangement, given certain constrains (see for example Edlund et al. 2011).

The number of gates, their connections, and how they work is subject to evolution and changes across individuals and through generations. For this purpose, the agent's brains are encoded in a genome, which is an ordered sequence of integers, each in the range [0,255], i.e., one byte. Each integer (or byte) is a locus in the genome and specific sequences of loci construct genes, where each gene codes for one HMG. The "start codon" for a gene (i.e., the sequence that determines the beginning of the gene) in our encoding is the pair (42,213) (these numbers are arbitrary). Each gene encodes exactly one HMG, for example as shown in Figure 3. The gene specifies the number of inputs/outputs in each HMG, which nodes it reads from and writes to (the connectivity) and the probability table that determines the gates' function. As shown in Figure 3, the first two bytes are the start codon, followed by one byte that specifies the number of inputs and one byte for the number of outputs. The bytes are modulated so as to encode the number of inputs and out-
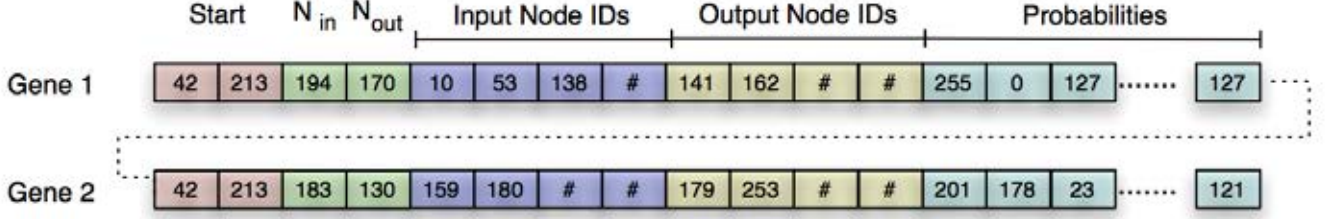
Figure 3: An illustration of a portion of genome containing two genes that encode two HMGs. The first two loci represent start codon (red blocks), followed by two loci that determine the number of inputs and outputs respectively (green blocks). The next four loci specify which nodes are inputs of this gate (blue blocks) and the following four specify output nodes (yellow blocks). The remaining loci encode the probabilities of HMG's logic table (cyan blocks).

puts unambiguously. For example, the bytes encoding the number of inputs is an integer in [0,255] whereas a HMG can take a maximum of four inputs, thus we use a mapping function that generates a number $\in$ [1,4] from the value of this byte. The next four bytes specify the inputs of the HMG, followed by another four bytes specifying where it writes to. The remaining bytes of the gene are mapped to construct the probabilistic logic gate table. MNBs have been used extensively in the last five years to study the evolution of navigation (Edlund et al., 2011; Joshi et al., 2013), the evolution of active categorical perception (Marstaller et al., 2013; Albantakis et al., 2014), the evolution of swarming behavior as noted earlier, as well as how visual cortices (Chapman et al., 2013) and hierarchical groups (Hintze and Miromeni, 2014) form. In this work, we force the gates to be deterministic rather than probabilistic (all values in the logic table are 0 or 1), which turns our HMGs into classical logic gates.

Experimental Configurations

We construct an initial population of 100 agents (digital flies), each with a genome initialized with 5,000 random integers containing four start codons (to jump-start evolution). Agents (and by proxy the genomes that determine them) are scored based on how they perform in their living environment. The population of genomes is updated via a standard Genetic Algorithm (GA) for 50,000 generations, where the next generation of genomes is constructed via roulette wheel selection combined with mutations (detailed GA specifications are listed in Table 1). To control for the effects of reproduction and similar effects, there is no crossover or immigration in our GA implementation.

Each digital fly is put in a virtual world for 25,000 time steps, during which time its fitness score is evaluated. During each time step in the simulation, the agent perceives its surrounding environment, processes the information with its MNB, and makes movement decisions according to the MNB outputs. The sensory system of a digital fly is designed such that it can see surrounding objects within a limited distance of 250 units, in a 280° pixelated retina shown

in Figure 4. The state of each sensor node is 0 (inactive) when it does not sense anything within the radius, and turns to 1 (active) if an object is projected at that position in the retina. Agents in this experiment have one actuator node that enables them to move ahead or stop, for active (firing) and non-active (quiescent) states respectively.

GA Parameters		Environment Parameters	
Population size	100	Vision range	250
Generations	50,000	Field of vision	280°
Point mutation rate	0.5%	Collision range	60
Gene deletion rate	2%	Agent velocity	15
Gene duplication rate	5%	Event time steps	250
Initial genome length	5,000	No. of events	100
Initial start codons	4	Moving reward	0.004
Crossover	None	Collis. penalty	1,2,3,5,10
Immigration	None	Replicates	20

Table 1: Configurations for GA and Environmental setup

In our experiment, the digital flies exist in an environment where they should move to gain fitness, representing the fact that organisms should forage for resources, mates, and avoiding predators. Thus, the fitness function is set so that agents are rewarded for moving ahead at each update of the world, and are penalized for colliding with objects. The amount of fitness they gain for moving (the benefit) is characteristic of the environment, and we change it in different treatments. The penalty for collisions represents the importance of collision avoidance for their survival and reproduction, and we vary this cost also. Each digital fly sees 100 moving objects (one at a time) during its lifetime, and we say that it experiences 100 "events." The penalty-reward ratio (PR) determines the amount of penalty of collision divided by the reward for moving during the entirety of an event. So for example, PR=1 means the agent loses all the rewards it gained by walking during the whole event if it collides with the object in that event:

$$\text{fitness} = \sum_{\text{events}} \left(\text{reward} - \text{PR} \times \text{collision} \right), \qquad (1)$$
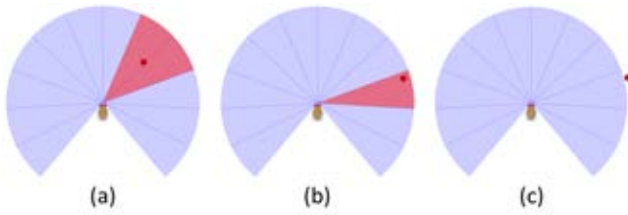
Figure 4: The digital fly and its visual field in the model. Flies have a 12 pixel retina that is able to sense surrounding objects in 280° within a limited distance (250 units). The red circle is an external object that can be detected by the agent within its vision field. Activated sensors are shown in red, while inactive sensors are blue. In (a) the object activates two sensors, in (b) the object is detected in one sensor, and in (c) the object is outside the range.

where reward $\in [0, 1]$ reflects how many time steps the agent moved during the event. Our experiments are constructed such that *all* objects that produce regressive motion in the digital retina *will* collide with the fly if it keeps moving. The reason for biasing our experiments in this manner is explained in the following section.

Collision Probability in Events with Regressive Optic Flow

As mentioned earlier, Chalupka et al. (2015) showed that for two flies moving on straight, intersecting trajectories with constant velocities, the fly that reaches the intersection first always perceives progressive motion on its retina while the counterpart that reaches the intersection later perceives regressive motion at all times before the first fly reaches the intersection. However, this does *not* imply that all objects that produce a regressive motion on a fly's retina will necessarily collide with it. In this section we present a mathematical analysis to discover how often objects that produce regressive motion in the fly's retina will eventually collide with the fly if it continues walking.

Suppose a fly moves on a straight line with constant velocity $V_{\mathbf{fly}}$ and an object is also moving on a straight line with constant velocity $V_{\mathbf{obj}}$ (Figure 5-a). The fly is able to perceive objects within distance R_{vis}, its vision range (Figure 5-a). The object is assumed to be a point in the plane and the distance between this point and the center of the visual field of the fly is defined to be the distance between them. We define "the onset of the event" as the first time the object is detected by the fly. At the onset of the event, the object is at the distance R_{vis} of the fly at relative azimuthal angle $\alpha \in [0, \frac{\pi}{2}]$ (Figure 5-a). We assume that the object can be at any relative position $R_{\mathbf{vis}} = (R_{\text{vis}}, \alpha)$[1] with equal prob-

---

[1] Here and below, we represent vectors either in boldface or by the parameters that determine them within a planar polar coordinate system. Thus the vector $\mathbf{R}$ is represented by $(|\mathbf{R}|, \phi)$, where $R_x = R \cos \phi$ and $R_y = R \sin \phi$.
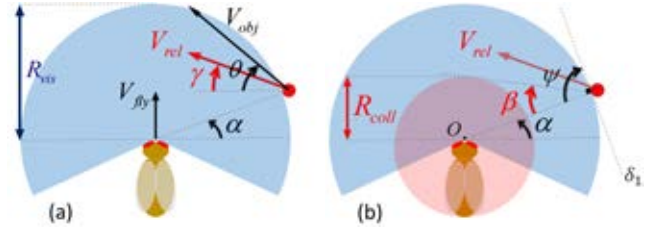

Figure 5: An illustration of a moving fly at the onset of the event.

abilities (the probability distribution of α is uniform around the fly). The velocity of the object can be represented as $V_{\mathbf{obj}} = (V_{\text{obj}}, \theta)$ where $\theta \in [\frac{-\pi}{2}, \frac{\pi}{2}]$ (note that $V_{\mathbf{obj}}$ is constant). We also assume that the velocity of the object can point in all directions with equal probabilities (the probability distribution of θ is uniform). The relative velocity of the object with respect to the fly is $V_{\mathbf{rel}} = V_{\mathbf{obj}} - V_{\mathbf{fly}}$ (Figure 5). Since both $V_{\mathbf{obj}}$ and $V_{\mathbf{fly}}$ are constant, $V_{\mathbf{rel}}$ is also a constant vector.

Proposition 1. A moving object produces regressive motion on a fly's retina if:

$$\theta > -\alpha + \arcsin(\frac{V_{\text{fly}}}{V_{\text{obj}}} \cos \alpha) . \tag{2}$$

Proof. In order for the object to produce regressive motion on the retina, the relative velocity should be pointed above the center point O. The relative velocity direction γ can be found awith $V_{\mathbf{rel}} = (V_{\text{rel}}, \gamma)$, as

$$\gamma = \arctan\left(\frac{V_{\text{rel}_y}}{V_{\text{rel}_x}}\right) = \arctan\left(\frac{V_{\text{obj}} \sin \theta - V_{\text{fly}}}{V_{\text{obj}} \cos \theta}\right) . \tag{3}$$

The angle γ should be greater than the central angle (Figure 5-b), that is, $\gamma > -\alpha$. Replacing γ and simplifying, we obtain:

$$\theta > -\alpha + \arcsin(\nu \cos \alpha), \quad \nu = \frac{V_{\text{fly}}}{V_{\text{obj}}} . \tag{4}$$

For smaller values of θ, the object produces progressive optic flow. We thus define $\theta_{\min} = -\alpha + \arcsin(\nu \cos(\alpha))$ as the minimum angle θ that produces regressive motion on the retina.

Definition 1. The object remains "observable" to the fly after the onset of the event if its relative velocity is directed toward the inside of the fly's vision field (to the left of the tangent line δ_1 in Figure 5-b).

Proposition 2. The object remains observable to the fly if:

$$\theta < \arccos(-\frac{V_{\text{fly}}}{V_{\text{obj}}} \sin \alpha) - \alpha . \tag{5}$$

Proof. According to the definition the sufficient condition for observability is that γ should be less than the tangent line δ_1 angle: $\gamma < -\alpha + \frac{\pi}{2}$. Replacing γ and simplifying we obtain

$$\theta < \arccos(-\nu \sin \alpha) - \alpha \ . \qquad (6)$$

For greater values of θ, the object will be out of vision range of the fly. Thus the maximum value that θ can take on is:

$$\theta_{\max} = \arccos(-\nu \sin \alpha) - \alpha \ . \qquad (7)$$

In order for the object to produce regressive motion on fly's retina and also remain observable to the fly, relative velocity should be within the arc ψ (Figure 5-b).

Definition 2. The object collides with the fly if its distance with the fly is less than "collision range" R_{coll} (Figure 5-b).

Proposition 3. An object that creates regressive optic flow on the fly's retina and remains observable will collide with it if:

$$\theta < \phi + \arcsin(\nu \cos \phi), \quad \phi = \arcsin(\frac{R_{\text{coll}}}{R_{\text{vis}}}) - \alpha \ . \qquad (8)$$

Proof. The relative velocity of such object is within arc ψ. This object will collide with the fly if its relative velocity is within the arc spanned by the angle β, i.e. lower than tangent line to collision circle (Figure 5-b). This condition holds true if:

$$\gamma < \beta - \alpha, \quad \beta = \arcsin(\frac{R_{\text{coll}}}{R_{\text{vis}}}) \ . \qquad (9)$$

Let $\rho = \frac{R_{\text{coll}}}{R_{\text{vis}}}$ and $\phi = \beta - \alpha$. Replacing γ and rearranging gives:

$$\theta < \phi + \arcsin(\nu \cos \phi) \ . \qquad (10)$$

For greater values of θ, the object produces regressive motion on the fly's retina but does not collide with it. So the threshold collision angle is given by:

$$\theta_{\text{col}} = \phi + \arcsin(\nu \cos \phi) \ . \qquad (11)$$

As mentioned, we assume that the probability distribution of the direction of the object velocity, θ is uniform.

Definition 3. For an object at initial position α, the probability Π_{coll} is the range of velocity directions θ such that the object collides with the fly divided by the range of directions with which it creates regressive optic flow on fly's retina (see Figure 5-b):

$$\Pi_{\text{coll}}(\alpha, \nu, \rho) = \frac{\theta_{\text{col}} - \theta_{\min}}{\theta_{\max} - \theta_{\min}} \ . \qquad (12)$$

Integrating this function over the range of possible initial relative positions, the probability that an event results in a
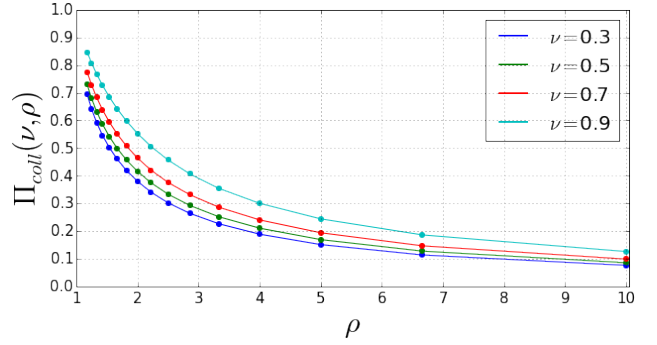

Figure 6: Probability of collision $\Pi_{\text{coll}}(\nu, \rho)$ with an object that creates regressive motion on the retina as a function of the ratio of vision radius to collision radius ρ, for different fly-object velocity ratios ν.

collision given that the object produces regressive motion on an fly's retina can be found as:

$$\Pi_{\text{coll}}(\nu, \rho) = \int_{\alpha_{\min}}^{\alpha_{\max}} \Pi_{\text{coll}}(\alpha, \nu, \rho) d\alpha \ , \qquad (13)$$

where $\alpha_{\min}$ is either 0 or the minimum value of α for which there exists a θ with which the object can produce a regressive motion on fly's retina, and $\alpha_{\min}$ is either 90 or maximum value of α for which there exists a θ with which the object remains observable to the fly.

We calculated the integral (13) numerically and show the results in Figure 6 for different values of fly-object velocity ratios ν and different collision range-vision range ratios ρ. As can be seen from Figure 6, for R_{vis}=60 mm (Zabala et al., 2012) and R_{coll}=15 mm (our assumption), the collision probability is around 0.2-0.3. This implies that if encounters are created randomly, regressive motion on the retina is not predictive of collision, and as a consequence it is unreasonable to expect that digital evolution will produce collision avoidance in response, as only 1 in 5 to 1 in 3 regressive motions actually lead to collisions. This was borne out in experiments, and we thus decided to bias the events in such a manner that *all* events that leave a regressive motion signature in the retina will lead to collision. Note that this is not necessarily an unrealistic assumption, as we have not analyzed a distribution of realistic "events" (such as is available in the data set of Branson et al. 2009). It could very well be that the way real flies approach each other differs from the uniform distributions that went into the mathematical analysis presented here.

Results

We conducted experiments with five different fitness functions representing different environments. Environments differ in the amount of fitness individuals gain when moving

and in the penalty incurred by a collision. Evolved agents use various strategies to avoid collisions and maximize the travelled distance, but one of the most successful strategies they use is indeed to categorize visual cues into regressive and progressive optic flows. We find that agents categorize these visual cues only in some regions of the retina: the regions in which collisions take place more frequently. They then use this information to cast a movement decision: they keep moving when seeing an object creating progressive optic flow on their retina, and stop when the object creates regressive optic flow on their retina. However, they do not stop for the entire duration of the event, i.e., the whole time they perceive regressive optic flow. Rather they stop during only a portion of the event, which helps the agent to avoid a collision with the object while maximizing their walking duration and hence gaining higher fitness.

The strategy of using regressive motion as a cue for collision (Chalupka et al., 2015), similar to the observed behavior in fruit flies (Zabala et al., 2012) evolves in our experimental setup under some environmental circumstances (discussed below). We refer to this strategy as regressive-collision-cue (RCC) and we define it in our experimental setup as follows:
1) The moving object produces *regressive* motion on the agent's retina during an event and the agent stops at least for some time during that event, or
2) The moving object produces *progressive* motion on the agent's retina during an event and the agent does not stop during that event. The number of events (out of 100) in which the agent uses this strategy is termed the "RCC value."

We now discuss the results of an experiment in which the RCC strategy has evolved. We take the most successful agent at the end of that experiment and analyze its behavior. This agent evolved in an environment with penalty-reward ratio of 2, meaning the penalty of each collision equals twice the maximum reward the agent can gain in 2 events. Figure 7 shows whether the agent stopped during an event, stop probability (blue triangles), as a function of the angular velocity of the image on the agent's retina for 100 events. In that figure, the angular velocity of the image on agent's retina is negative for regressive optic flow and positive for progressive events. Simulation units are converted to plotted values (in deg/s and mm/s) by equalizing dimensionless values ν, and ρ in simulation and actual values: R_{vis}=60 mm (Zabala et al., 2012), V_{fly}=20 mm/s (Zabala et al., 2012), R_{coll}=15 mm (our assumption). We can see from the figure that out of all 100 events, the agent did not stop during one event with regressive motion while for two progressive events, it stopped. In the remaining events the agent accurately uses the RCC strategy (resulting in an RCC value=97). The average velocity of the agent during each event is also shown (solid orange circles), which reflects the number of time steps the agent moves during that event (and thus indirectly how often it stops). For progressive motions, the stop probability is zero (the agent continues to move dur-
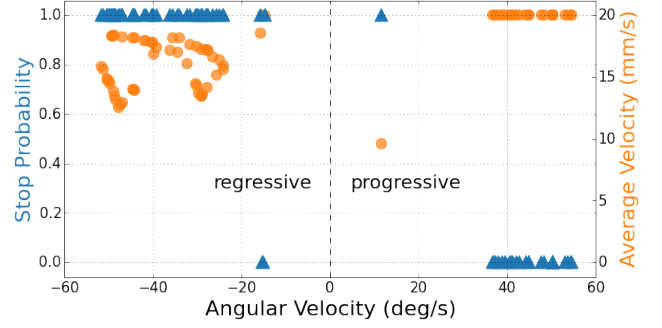

Figure 7: The stop probability of the evolved agent vs. the angular velocity of the image on its retina for 100 events. Positive values of angular velocity show progressive motion events and negative angular velocities stand for regressive motion events. The average velocity of the agent is also shown during each event.

ing the event) and thus the velocity of the agent is maximal during that event. For regressive optic flow (negative angular velocities), the average velocity during each event is less than maximum and for extreme angular velocities, as it only needs to stop for shorter durations to avoid collisions.

In order to quantitatively analyze how using regressive motion as a collision cue benefits agents to gain more fitness, we traced this particular agent's evolutionary line of descent (LOD) by following its lineage backwards for 50,000 generations mutation by mutation until we reached the random agent that we used to seed the initial population (see Lenski et al. 2003 for more details on how to construct evolutionary lines of descent for digital organisms). Figure 8 shows the fitness and the RCC value vs. generation for this agent's LOD. It is evident from these results that evolving this strategy benefits agents in gaining fitness compared to the rest of the population in this environment as high peaks of fitness occur at high RCC values and conversely, the fitness drops as the RCC value decreases. Nevertheless, this strategy does not evolve all the time. Figure 9 shows the fitness and RCC for all 20 replicates in the environment with penalty-reward ratio of 2. We can see that the mean fitness of all 20 replicates is around 20% less than the fitness of the agent that evolved the RCC strategy. The mean RCC value for all 20 replicates is also $\approx 20\%$ less than that of an agent that evolved the RCC strategy.

The difficulty to evolve the RCC strategy is not limited to the number of runs in which this behavior evolved out of all replicates in some environment (we also tried running the experiment for longer evolutionary times but the results do not change significantly). Environmental conditions also play a key role in the evolution of this behavior. Figure 10 shows the RCC value distribution for 20 replicates in five different environments. In order to calculate the RCC value in each replicate, we took the average of the RCC value in
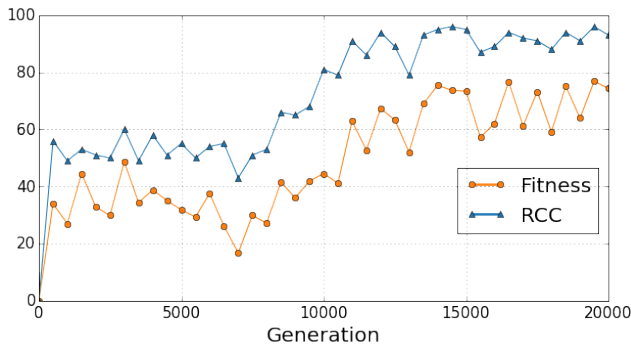
Figure 8: Fitness and regressive-collision-cue (RCC) value on the line of descent for an agent that evolved RCC as a strategy to avoid collisions. Only the first 20,000 generations are shown, for every 500 generations.
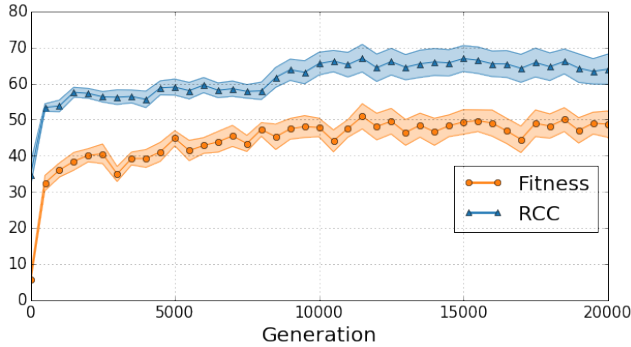

Figure 9: Mean values of fitness and regressive-collision-cue (RCC) over all 20 replicates vs. evolutionary time in the line of descent in the environment with penalty-reward ratio of 2. Standard error lines are shown with shaded areas around mean values. Only the first 20,000 generations are shown, for every 500 generations.

the last 1,000 generations on the line of descent to compensate for fluctuations. We observe that the RCC strategy only evolves in a narrow range of penalty-reward ratio, namely for PR=2 and PR=3. According to Figure 10, higher values of penalty on the one hand discourage the agents from walking in the environment (they simply choose to remain stationary), and therefore prevent them from exploring the fitness landscape. Lower values for the penalty, on the other hand, result in indifference to collisions and thus, the optimal strategy (probably the local optimum) in these environments is to keep walking and ignore all collisions. For lower values of the penalty, the RCC value is $\approx 55\%$, which means they evolve to stop in obvious cases that end up in collision (if they keep moving, the RCC value should be 50).

Discussion

We used an agent-based model of flies equipped with MNBs that evolve via a GA to study the selective pressures and environmental conditions that can lead to the evolution of collision avoidance strategies based on visual information. We specifically tested cognitive models that invoke "regressive motion saliency" and "regressive motion as a cue for collision" to understand how flies avoid colliding with each other in two-dimensional walks. We showed that it is possible to configure the experiment in such a manner that "regressive-collision-cue" (RCC) evolves as a strategy to avoid collisions. However, the conditions under which the RCC strategy evolved in our experiments are limited: the strategy only evolved in a narrow range of environmental conditions and even in those environments, it does not evolve all the time. In addition, we showed that from general principles, only a small percentage of events in which an agent perceives regressive optical flow eventually leads to a collision, so that RCC as a sole strategy is expected to have a large false positive rate, leading to unnecessary stops.

As discussed in the Methods section, our experimental implementation is biased in such a way that *all* regressive motion events lead to a collision if the agent does not stop during that event. If the moving object's velocity direction is distributed uniformly randomly in all directions, the probability that a regressive event ends up in a collision is rather low ($\approx 20\%$ in our implementations). Because the false positive rate of using regressive optical flow as the only predictor of collisions is liable to thwart the evolution of an RCC strategy, we biased our setup in such a way that the false-positive rate is zero, a bias that does not significantly influence the outcome of our experiments. Consider an environment in which only a percentage of events with regressive motion end up in collision. This is similar to an environment with a lower penalty for collisions (as long as the strategy evolves at all) since the agent's fitness is scored at the end of its lifetime (all 100 events) not during each event.

However, there is a difference between a lower percentage of collisions in regressive events and lower penalty for collisions, namely a lower probability of collision in regressive motion events is equivalent to a higher amount of noise in the cue that the agent takes from the environment, compared to the case of lower penalties for collision. In other words, if 100% of all regressive motion events lead to collisions, the agent associates regressive motion events with collisions with certainty. Thus, implementing the experiments with 100% collisions in regressive motion events is tantamount to eliminating the noise in sensory information, which generally aids evolution. Compensating for noise in sensory information could also be achieved if we scored agents in every single event, and informed them about their performance in that event (feedback learning). We did not use feedback learning here, but plan to do so in future experiments.

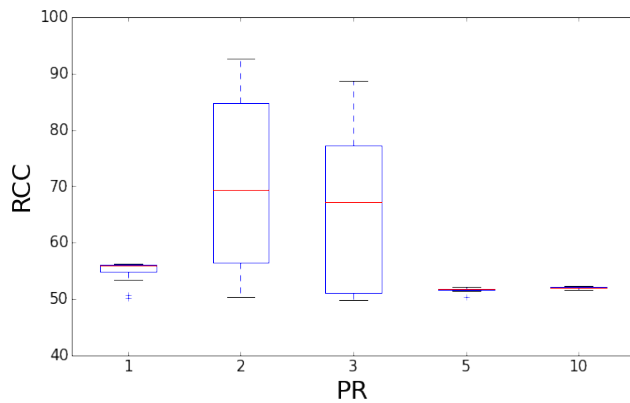We conclude that the evolution of "regressive motion

Figure 10: RCC value distribution in environments with different penalty-reward ratios. Each box-plot shows the RCC value averaged over the last 1000 generations on the line of descent for 20 replicates.

saliency" is unlikely to have happened only due to collision avoidance as the selective pressure. It is important to remember that walking is not the most frequent activity in fruit flies. Further, flies do not usually live in high density colonies and therefore do not find themselves on collision courses very often. It may be the case that components of this strategy (namely categorizing the optic flow as regressive or progressive) have evolved under different selective pressures entirely unrelated to the present test situation, and was further evolved to enhance collision avoidance with conspecifics while moving (a type of exaptation). For example, detecting predators is a strong selective pressure in the evolution of visual motion detection, including the categorization of that cue so as to take appropriate actions. It may be interesting to study the behavior of flies interacting with animals or objects that are not perceived as conspecifics.

Acknowledgements

References

Albantakis, L., Hintze, A., Koch, C., Adami, C., and Tononi, G. (2014). Evolution of integrated causal structures in animats exposed to environments of increasing complexity. *PLoS Comput Biol*, 10:e1003966.

Borst, A. and Egelhaaf, M. (1989). Principles of visual motion detection. *Trends Neurosci*, 12:297–306.

Branson, K., Robie, A. A., Bender, J., Perona, P., and Dick-inson, M. H. (2009). High-throughput ethomics in large groups of drosophila. *Nature Methods*, 6:451–457.

Chalupka, K., Dickinson, M., and Perona, P. (2015). Generalized regressive motion: A visual cue to collision. *arXiv preprint arXiv:1510.07573*.

Chapman, S., Knoester, D., Hintze, A., and Adami, C. (2013). Evolution of an artificial visual cortex for image recognition. In *Advances in Artificial Life, ECAL 12*, pages 1067–1074.

Edlund, J. A., Chaumont, N., Hintze, A., Koch, C., Tononi, G., and Adami, C. (2011). Integrated information increases with fitness in the evolution of animats. *PLoS Comput Biol*, 7:e1002236.

Hintze, A. and Miromeni, M. (2014). Evolution of autonomous hierarchy formation and maintenance. In *AL-IFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, pages 366–367.

Joshi, N. J., Tononi, G., and C., K. (2013). The minimal complexity of adapting agents increases with fitness. *PLoS Comput Biol*, 9:e1003111.

Lenski, R. E., Ofria, C., Pennock, R. T., and Adami, C. (2003). The evolutionary origin of complex features. *Nature*, 423:139–144.

Maloney, E. S. (1989). *Chapman Piloting, Seamanship and Small Boat Handling*. Hearst Marine Books.

Marstaller, L., Hintze, A., and Adami, C. (2013). The evolution of representation in simple cognitive networks. *Neural Comput*, 25:2079–2107.

McFarland, D. J. (1977). Decision making in animals. *Nature*, 269:15–21.

Olson, R. S., Haley, P. B., Dyer, F. C., and Adami, C. (2015). Exploring the evolution of a trade-off between vigilance and foraging in group-living organisms. *Royal Society Open Science*, 2:150135.

Olson, R. S., Hintze, A., Dyer, F. C., Knoester, D. B., and Adami, C. (2013). Predator confusion is sufficient to evolve swarming behaviour. *Journal of The Royal Society Interface*, 10:20130305.

Schiff, W., Caviness, J. A., and Gibson, J. J. (1962). Persistent fear responses in rhesus monkeys to the optical stimulus of "looming." *Science*, 136:982–3.

Zabala, F., Polidoro, P., Robie, A., Branson, K., Perona, P., and Dickinson, M. H. (2012). A simple strategy for detecting moving objects during locomotion revealed by animal-robot interactions. *Current Biology*, 22:1344–1350.

Eukaryo: An Agent-based, Interactive Simulation of a Eukaryotic Cell

Douglas Yuen[1] and Christian Jacob[1,2]

[1]Department of Computer Science, University of Calgary
[2]Department of Biochemistry & Molecular Biology, University of Calgary
dwkyuen@ucalgary.ca, cjacob@ucalgary.ca

Abstract

Eukaryo is a 3D, interactive simulation of a eukaryotic cell. In comparison to existing cell simulations, our model illustrates the structures and processes within a biological cell with increased fidelity and a higher degree of real-time interactivity using a virtual reality environment. Implemented in a game engine, *Eukaryo* is a hybrid model that combines agent-based and mathematical modelling.

Through the use of visual scripting, *Eukaryo* incorporates both agent-based modelling and mathematical representations to describe gene expression, energy production and waste removal within the cell in a highly visual, interactive simulation environment. With the help of virtual reality displays, users can be immersed in the crowded spaces of biomolecular worlds and observe metabolic reactions at a high level of detail. Compared to traditional media, such as illustrations and videos, *Eukaryo* offers superior representations of cellular architecture, its components and dynamics of the machineries of life.

Motivation

The concept of scaling humans to a molecular size to facilitate exploration of biological systems at the cellular level forms the premise for the 1966 film "Fantastic Voyage" (Fleischer, 1966; Asimov, 1988). A group of scientists explores the human body by means of a miniaturised submarine. If such journeys at molecular scale were feasible today, they would offer unparalleled experiences to explore the molecular universes of a biological cell. Illustrations of biomolecular worlds have been limited to a few prominent examples only. David Goodsell's "The Machinery of Life" is one such example (Goodsell, 2009). Meticulously constructed from X-ray crystallography, NMR and high-resolution electron micrographs, Goodsell's illustrations capture the densely packed environments inside a cell at the molecular level. Contrary to impressions from typical textbook illustrations, these cellular spaces are highly crowded. Accentuating the complexity of a cell, Goodsell suggests that every structure visible in his illustrations is likely supported and regulated by a myriad of other structures that are not visible.
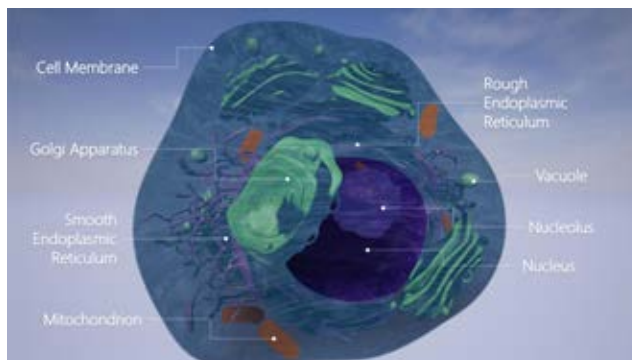

Figure 1: A Generic Eukaryotic Cell. The major organelles of a eukaryotic cell are replicated in the *Eukaryo* model.

Despite capturing the density of materials within a cell, renderings – even at the highly detailed level of Goodsell's illustrations – remain static and cannot depict how each of the different structures do interact with one another. As a result, textbook illustrations tend to be supplemented with videos that can portray the progression of sophisticated cellular biochemical reactions and pathways. The BioVisions video "Inner Life of a Cell" makes use of 3D computer animation to represent some of the key processes that occur in a eukaryotic cell, ranging from gene expression to cellular transport (Harvard BioVisions, 2007). Despite being more expressive, videos limit viewers to observing events from predetermined camera perspectives. Hence, videos do not permit exploration or interaction with the model.

In order to provide an interactive, exploratory environment that – to a certain degree of accuracy – captures the sense of complexity underlying the machinery of life, we have implemented *Eukaryo*, a virtual model of a generic eukaryotic cell (Fig. 1). Built using the game development software Unreal Engine (Epic Games, 2015), our model strives to capture the key structures and functioning units of a cell, similar to Goodsell's illustrations and the BioVisions animations. Furthermore, by utilizing virtual reality (VR) interfaces, *Eukaryo* enables users to interact with the simulation, navigate to different locations within the cell,

investigate its architecture, and explore molecular structures and metabolic pathways.

Using virtual reality visualizations and by combining the interactivity of a video game and artistic renderings of cellular structures with accurate representations of proteins (downloaded from the Protein Database (Westbrook and Fitzgerald, 2003)), *Eukaryo* can provide a sense of dynamics and immersion reminiscent of "The Fantastic Voyage" (Fig. 2). Not only is this a novel and engaging learning approach, this experience also provides a more effective learning method through computational models (Laha et al., 2014).

Related Work

While biological systems are still mostly studied *in vivo*, many processes often cannot occur in isolation from living systems or are too difficult to explore. Computer models overcome this constraint, allowing for individual processes to be observed in greater detail and providing solutions towards virtual experiments – ideally starting at the cellular level.

Cells   Constituting the foundational building block of any biological system, simulations of cells are of particular interest. Projects such as *E-CELL* (Tomita et al., 1999) and *Virtual Cell* (Loew and Schaff, 2001) provide the frameworks for modelling interconnected processes inside a biological cell. *E-CELL* can simulate signaling, cellular reactions and gene regulation, while *Virtual Cell* is intended to act as a more general platform for simulations of micro-biological systems.

Both *E-CELL* and *Virtual Cell* produce numerical outputs only. Actual visualization or illustration of how the system evolves over time is left to other tools and the user's imagination. Moreover, these models are limited in their interactivity: while they are running, users cannot visually observe a process or pause a simulation to inspect its current state.

More recent cell models have been built as sophisticated mathematical models for predictive simulations. For example, Karr et al.'s whole-cell model captures all known processes in the bacterium *M. genitalium* (Karr et al., 2012). While such models are powerful, they are also complex to set up. For instance, the whole-cell model requires twenty-eight separate modules running concurrently to represent one cell and is controlled by $1,900$ empirically-determined values as input parameters. In order to better manage such complex control structures, we utilize hierarchical visual scripting similar to pathway interaction diagrams.

Molecular Dynamics.   UnityMol (Lv et al., 2013) and MolecularRift (Norrby, 2015) have been developed using the Unity game engine to visualize molecules (Unity Technologies, 2012). MolecularRift works on the Oculus Rift (Oculus, 2015), whereas UnityMol has some virtual reality

Figure 2: The cellular space with control elements to adjust the level of detail and a minimap for location reference.

support, such as for immersive CAVE visualizations. These applications further demonstrate the opportunities offered by game engines for biomolecular visualization and simulation. These systems also show the benefit of virtual reality for molecular visualization. However, they are focused on molecular dynamics and drug design applications rather than full cell simulations.

Agent-based Modeling.   Agent-based methods have been used as simulation techniques complementary to purely mathematical models (Haefner, 2005). For example, a gene regulatory model of the λ-switch has been recreated in a 3D, purely agent-driven simulation (Jacob et al., 2006). A classic gene regulation model studied in *E. coli* bacteria, the lactose operon, has been implemented to illustrate the protein interactions that determine gene expression (Jacob and Burleigh, 2004, 2006). In comparison to other simulators, *LINDSAY Composer* offers a 3D, interactive environment where models can be directly constructed in virtual, 3-dimensional spaces (Jacob et al., 2012). The feasibility of such agent-based models in game engine-inspired environments has been demonstrated in an immune system simulation, purely based on interacting components (Sarpe and Jacob, 2013) and gene expression in *E. coli*, including transcription and translation (Esmaeili et al., 2015) as well as chemotaxis.

Implementating a Eukaryotic cell

Eukaryo is implemented in Unreal Engine (Epic Games, 2015) as a hybrid, interactive 3D model that combines agent-based modeling with mathematical techniques (using differential equations). Agent-based models (ABMs) simulate behaviours by defining a set of interactions between *agents* in a system. We adopt a general approach to define an agent as (1) a set of *situations* an agent may be in, (2) its set of *actions*, (3) all of the possible combinations of its *internal data* and (4) a *decision function* that triggers an action based on the situation and internal data (Afsharch et al., 2006). An
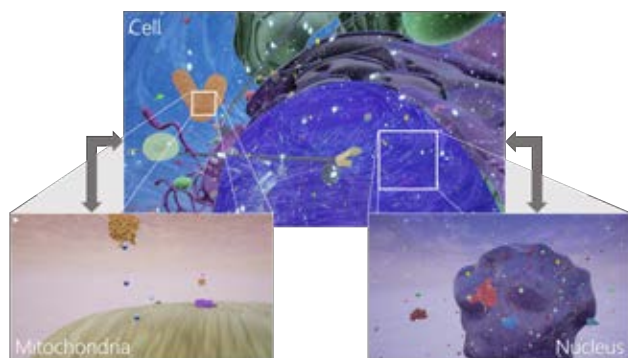
Figure 3: Level setup in *Eukaryo*. The simulation begins in the cytosol (Cell level) from which users may explore organelles such as Mitochondria and the Nucleus in greater detail, which are implemented as separate levels.

ABM constitutes the set of all agents, where the total activity of the agents in an environment forms the ABM's overall behaviour. As such, ABMs are suitable for describing biological systems because observations in biology can be translated into rules for agents within an ABM. Moreover, new information – such as higher levels of detail regarding biomolecular processes – can be incorporated into an ABM simply by adding new agents and/or rules (Jacob et al., 2012). Compared to mathematical models, where equations define system behaviour, ABMs are more difficult to validate and require more computational power to run. However, mathematical models are specific to the processes they formalize and tend to be harder to adapt for describing other systems (Haefner, 2005; Edelstein-Keshet, 1988). A hybrid model confers benefits of both agent-based and mathematical modelling (Esmaeili et al., 2015), complementing one another to produce more powerful, flexible and extensible simulations, as we are about to demonstrate with the *Eukaryo* model.

Cell Universes as Game Levels

In comparison to prokaryotic (i.e., bacterial) cells, eukaryotic cells are more complex, consisting of organelles that carry out specific functions (Figs. 1 and 2). Compartmentalization allows the cell to regulate distinct environments (Jékely, 2007). In *Eukaryo*, the simulation consists of three interconnected environments implemented as (game) levels (Fig. 3). Each level contains a set of Actors, to represent the cell as a whole (Cell) and two organelles (Mitochondria and Nucleus). In Unreal Engine, an Actor[1] is a collection of components that define its location, size and appearance. An Actor is inert, unless their *event graph* is implemented to specify its attributes. The permissible values for these attributes is the set of possible actions available to the Actor

---

[1]We use the term Actor, instead of agent, to refer to an actor entity within the Unreal Engine game programming system.
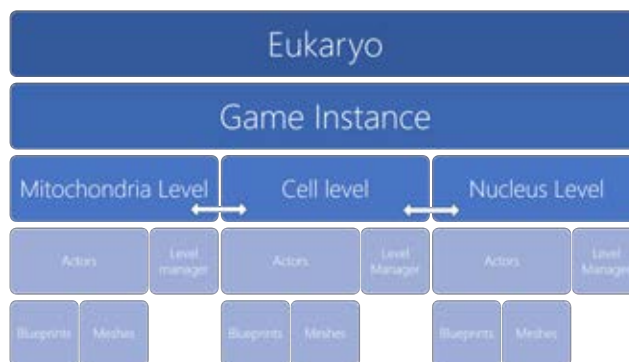

Figure 4: Modular Architecture of *Eukaryo*. The game instance stores the cell's state. Different levels receive information from the game instance to determine their local environment. Each level consists of a collection of Actors. Some Actors have meshes for graphical representation and Blueprints to specify their actions.

and the conditions that result in a particular action.

Actors can be affected by and affect their environment (Fig. 4): information applying to the entire level is stored within persistent data structures known as Game Instances, which enables users to transition between different levels to view cellular processes at different levels of detail (Fig. 3). Blueprint Actors can communicate with the Game Instances in a similar manner as they would with other Actors. Event graphs for each Blueprint Actor (Fig. 6[2]) make use of events, special conditions tracked by the game engine, such as collisions and frame updates that serve as triggers for interactions among agents.

Visual Scripting with Blueprints

Biochemical reactions proceed after collisions that bring molecules and enzymes in close proximity to each other. This forms the basis for all reactions within a cell (Gold, 2014). In a game engine, a biological reaction corresponds to a collision between two Actors that results in a state change in at least one Actor. While this makes it straightforward to describe reactions in terms of collision events, actual biological reactions occur at very high rates determined (and measured) by the densities of, e.g., substrate, enzyme and signal molecules in the cytosol. The concentrations of these reactants are high enough in the cell such that the probability of two reactants colliding and reacting is also high (in humans, each cell may have upwards of 1.7 billion proteins) (Milo, 2013). However, using a large number of Actors to reproduce such a high-density, reactive environment in a game engine is (currently) computationally unfeasible, especially if we want to execute our model on commodity computing devices.

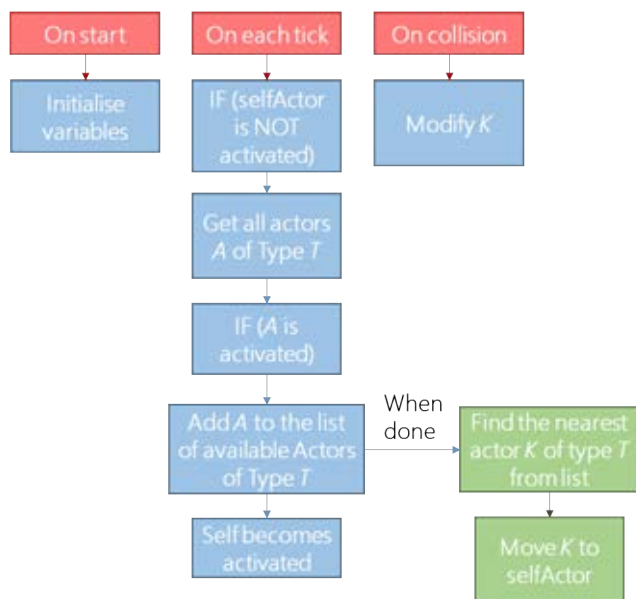[2]Details of this script are explained in a later section.

Figure 5: Layout for a Generic Blueprint. Every Blueprint in *Eukaryo* that an Actor uses follows a similar layout. After its variables are initialised, on every step in the simulation, the Actor will look for the nearest Actor A of type T. If ready for a reaction, both Actors will move towards each other. On a collision, they become activated and partake in the reaction.

Figure 6: Event Graph for Glycogen Phosphorylase implemented as a Blueprint visual script. While all Actors use similar blueprints, their behaviours may be further specified: for instance, in glycogen phosphorylase, the enzyme will only continue breaking down glycogen into G1P provided that the glycogen chain has not been consumed. After a certain period of inactivity, the enzyme will be broken down.

Case Studies

Using a game level approach in combination with visual scripting turns out to be highly flexible in composing pathways, including 3D visualizations, and enabled us to implement a number of key cell metabolism processes within *Eu-*
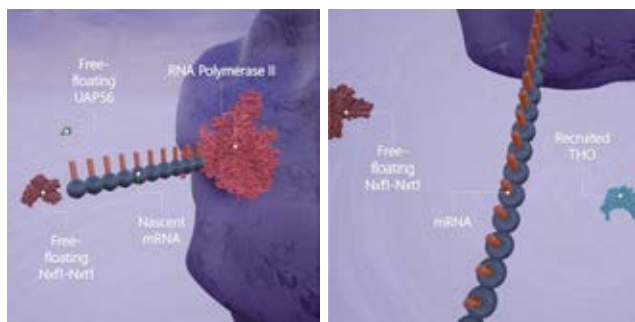
karyo. In this paper, we only describe mRNA transport, transcription, translation, glycogenolysis, and the effect of pH on carbonic anhydrase in greater detail. Yet, the underlying mechanisms have been applied to represent other processes within *Eukaryo* at similar levels of detail, such as peroxide decomposition and microtubule assembly.

Case Study 1: Transcription and Translation

Within the nucleus game level (Fig. 3), mRNA is represented by a single Actor with a preset number of static mesh components. An mRNA Actor spawned into the level makes its segments visible, one at a time, to mimic the synthesis of individual mRNA segments during transcription (Fig. 7a). Once all segments are visible, the entire strand detaches from RNA polymerase. Beginning with the THO complex (Fig. 7b), transport signals can now identify and bind to mRNA (Carmody and Wente, 2009). The THO Actor continuously looks through the level for any nascent mRNA to bind to. When a binding partner is found, the actor is moved to the mRNA, where, on collision, THO is attached to it. Now with bound THO, mRNA is ready for a UAP56 signal protein to bind (Fig. 7c). Once UAP56 is bound, ALY binds in a similar fashion (Fig. 7d). When Nxf1-Nxt1 binds, the other Actors are detached from the mRNA, which is then ready to be transported out of the nucleus and undergo additional processing for translation (Fig. 7e). Once an mRNA actor is transported into the cytosol and tagged by Gle1 and Dbp5, it is ready for translation (Fig. 7f).

Ribosomes carry out translation in the cell level. After a ribosome Actor finds an mRNA strand, it follows a spline along the mRNA and synthesizes nucleic acid chains that "fold" into a protein once the end of the spline is reached. The mRNA Actors are animated to improve their visual im-

As such, *Eukaryo* makes use of an alternative mechanism to facilitate an accurate yet illustrative representation of biochemical reactions. In Unreal Engine, each Actor has a *transform component* that keeps track of its location within a level. Other Actors may access this value, allowing for an Actor to compute its distance relative to any other Actor. Using the Blueprint Function Library common to all Actors, one can calculate an Actor's distance to all Actors of a certain type and determine the closest agent (Fig. 5). We use this information in conjunction with the *iTween* animation plugin to smoothly translate Actors from one point to another (Therriault, 2016). Together, this allows for different Actors to be moved rapidly towards other Actors and initiate a reaction without having to wait for molecule agents to collide with one another at random. While this is not a true representation of how chemical reactions proceed *in vivo*, it allows for reactions to be replicated more readily for illustrative purposes. The actual densities of highlighted proteins and other molecules are controlled by mathematical equations and visualized by particle systems as described below.

(a) mRNA transcription

(b) THO binds
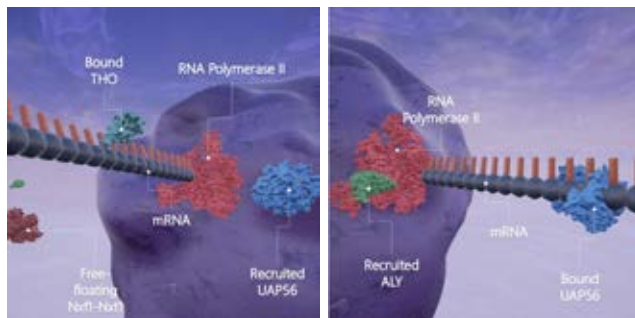
(c) UAP56 recruited

(d) ALY recruited

(e) Nxf1-Nxt1 signals export

(f) Gle1, Dbp5 binds

Figure 7: A step-by-step illustration of mRNA export.

(a) mRNA translation

(b) Amino acids fold into PKA
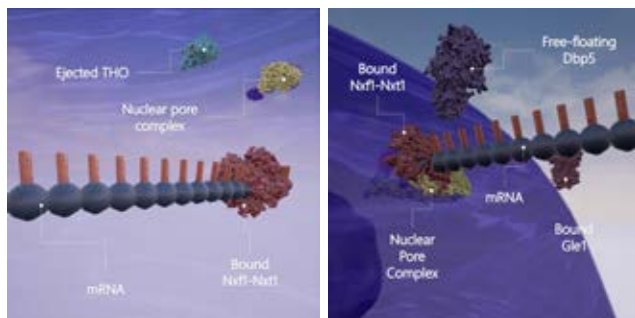

(c) PKA activates PK

(d) PK activates GP
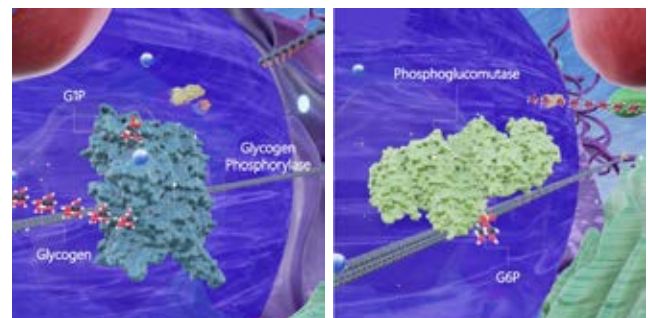
(e) GP breaks glycogen to G1P

(f) Conversion of G1P to G6P

Figure 8: Translation and Glycogenolysis.

pact within the simulation; its children mesh components are positioned along an animated spline. This spline also forms the path for the ribosome Actor to move along during translation (Fig. 8a). Following translation, exosome complex Actors "degrade" the mRNA so that mRNA does not accumulate within the cell (Fig. 9). Exosome complexes are responsible for mRNA degradation, removing inactive mRNA as a part of mRNA turnover (Makino et al., 2013).

Each mRNA Actor possesses an attribute that specifies which protein the ribosome produces during translation. In the current version of *Eukaryo*, mRNA transported into the cytosol ecodes for adenylase cyclase, protein kinase A (PKA) and phosphorylase kinase, and signal proteins for glycogenolysis. Adenlyase cyclase converts adenosine monosphosphate (AMP) into its cyclyic form, cAMP, activating PKA, which, in turn, activates phosphorylase kinase (Alberts et al., 2015). mRNA also codes for the glycogen

phosphorylase enzyme, which is activated by phosphorylase kinase and carries out glycogen breakdown, as described in the next section.

Case Study 2: Glycogenolysis

As a second case study, we present the replication of the glycogenolysis pathway. After translation (Fig. 8a) and protein folding (Fig. 8b), the signal proteins adenylase cyclase, protein kinase A, phosphorylase kinase and glycogen phosphorylase participate in the regulatory pathway that leads to the release of glucose-1-phosphate (G1P). We have implemented this pathway and describe it here in detail (Venkataraman and Luck, 1949).

Protein kinase A is activated by cyclic AMP (cAMP, a second messenger synthesised by adenylase cyclase) (Lodish et al., 2012). Once activated, protein kinase A phosphorylates phosphorylase kinase to stabilise it (Fig. 8c). In
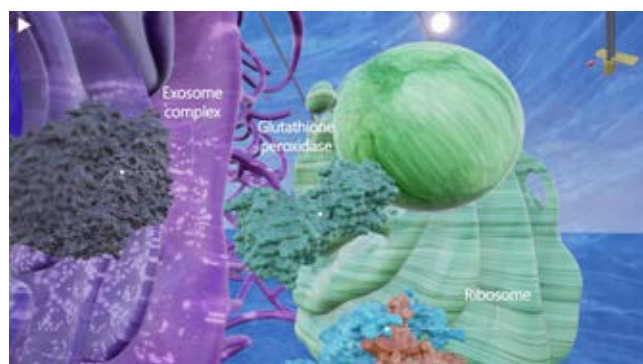
Figure 9: Glutathione Peroxidase within the Cell Level. An exosome complex can be seen (in grey) left of glutathione peroxidase.

turn, phosphorylase kinase activates glycogen phosphorylase (Fig. 8d), which can begin cleaving glucose units from glycogen: G1P is formed (Fig. 8e). Phosphoglucomutase converts G1P into glucose-6-phosphate (G6P), a reactant in glycolysis (Fig. 8f). The visual scripts to control these reactions are depicted in Figure 6.

Case Study 3: pH Effect on Carbonic Anhydrase

It has been found that varying the pH level in a eukaryotic cell's environment alters its enzymatic activity. This was also replicated in a mathematical model by Khalifah and Edsall (1972). We followed their reaction schema and implemented a C++ Blueprint Function that calculates the resulting Michaelis-Menten (pK_m) and catalytic (k_{cat}) constants based on the pH value (Fig. 10). The Michaelis-Menten constant pK_m is the substrate concentration that results in a reaction rate half of the maximum rate; inversely, pK_m measures a substrate's affinity for an enzyme. A smaller pK_m value corresponds to a reaction that approaches the maximum rate of reaction more quickly. The catalytic constant, k_{cat}, describes how fast an enzyme may react with substrates to form the product. Specific values for pK_m and k_{cat} can be calculated using rate constants as described in (DeVoe and Kistiakowsky, 1961).

The Blueprint Function is attached to a level manager actor that takes the environment pH to calculate pK_m and k_{cat} (Fig. 4). The values are sent to all of the carbonic anhydrase Actors in the scene, thus controlling their rate of reaction. Each carbonic anhydrase Actor has an attached particle emitter which releases bicarbonate ions at a rate corresponding to the calculated reaction rate.

As illustrated in Figure 10, we found that our model qualitatively aligns with the values reported in (Khalifah and Edsall, 1972). The pK_m and k_{cat} values in our model suggest higher affinities and reaction turnovers. Further adjustments to the experimental parameters will need to be made to provide more accurate representations.
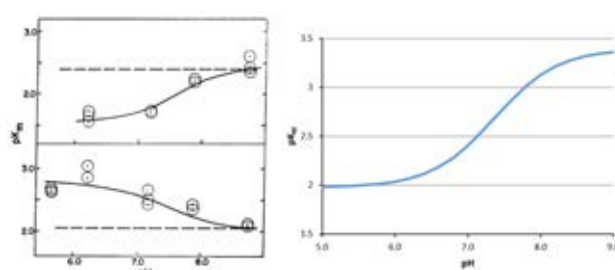
(a) pK_m values in response to varying pH
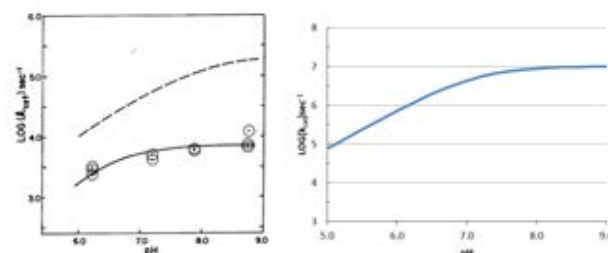
(b) k_{cat} values in response to varying pH

Figure 10: pH affecting carbonic anhydrase. Left: Khalifah and Edsall model; right: replicated *Eukaryo* model. This deterministic model controls the visual effects and average carbonic anhydrase agent concentrations (i.e., activity) in the simulated cell.

The inclusion of a Michaelis-Menten model in *Eukaryo* illustrates how mathematical models can be incorporated into agent-driven models: functions that keep track of reaction rates and concentration changes are used to manipulate attributes of individual agents, rather than directly influencing their decision functions. While our model is relatively simple, for now, this approach can be applied to more detailed models. For example, C++ libraries can be imported to construct functions that solve differential equations and generate biologically relevant outputs for predictive modelling, similar to those seen in programs such as Virtual Cell (Loew and Schaff, 2001) and other studies (Esmaeili et al., 2015; Sarpe and Jacob, 2013).

Virtual Cell Spaces

In a gaming environment with immersive visualization and real-time interaction, one has to think about how to visualize the "cellular universes", how to navigate through them and create convincing, yet scientifically justified effects to highlight the implemented biomolecular processes.

Visualisations

Similar to illustrations found in textbooks, organelles in *Eukaryo* have been given specific colours to ensure that they are distinguishable from one another, while simultaneously providing some indication of their function (Fig. 1). The nucleus and endoplasmic reticula are purple for proximity
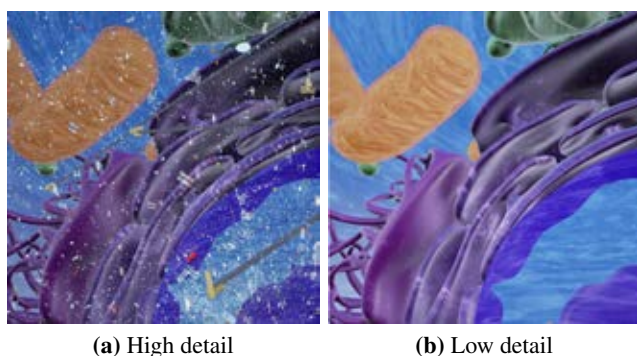
(a) High detail   **(b)** Low detail

Figure 11: Side-by-side comparison of the simulation at different levels of detail. (a) All of the elements are visible to capture the crowded cell environment. (b) All game objects and particle systems disabled, allowing users to focus on the cell's structures.

to the cell's core and involvement in biomolecule synthesis. The Golgi apparatus and vacuoles are green: both are involved in storage and transport. Mitochondria are orange, standing out from the other organelles. Together, they provide the context for the processes described in this paper.

Particle Systems. To convey that cellular spaces are highly packed, multiple particle systems are incorporated into *Eukaryo*. Typically used for representing transient entities like smoke and flames, particle systems can be configured to emit billboards, which are entities that always face the camera (Reeves, 1983). These billboards are used in *Eukaryo* to represent water molecules, sodium, potassium and chlorine ions, as well as albumin, fatty acids and carbon dioxide (Fig. 11). By utilizing the graphics processing unit (GPU) for rendering, more than a million particles can be displayed on screen without compromising performance. This means that the simulation remains responsive to user input at any point. Each particle has a random velocity, angular motion and lifespan to convey a sense of stochasticity to mirror Brownian motion.

Protein Data. In order to accurately represent each protein in *Eukaryo* we have imported their 3D structure from the Protein Databank (PDB) repository (Westbrook and Fitzgerald, 2003). A PDB file contains the coordinates, rotation and amino acid sequence in a protein. The resulting structural information can be converted into surface meshes and imported into 3D modeling software (such as Autodesk(Autodesk)). Before import into the game engine, we apply textures to the mesh to improve a structure's visibility in the cell. Hence, we can present protein structures consistent with accurate, empirical data.

Navigation and Level of Detail Controls

In comparison with other virtual environments that depict entities such as buildings and terrain, traditional notions of

up and down are not apparent within a cell as familiar reference points are absent. This makes it more difficult for users entering the virtual cell spaces to orient themselves using visual cues present in the cell alone. Therefore, additional navigational aids were implemented. A minimap is displayed in the lower left-hand corner, where a cursor indicates the current camera position (Fig. 2). A slider allows the user to hide or show varying levels of detail (Fig. 11). At the maximum level, all proteins and molecules are visible. A lower level of detail reduces the number of visible elements. Actors that are invisible still continue running in the background, so the simulation itself is never interrupted. Lastly, users can highlight and bring up information about any entity they are looking at or approaching within the cell.

Conclusion

We have described an implementation of *Eukaryo*, an interactive, hybrid 3D model of a eukaryotic cell created in the Unreal Engine game environment. Using an agent-based approach to describe biological processes as a series of shared interactions between Actors, our model becomes highly versatile. We have highlighted some of the pathways and processes implemented in *Eukaryo*: gene expression with transcription, translation, and mRNA transport; glycogenolysis and pH effects on carbonic anhydrase activity. Built on a game engine architecture, the *Eukaryo* system incorporates mathematical models as additional components, thus facilitating and further extending the expressiveness and accuracy of the simulations. By combining state-of-the-art biological modelling with 3D visualization and real-time interactivity, *Eukaryo* provides an immersive cell model that can serve as a learning tool as well as an environment to perform virtual experiments.

The *Eukaryo* software, videos and virtual experiments are available on the Lindsay Virtual Human website.

References

Afsharch, M. et al. (2006). Ontology-Guided Learning to Improve Communication. In *AAMAS 2006*, pages 923–930, Hakodate, Hokkaido, Japan.

Alberts, B. et al. (2015). *Molecular Biology of the Cell*. Garland Science, Taylor & Francis Group, LLC, sixth edition.

Asimov, I. (1988). *Fantastic Voyage*. Bantam.

Autodesk. Maya 3D Animation Software.

Carmody, S. R. and Wente, S. R. (2009). mRNA nuclear export at a glance. *Journal of Cell Science*, 122(12):1933–7.

DeVoe, H. and Kistiakowsky, G. B. (1961). The enzymic kinetics of carbonic anhydrase from bovine and human

erythrocytes. *Journal of the American Chemical Society*, 83(2):274–280.

Edelstein-Keshet, L. (1988). *Mathematical models in biology*. McGraw-Hill.

Epic Games (2015). Unreal Engine 4 Documentation.

Esmaeili, A. et al. (2015). PROKARYO: an illustrative and interactive computational model of the lactose operon in the bacterium Escherichia coli. *BMC Bioinformatics*, 16(1):311.

Fleischer, R. (1966). *Fantastic Voyage*. Twentieth Century Fox Film Corporation.

Gold, V. (2014). International Union of Pure and Applied Chemistry Compendium of Chemical Terminology. *Iupac*, page 1670.

Goodsell, D. S. (2009). *The Machinery of Life.* Springer Science, New York, USA, second edition.

Haefner, J. W. (2005). *Modeling Biological Systems:: Principles and Applications*. Springer Science & Business Media.

Harvard BioVisions (2007). The Inner Life of the Cell (Video).

Jacob, C. and Burleigh, I. (2004). Biomolecular swarms: an agent-based model of the lactose operon. *Natural Computing*, 3(4):361–376.

Jacob, C. and Burleigh, I. (2006). Genetic programming inside a cell. *Genetic Programming Theory and Practice III*, 9:191–206.

Jacob, C. et al. (2006). Swarms and genes: Exploring λ-switch gene regulation through swarm intelligence. In *IEEE Congress on Evolutionary Computation*.

Jacob, C. et al. (2012). LINDSAY Virtual Human: Multiscale, Agent-based, and Interactive. In Kolodziej, J., Ullah Khan, S., and Burczynski, T., editors, *Advances in Intelligent Modelling and Simulation*, pages 327–349. Springer, Berlin.

Jékely, G. (2007). *Eukaryotic Membranes and Cytoskeleton*, volume 607. Springer, New York, 1st edition.

Karr, J. R. et al. (2012). A whole-cell computational model predicts phenotype from genotype. *Cell*, 150(2):389–401.

Khalifah, R. G. and Edsall, J. T. (1972). Carbon dioxide hydration activity of carbonic anhydrase: kinetics of alkylated anhydrases B and C from humans (metalloenzymes-isoenzymes-active sites-mechanism). *PNAS*, 69(1):172–6.

Laha, B. et al. (2014). Effects of VR system fidelity on analyzing isosurface visualization of volume datasets. *IEEE Transactions on Visualization and Computer Graphics*, 20:513–522.

Lodish, H. et al. (2012). *Molecular Cell Biology*. W.H. Freeman and Company, seventh edition.

Loew, L. M. and Schaff, J. C. (2001). The Virtual Cell: a software environment for computational cell biology. *Trends in Biotechnology*, 19(10):401–6.

Lv, Z. et al. (2013). Game On, Science - How video game technology may help biologists tackle visualization challenges. *PLoS ONE*, 8(3):e57990.

Makino, D. L., Halbach, F., and Conti, E. (2013). The RNA exosome and proteasome: common principles of degradation control. *Nature Reviews. Molecular Cell Biology*, 14(10):654–60.

Milo, R. (2013). What is the total number of protein molecules per cell volume? A call to rethink some published values. *BioEssays*, 35(12):1050–1055.

Norrby, M. (2015). MolecularRift, a Gesture Based Interaction Tool for Controlling Molecules in 3-D. Master's thesis, Department of Design Sciences, Lund University, Sweden.

Oculus (2015). Oculus Rift VR Headset.

Reeves, W. T. (1983). Particle systems—a technique for modeling a class of fuzzy objects. *ACM SIGGRAPH Computer Graphics*, 17(3):359–375.

Sarpe, V. and Jacob, C. (2013). Simulating the decentralized processes of the human immune system in a virtual anatomy model. *BMC bioinformatics*, 14(Suppl 6):1–26.

Therriault, J. (2016). iTween for Unreal Engine.

Tomita, M. et al. (1999). E-cell: software environment for whole-cell simulation. *Bioinformatics*, 15(1):72–84.

Unity Technologies (2012). Unity Manual.

Venkataraman, J. and Luck, J. M. (1949). Phosphoglucomutase: II. Mechanism of Action. *Journal of Biological Chemistry*, 179(2):569–75.

Westbrook, J. D. and Fitzgerald, P. M. D. (2003). The PDB format, mmCIF formats, and other data formats. In Bourne, P. E. and Weissig, H., editors, *Structural Bioinformatics*, chapter 8, pages 161–179. Wiley-Liss Inc., 2nd edition.

A Boolean network model for invasive thyroid carcinoma

Jesús Espinal-Enríquez [1,2], Raúl A. Mejía-Pedroza [1], and Enrique Hernández-Lemus[1,2]

[1] Computational Genomics Division, National Institute of Genomic Medicine, México
[2] Center for Complexity Sciences, Universidad Nacional Autónoma de México
ehernandez@inmegen.gob.mx

Abstract

Thyroid cancer is a common endocrine system neoplasm characterized by being extremely heterogeneous and of unexplained incidence (idiopathic). Some subtypes of thyroid cancer are more aggressive than others and for this reason treatment needs to be differential. Nonetheless, due to its inherent variability, prognosis based on pathology and/or biochemical profiling often fails leading to a delay in proper therapeutics that increases significantly the associated mortality. The most aggressive thyroid tumors are characterized by an increase in the destruction of extracellular matrix, this is done by the matrix metalloproteinases (MMPs). The regulation of MMPs is finely tuned by several molecules, but the dynamical mechanisms which control this pathway are still unknown. Here, based on detailed molecular interaction information coming from functional tests and gene expression experiments, we develop a boolean model of the matrix metalloproteinases pathway in thyroid cancer. By observing steady state conditions perturbing the network by simulating a specific drug, we find that TNFA could be a major target of this pathway. The approach performed here could allow to understand the finely regulated process to maintain extracellular matrix homeostasis.

Introduction

Thyroid carcinomas (TCs) are the most common endocrine-related cancers. In recent years, the incidence of TCs has drastically increased, at the same time mortality rates remain largely unchanged (Davies and Welch, 2006; Veronese et al., 2015). Sub-optimal diagnostics have lead to thousands of TC-related deaths annually, in particular for the case of poorly-differentiated, anaplastic and medullary cancers whose ethiology remains to be fully disclosed (Giuffrida and Gharib, 2000; Hernández-Lemus and Mejía, 2012). A number of environmental and nutritional factors have been statistically associated to TCs. It is believed that any process leading to compensatory increases in the hormone thyrotropin will increase the risk of thyroid tumors. Apart from this, an elevated risk has been documented for women who use estrogen for gynecological reasons, especially those in pre-menopause stages (Ron et al., 1987; Hima and Sreeja, 2015).

One of the main challenges to thyroid neoplasms prognostics and therapeutics is the enormous variability that the tumors present both at the cellular morphology and at the gene expression levels (Espinal-Enríquez et al., 2015). Tumor heterogeneity generates two main classes of problems for clinicians: first of all the determination of the specific type of tumor since fine-needle aspiration cytology is conclusive in only around 20% of the cases. Secondly, determination of the most aggressive tumor subtypes (usually anaplastic and some papillary) to decide which patients are candidates for pharmacological therapy with close follow-up and which ones should undergo invasive and costly surgical procedures (Baudin and Schlumberger, 2007).

It is known that thyroid carcinomas present quite complex patterns of evolution (Espinal-Enríquez et al., 2015). The entangled dynamics of these tumors is shaped by a non-trivial interplay of genomic and epigenomic changes including not only mutations and gene expression changes but also effects of copy number variations, gene translocations, methylation deregulation and altered signaling pathways. A detailed analysis of all the afore-mentioned features seems unattainable, however a complex systems approach based on integrating most of this information into a simplified dynamic model may allow us to have a global but coarse-grained view of the phenomenology behind thyroid cancer evolution.

Histologically, thyroid carcinomas are classified into follicular adenoma (FA), follicular thyroid cancer (FTC), papillary thyroid cancer (PTC) and anaplastic thyroid cancer (ATC). FTC and PTC are differentiated tumors with low risk of recurrence and good prognosis. On the other hand, ATC is more aggressive, usually diagnosed at an advanced stage, therefore frequently leading to a fatal prognosis (Baudin and Schlumberger, 2007). A transition from FTC to PTC and PTC to ATC has been reported depending on the differential expression of molecules related to degradation of Extracellular matrix (Espinal-Enríquez et al., 2015).

A relevant feature present in the most aggressive thyroid carcinoma and also a well known hallmark of cancer (Hanahan and Weinberg, 2000, 2011) is the invasiveness and migra-

tion of tumor cells via degradation of extracellular matrix (ECM) (Gu et al., 2005; Ii et al., 2006). The components of ECM, collagen, elastin and gelatin, are degraded by the matrix metalloproteinases (MMPs). The activity of MMPs are negatively controled by the tissue inhibitors of matrix metalloproteinases (TIMPs). So far, four TIMPs have been described and charaterized: TIMP-1, TIMP-2, TIMP-3 and TIMP-4. The interplay between MMPs and TIMPS in normal conditions maintains the balance of extracellular matrix. However, in metastatic tumors, overexpression of MMPs falls into an exacerbated destruction of ECM and the consequent invasion of adjacent tissues Jacomasso et al. (2014). MMPs are also negatively regulated by other molecules, such as A2M, TSP2, TFPI2 and RECK (Krady et al., 2008; Oh et al., 2001). MMPs have positive regulation by TGFB (Kim et al., 2004), plasmin (Ramos-DeSimone et al., 1999) or furin (Remacle et al., 2006). At the same time, other molecules participating in the matrix metalloproteinases pathway are a1CT, a1PI, a2AP, which regulate the levels of plasmin; ADAM17, LRP1, and ELANE (Figure 1).

The dynamics of this pathway is still unknown. Understanding the temporal behavior of all those elements in this pathway becomes crucial to have a better image of the ECM maintenance in cancer. To achieve this, to construct a model which captures the dynamics of the participants in the MMP pathway is necessary. Here we construct a Boolean network model of the MMPs pathway (Figure 1), based on the known relationships among components. We also use a previous work (Espinal-Enríquez et al., 2015) in which we propose a transition between thyroid carcinoma subtypes depending on the expression of the molecules involved in the MMP pathway.

In the next section we will introduce the main theoretical tool we will use to tackle with this problem, namely a discrete dynamics model for the MMP network. Such class of models represent a dynamical systems approach to simulate the time evolution of cellular phenotypes constrained by a set of cellular biochemical processes subject to environmental influences. After that, we will present the specific model of a Boolean dynamical network of thyroid cancer invasiveness, whose main results, in particular those related with a classification of the more aggressive (and invasive) phenotypes for TC, will be discussed in the following section. Finally we will further elaborate on the biomedical implications of our findings and the usefulness of such dynamical systems approaches to understand complex diseases.

Discrete network dynamics

Several approaches have been developed to model the dynamic evolution of complex systems based on partial knowledge about them. One particularly powerful yet somehow simple approach is given by sequential dynamical systems, in particular discrete dynamical networks (Kauffman, 1969; Aldana and Cluzel, 2003) also known as Boolean networks.

Such systems possess mathematical features that may their implementation computational effective for medium-sized problems also providing stable results. At the same time, discrete dynamical networks allow for the codification of functional information (such as the one obtained by experimental data) in the form of *logical rules* (Zinovyev et al., 2012) which can be curated and refined by a close interplay between experimentalists and theoreticians (Espinal et al., 2011). In concrete, a Boolean model is constructed based on the assumption that any element in the network may have two possible dynamical states: active or inactive; open or closed; expressed or inhibited; etc. These values can be translated to 1 or 0. The dynamical state of any node will depend on the state of its regulators, which are other nodes in the network directly connected to it.

In a formal description, the dynamical state of the network consists of a set of N discrete variables $\{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, each representing the aforementioned dynamical state of the node. The state of each node σ_n (0 or 1) is determined by its set of regulators. We denote as $\sigma_{n_1}, \sigma_{n_2}, \ldots, \sigma_{n_k}$ the k regulators of σ_n. Then, at each time step the dynamical state of σ_n will be given by

$$\sigma_n(t+1) = F_n\left(\sigma_{n_1}(t), \sigma_{n_2}(t), \ldots, \sigma_{n_k}(t)\right),$$

where F_n is a regulatory function constructed by taking into account the activating/inhibiting nature of the regulators. Each node has its own regulatory function. It is worth noticing that the curation of the set of logical rules was firmly grounded on the data-driven analysis of whole-genome gene expression experiments (Espinal-Enríquez et al., 2015) via global statistics and causal network inference (Krämer et al., 2014).

Steady state conditions

For Boolean networks (finite number of nodes which take a finite number of values), all initial conditions lead to a periodic behavior where the network configuration is replicated after a certain number of steps. This pattern of repeated network configurations is called an *attractor*. For the same network, several attractors may coexist. The total amount of initial conditions which reach a particular attractor is called the *basin of attraction*. The time required to reach this condition is known as the *transient time* and the number of iterations between the repeated configurations is the *period of the attractor*. These values reflect global properties of the network. In this case the steady states of the network allow to us to identify the temporal behavior of particular phenotypes, by observing the interplay between metalloproteinases and their regulators. Furthermore, the Boolean approach that we have implemented is also capable to determine the steady state conditions after elimination of one node (*in silico* knock outs). This has been done to observe the global properties of the perturbed network simulating the action of a directed drug, the ultimate goal of this model.

571

Figure 1: **The regulatory network for thyroid cancer invasiveness.** A) This biochemical pathway takes place in the plasma membrane as well as in the extracellular space. Here, the main molecules involved in the pathway are depicted. B) The graph consistent with the minimal network representing the invasive phenotype response in thyroid cancer.
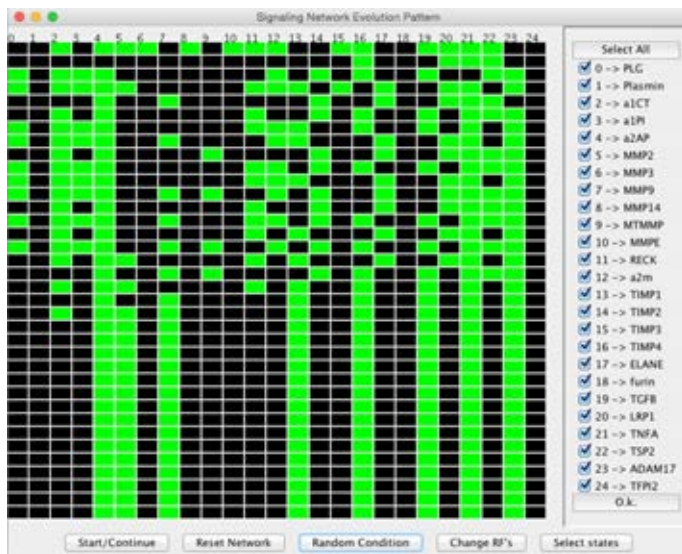
Figure 2: **Network evolution pattern.** Columns represent the states in terms of molecule activity: Green = active, Black = inactive or absent. Rows represent discrete time steps. Time runs from above to below. As it can be observed, after a small set of iterations, the network reaches a steady state condition (black and green squares do not change anymore). This is the attractor for that particular initial condition (first row).

Results

A Boolean dynamic network model for thyroid network invasiveness

Gene regulatory networks have been used previously to characterize the complexity associated with thyroid cancer phenotypes, such as malignancy associated with mechanisms of cell death resistance (Hernández-Lemus and Mejía, 2012). It has also been discussed the important role that extra-cellular matrix (ECM) maintenance and repair processes play in the development of invasive tumors which are the more aggressive forms of thyroid carcinomas (Espinal-Enríquez et al., 2015). Among such processes it was established that the regulation of ECM remodeling by the family of Matrix metalloproteinases (MMPs) and their inhibitors is of foremost importance.

For this network, calculated over all initial conditions (2^{24}), the dynamics converges to 10 different attractors (Figure 3). Two out of the ten attractors are period three, meanwhile the other eight are punctual attractors. For clarity, figure 3 contains only 4 attractors, two of period 1 (3A and 3B), and two of period 3 (3C and 3D).
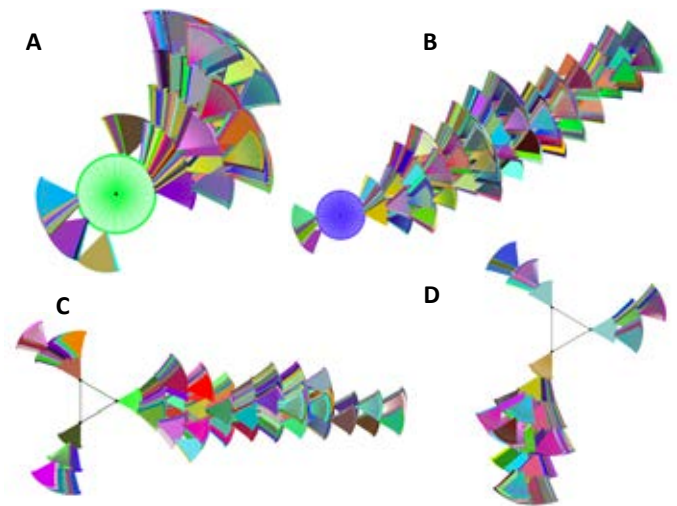

Figure 3: **Dynamic landscape for different attractors.** This representation shows the basin of attraction (fan-like structures). By observing from the outside of the figure to the center, each point of the fan-like structure represent one configuration of the network, meanwhile the lines correspond to time steps of the network dynamics. Two dots are connected if one of them is a successor of the other under the dynamics. In this sense, the fan-like structures are a set of network configurations which converge to one dynamical network state. Eventually, all configurations converge to an attractor. For panels A and B, the attractor is a single-point attractor, whereas Panels C and D present three-state attractors. The colors of links are different for graphical purposes only.

Elimination of nodes remarks the relevance of TIMP regulation

As we mentioned before, the interplay between MMPs and their inhibitors TIMPs, is the most relevant feature to determine whether the ECM is compromised or not. To quantify this parameter, we construct an invasiveness score (*IS*) which consists on the ratio between MMPs and TIMPs state values once an attractor has been reached. We eliminate one node of the network for performing the dynamics and observe whether the *IS* grows or decreases. The *IS* after elimination of each node is presented in table 1.

As it can be observed, the *IS* changes depending on which element was eliminated. Elimination of the node corresponding to TIMP-1 presents the largest *IS*, meanwhile the elimination of TNFA present the less aggressive situation. This last result is not intuitive, however, the network dynamics shows this is the most important node to have less invasiveness. This could be relevant in the context of directed therapies targeting the nodes whose elimination cause a decreasing in the invasiveness.

Table 1: **Invassiveness score *IS* after elimination of one node in the network dynamics.** The value of the Wild type network *IS* (with all nodes present) is bold. An *IS* above the WT represent dynamics which are more invasive after elimination of that node, meanwhile lower *IS* means the deletion of the node decreases the invasiveness and concomitant destruction of the ECM. It is worth to mention that the less aggressive dynamics is observed by eliminating the tumor necrosis factor α, (TNFA).

knock-out node	*IS*
TIMP-1	2.2
TIMP-4	2.2
a2m	1.333333333
TIMP-2	1.25
TIMP-3	1
MMP9	0.928571429
LRP1	0.882352941
TSP2	0.882352941
ELANE	0.85
ADAM17	0.833333333
MMP2	0.777777778
WT	**0.77173913**
PLG	0.764705882
MMPE	0.764705882
Plasmin	0.764705882
TFPI2	0.764705882
a1CT	0.764705882
a2AP	0.764705882
MMP3	0.764705882
MT-MMP	0.705882353
RECK	0.666666667
Furin	0.666666667
MMP14	0.6
TGFB	0.538461538
a1PI	0.486486486
TNFA	0.384615385

Dynamical regime of the network exhibits criticality

Dynamical Regime Discrete networks can operate in three different dynamical regimes: ordered, critical and chaotic (Derrida and Weisbuch, 1986; Aldana and Cluzel, 2003). These regimes are characterized by how perturbations are propagated across the network and also by how these perturbations modifies or not the network dynamical state. In the ordered regime, the network is not sensitive to perturbations. In the chaotic regime, a small perturbation often generates a perturbation avalanche that grows in time. However, in the critical regime, small perturbations neither increase nor decrease in time. A more profound description of the dynamical regime of discrete networks can be found in (Espinal et al., 2011).

To determine the dynamical regime in which this network is operating, it can computed the Derrida map $M(x)$. This mapping relates the size of the perturbation avalanche at two consecutive time steps: $x(t+1) = M(x(t))$. It is known that $S = \left.\frac{dM(x)}{dx}\right|_{x=0}$, (the slope of this map at $x = 0$), determines the dynamical regime: ordered if $S < 1$, chaotic if $S > 1$ and critical if $S = 1$ (Espinal et al., 2011; Balleza et al., 2008; Derrida and Weisbuch, 1986; Aldana and Cluzel, 2003).

Quite remarkably, the Derrida map of the MMP network shows with high accuracy that this network operates in the critical regime. This is evident from Fig. 4, where it is shown that the slope at the origin is close to 1. Systems operating close to a critical point have remarkable properties that would be very difficult to understand in the absence of criticality. In particular, a property of regulatory networks operating close to criticality relevant to the present study, is the interplay between robustness and adaptability observed in this network. Maintenance of the homeostasis of ECM is a major issue in the cell, since the stability of the tissue structure is highly dependent of the status of the ECM. It is necessary to have the mechanisms to degrade ECM but, at the same time, with a strong and sophisticated mechanism for a correct negative feedback of them. The property of criticality remarks the fact that this network can evolve with those characteristics (robustness under perturbations and evolvability to change under certain conditions) with a high accuracy. Therefore, the critical dynamics revealed in figure 5 is indicative of an optimized mechanism of degradation and maintenance of extracellular matrix via the interplay of MMPs and TIMPs as well as their regulators.

Perturbation of the network remarks loss of criticality

As we mentioned in the previous section, in a Derrida plot, pairs of initial states are sampled at defined initial distances, $H(0)$, from the entire state space, and their mean Hamming distance, $H(t)$, after a fixed time, t, is plotted against the initial distance $H(0)$. For this case, $t = 1$. The curve above/below the line (slope), $H(1) = H(0)$, reflects instability/stability, respectively (Kauffman, 1969).

To investigate the significance of each node in the network we calculated the perturbation measure for the individual nodes, by modifying the Derrida plot. Perturbation calculations were performed between the normal network and each of the 24 mutated networks. A mutated network for a specific node contains forced (0) value for that specific node in the input and output states, therefore it contains 2^{n-1} dynamical states. For perturbation calculation of each individual node, we perform a modified Derrida plot by measuring This modified Derrida plot highlights the effect of a directed drug whose target is a particular node in the network. The perturbation calculations were normalized with respect to the slope, $H(t) = H(0)$. In figure 5 it is shown the modified Derrida map according to (Gupta et al., 2007; Espinal et al., 2011) in which after elimination of one node, the Derrida map is plotted and shows the dynamical regime of the network under a perturbation. The most important nodes to achieve the critical regime are those shown in the figure. Loss of them cause a chaotic behavior. As it can be observed, the nodes which maintain the dynamical regime are mainly the TIMPs and MMPs. Elimination of them in the pathway cause a disregulation which is also observed under cancer phenotypes.
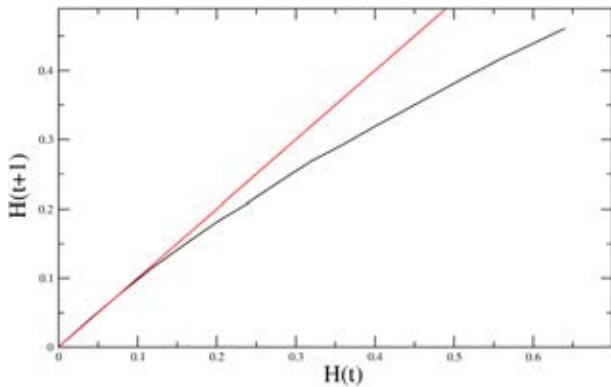

Figure 4: **Critical dynamics for the MMPs signaling network.** Plot of the Derrida map $M(x)$ which relates the size of the perturbation cascade at two consecutive time steps. The convergence of this mapping to a stationary value under successive iterations, determines the dynamical regime in which the network operates. This figure relates an initial separation $x(t)$ against separation $x(t + \delta(t))$, ($\delta = 1$), averaged over all states which are initially separated by $x(t)$. The slope of the curve near the origin is practically 1 in a sizeable neighborhood of the origin, an indicative that this network operates in the critical regime.

Loss of criticality as a global property of the network may be indicative for a severe damage of the network which impedes the recovery from a perturbation or, on the other hand, loss of flexibility under certain conditions. Those nodes that

cause the chaoticity are thus relevant in the context of global maintenance of the dynamical features of the network.

Discussion

Thyroid cancer is an important disease that involves several processes related to remodeling of extracellular matrix, which becomes in invasiveness and migration of tumor cells. The main mechanisms which govern the interplay between matrix metalloproteinases and TIMPs is not fully understood yet. In this work we constructed a Boolean network model of the MMPs pathway, we observed the steady state conditions for the wild type network as well as networks without one node, to simulate the action of a specific drug or a mutation in one of those elements. We also developed an invasiveness score *IS*, which reflects the ratio between action of TIMPs and MMPs once the steady state conditions have been reached. We observe the property of criticality in the WT network as well as the loss of this property after elimination of some particular nodes, mainly TIMPs and MMPs.
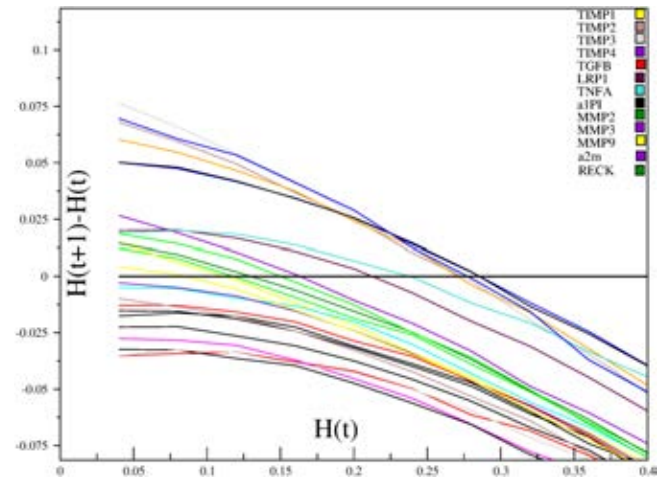
Figure 5: **Derrida modified plots.** Analogously to the Derrida map shown in Figure 5, this modified map relates the size of perturbations at two consecutive time steps. Curves above/below zero (black horizontal line) reflect instability/stability, respectively. Curves above zero represent those networks whose the knock-out produces a chaotic regime. Curves close to zero remain in the critical regime after the knock-out of a node. Those curves which do not change their dynamical regime after node elimination, indicate that the deleted node is not relevant to maintain the dynamics. That is the reason for which it can be eliminated. The most relevant nodes to preserve the dynamics of the network are shown in the upper right part of the figure.

Elimination of nodes in this work was implemented systematically in order to find crucial elements for the progres-

sion of the disease. This implementation could be applied elsewhere in larger networks to find critical nodes which are capable of determine the behavior of the whole network.

An important result is that the smallest *IS* was obtained after eliminating the Tumor necrosis factor α, TNFA. This result can be a promising therapy against the migration process that occurs during the most aggressive thyroid carcinomas. Therapies regarding blockage of TNFA has been developed for other pathologies, such as Reumathoid arthritis Brenner et al. (2015); Keffer et al. (1991). An opportunity to anti-TNF therapy could be opened with this study. Other approaches to find crucial nodes in a boolean network have been developed Kim et al. (2013). An interesting open question is whether the results observed here could be also obtained with other methodologies. That is matter of further research.

To our knowledge, this is the first time that a discrete theoretical model is implemented to understand the matrix metalloproteinases pathway, and furthermore, the particular dynamics of thyroid carcinoma progression. The finding of the TNFA as a crucial element for progression of this carcinoma could only be achieved with an approach such as the presented here.

It is worth to mention that the criticality exhibited by the WT network is consistent with the fact that most biological networks operate in this regime (Balleza et al., 2008; Shmulevich et al., 2005). Moreover, the loss of the property after node elimination could be explained as a loss of the equilibrium between those elements which degrade the ECM and those which maintain the basal levels of MMPs in order to preserve the homeostasis of Extracellular matrix.

This kind of approaches give to us a more accurate insight of how the temporal behavior of any biochemical network can be observed. It is worth to mention that despite the majority of reaction rates among the pathway elements are not known, the boolean modeling only needs the qualitative nature of the relationships. This is one of the greatest advantages of this coarse-grained approach. Experimental procedures must be performed to corroborate the results observed here. Notwithstanding, the boolean network developed in this work could suggest directed experiments in order to understand the complex nature of the invasiveness on thyroid cancer.

References

Aldana, M. and Cluzel, P. (2003). A natural class of robust networks. *Proceedings of the National Academy of Sciences*, 100(15):8710–8714.

Balleza, E., Alvarez-Buylla, E. R., Chaos, A., Kauffman, S., Shmulevich, I., and Aldana, M. (2008). Critical dynamics in genetic regulatory networks: examples from four kingdoms. *PLoS One*, 3(6):e2456.

Baudin, E. and Schlumberger, M. (2007). New therapeutic approaches for metastatic thyroid carcinoma. *The lancet oncology*, 8(2):148–156.

Brenner, D., Blaser, H., and Mak, T. W. (2015). Regulation of tumour necrosis factor signalling: live or let die. *Nature Reviews Immunology*, 15(6):362–374.

Davies, L. and Welch, H. G. (2006). Increasing incidence of thyroid cancer in the united states, 1973-2002. *Jama*, 295(18):2164–2167.

Derrida, B. and Weisbuch, G. (1986). Evolution of overlaps between configurations in random boolean networks. *Journal de physique*, 47(8):1297–1303.

Espinal, J., Aldana, M., Guerrero, A., Wood, C., Darszon, A., and Martínez-Mekler, G. (2011). Discrete dynamics model for the speract-activated ca 2+ signaling network relevant to sperm motility. *PloS one*, 6(8):e22619.

Espinal-Enríquez, J., Muñoz-Montero, S., Imaz-Rosshandler, I., Huerta-Verde, A., Mejía, C., and Hernández-Lemus, E. (2015). Genome-wide expression analysis suggests a crucial role of dysregulation of matrix metalloproteinases pathway in undifferentiated thyroid carcinoma. *BMC genomics*, 16(1):1.

Giuffrida, D. and Gharib, H. (2000). Anaplastic thyroid carcinoma: current diagnosis and treatment. *Annals of Oncology*, 11(9):1083–1089.

Gu, Z.-D., Li, J.-Y., Li, M., Gu, J., Shi, X.-T., Ke, Y., and Chen, K.-N. (2005). Matrix metalloproteinases expression correlates with survival in patients with esophageal squamous cell carcinoma. *American Journal of Gastroenterology*, 100(8):1835–1843.

Gupta, S., Bisht, S. S., Kukreti, R., Jain, S., and Brahmachari, S. K. (2007). Boolean network analysis of a neurotransmitter signaling pathway. *J Theor Biol*, 244(3):463–469.

Hanahan, D. and Weinberg, R. A. (2000). The hallmarks of cancer. *cell*, 100(1):57–70.

Hanahan, D. and Weinberg, R. A. (2011). Hallmarks of cancer: the next generation. *cell*, 144(5):646–674.

Hernández-Lemus, E. and Mejía, C. (2012). Inference and analysis of apoptotic pathways in papillary thyroid cancer. *Thyroid Cancer: Diagnosis, Treatment and Prognosis. Hauppauge, NY: Nova Science Publishing*.

Hima, S. and Sreeja, S. (2015). Modulatory role of 17β-estradiol in the tumor microenvironment of thyroid cancer. *IUBMB life*.

Ii, M., Yamamoto, H., Adachi, Y., Maruyama, Y., and Shinomura, Y. (2006). Role of matrix metalloproteinase-7 (matrilysin) in human cancer invasion, apoptosis, growth, and angiogenesis. *Experimental Biology and Medicine*, 231(1):20–27.

Jacomasso, T., Trombetta-Lima, M., Sogayar, M. C., and Winnischofer, S. M. (2014). Downregulation of reversion-inducing cysteine-rich protein with kazal motifs in malignant melanoma: inverse correlation with membrane-type 1-matrix metalloproteinase and tissue inhibitor of metalloproteinase 2. *Melanoma research*, 24(1):32–39.

Kauffman, S. (1969). Homeostasis and differentiation in random genetic control networks. *Nature*, 224:177–178.

Keffer, J., Probert, L., Cazlaris, H., Georgopoulos, S., Kaslaris, E., Kioussis, D., and Kollias, G. (1991). Transgenic mice expressing human tumour necrosis factor: a predictive genetic model of arthritis. *The EMBO journal*, 10(13):4025.

Kim, E.-S., Kim, M.-S., and Moon, A. (2004). Tgf-β-induced upregulation of mmp-2 and mmp-9 depends on p38 mapk, but not erk signaling in mcf10a human breast epithelial cells. *International journal of oncology*, 25(5):1375–1382.

Kim, J., Park, S.-M., and Cho, K.-H. (2013). Discovery of a kernel for controlling biomolecular regulatory networks. *Scientific reports*, 3.

Krady, M. M., Zeng, J., Yu, J., MacLauchlan, S., Skokos, E. A., Tian, W., Bornstein, P., Sessa, W. C., and Kyriakides, T. R. (2008). Thrombospondin-2 modulates extracellular matrix remodeling during physiological angiogenesis. *The American journal of pathology*, 173(3):879–891.

Krämer, A., Green, J., Pollard, J., and Tugendreich, S. (2014). Causal analysis approaches in ingenuity pathway analysis. *Bioinformatics*, 30(4):523–530.

Oh, J., Takahashi, R., Kondo, S., Mizoguchi, A., Adachi, E., Sasahara, R. M., Nishimura, S., Imamura, Y., Kitayama, H., Alexander, D. B., et al. (2001). The membrane-anchored mmp inhibitor reck is a key regulator of extracellular matrix integrity and angiogenesis. *Cell*, 107(6):789–800.

Ramos-DeSimone, N., Hahn-Dantona, E., Sipley, J., Nagase, H., French, D. L., and Quigley, J. P. (1999). Activation of matrix metalloproteinase-9 (mmp-9) via a converging plasmin/stromelysin-1 cascade enhances tumor cell invasion. *Journal of Biological Chemistry*, 274(19):13066–13076.

Remacle, A., Rozanov, D., Fugere, M., Day, R., and Strongin, A. (2006). Furin regulates the intracellular activation and the uptake rate of cell surface-associated mt1-mmp. *Oncogene*, 25(41):5648–5655.

Ron, E., Kleinerman, R. A., Boice, J. D., LiVolsi, V. A., Flannery, J. T., and Fraumeni, J. F. (1987). A population-based case-control study of thyroid cancer. *Journal of the National Cancer Institute*, 79(1):1–12.

Shmulevich, I., Kauffman, S. A., and Aldana, M. (2005). Eukaryotic cells are dynamically ordered or critical but not chaotic. *Proceedings of the National Academy of Sciences of the United States of America*, 102(38):13439–13444.

Veronese, N., Luchini, C., Nottegar, A., Kaneko, T., Sergi, G., Manzato, E., Solmi, M., and Scarpa, A. (2015). Prognostic impact of extra-nodal extension in thyroid cancer: A meta-analysis. *Journal of surgical oncology*, 112(8):828–833.

Zinovyev, A., Fourquet, S., Tournier, L., Calzone, L., and Barillot, E. (2012). Cell death and life in cancer: mathematical modeling of cell fate decisions. In *Advances in Systems Biology*, pages 261–274. Springer.

PajaroLoco: A suite of programs to study complex adaptive properties of animal language. An example of a Cassin's vireo syntax network.

Héctor Manuel Sánchez Castellanos[1] and Edgar E. Vallejo [1] and Charles E. Taylor [2]

[1]Computer Science Department,Tecnológico de Monterrey, Estado de México, 52926, México.
[2]Department of Ecology and Evolutionary Biology, University of California, Los Angeles, California, 90095, USA.
sanchez.hmsc@itesm.mx

Introduction

Animal language may be regarded as a complex adaptive system. Although there is software for identifying linguistic units or "phrases" in animal vocalisations, packages for analysing the grammatical properties of phrases sequences are scarce. *PajaroLoco* is an open-source *Mathematica* package for the study of these features. This paper is a demonstration of the capabilites of *PajaroLoco* using as an example the syntax network from a Cassins Vireo individual. Our intention is to illustrate how it could be used for other network analysis tasks in the artificial life community. It should be noted, though, that it is not our purpose to perform a thorough description of the species grammar as we are currently working on other aspects of that field of study (Arriaga et al., 2015; Hedley, 2015).

Bird song Analysis

We will describe one song from a Cassins Vireo male, "Mine" (sample id: 1156), singing 559 phrases on 13th of May of 2013 (for a description of such songs and a link to recordings see Hedley (2015)). An excerpt of the annotated sample is: *au, aj, ak, ai, ..., da, cj, ch, ci*; where each pair of letters represents a distinct sound composition (these phrases were obtained *a priori* by analysing its spectrogram with different software) and the position of the phrases in the sequence denotes temporal relationships (*au* occurs before *aj*, *ak* before *ai*, etcetera). All of the routines that will be shown can be performed directly in *PajaroLoco* (Sanchez et al., 2015).

Network Representation and Small-World Themes: The phrase sequences in the song were first represented as a network. This was achieved by setting the phrases as vertices in the graph and the transitions between them as weighted edges (an animation with sound of how this is done can be seen on the project's youtube playlist: https://www.youtube.com/playlist?list=PLRzY6w7pvIWrQXIcnxN5tKtpIVXjhqOlu). In figure 1a we can see the resulting network. It is easy to observe the tendencies of certain phrases to appear in communities or "themes". How tightly these themes are
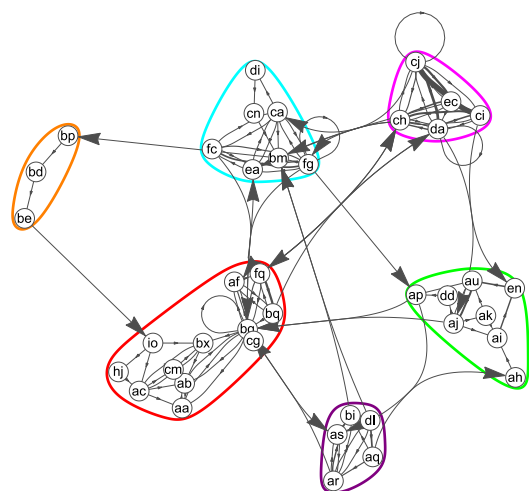
(a) Network representation of the transition frequencies of the song. Vertices are the phrases of the song and edges represent transitions. The coloured groups represent the themes or communities of phrases.
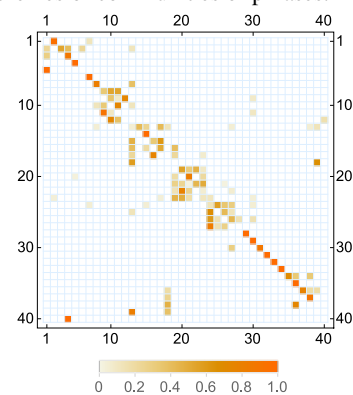
(b) Markov transitions probability matrix. The numbers on the frame are the identifiers of the phrases and the transition probability is represented by a color scale.

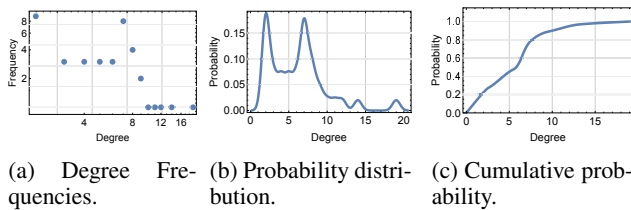Figure 1: Markov and network representation of the song's transitions.

(a) Degree Frequencies.

(b) Probability distribution.

(c) Cumulative probability.

Figure 2: Vertices degrees plots.

connected can be measured with the "small-worldness" coefficient which in this case is: 3.23. A random graph would have a "small-worldness" of approximately 1 so the song of the bird is not structured in a completely random manner. Instead, phrases are organised into clusters in which a collection of phrases are often used together for a period of time. We also obtained the Markov transitions matrix (figure 1b). In it we can see how probable is that the bird vocalises a certain phrase after he has produced another given one. We can observe once again that phrases follow a pattern and are not sung at random. Hedley et al. (2016) have analysed these data further, and suggested that the best representation is someplace between a first- and second-order Markov process, though some transitions may be more complex.

Morphological Analysis: While most of the network analysis takes place in undirected networks some measures deal with the in and out-degree of the vertices. In this case they are grouped in patterns or "motifs" according to how they are connected to other phrases (bottlenecks, hourglasses, one-ways and branches (Sasahara et al., 2012; Cody et al., 2015)). In our example we used the mean number of edges (3.05) in the network as the threshold for the detection of structures (this is a rather arbitrary way to define the threshold for demonstration purposes but the parameter can be easily changed if desired). Hourglasses and one-ways were the most common with 9 phrases while bottlenecks and branches were less frequent with 6 phrases.

Degree Distribution: Another important feature to study in graphs is whether or not they conform to certain degree distribution patterns. Figure 2 shows three plots that were used for this purpose. We see in figure 2a that most phrases have few connections although we have another peak around seven connections hinting at a bi-modal distribution. This can be further observed in figure 2b. We can also see in figure 2c that most of the phrases (around 90%) fall between one and eight connections.

Conclusions and Future Work

Analysing animal language is important from a computational and complex adaptive systems point of view. Taylor and Cody remark this importance by emphasising the different varieties in which bird's vocalisations come and how they compare with other complex phenomenon such as

cellular automata (Taylor and Cody, 2015). *PajaroLoco* is a tool developed for that purpose. Our program is part of an ongoing project and as such is updated and documented regularly. Although its main application is intended to be the analysis of annotated animal vocalisations, the package presented here can be used for the study of other complex adaptive phenomena in artificial life research, specially in those applications where phenomena can be described as sequences of elements and are amenable to network analysis.

References

Arriaga, J. G., Sanchez, H., Vallejo, E. E., Hedley, R., and Taylor, C. E. (2015). Identification of Cassin's Vireo (Vireo cassinii) individuals from their acoustic sequences using an ensemble of learners. *Neurocomputing*, pages 1–14.

Cody, M. L., Stabler, E., Sánchez Castellanos, H. M., and Taylor, C. E. (2015). Structure, syntax and small-world organization in the complex songs of California Thrashers (Toxostoma redivivum). *Bioacoustics*, 4622(November):1–14.

Hedley, R. W. (2015). Composition and sequential organization of song repertoires in Cassin's Vireo (Vireo cassinii). *Journal of Ornithology*, 157(1):13–22.

Hedley, R. W. et al. (2016). Complexity, Predictability and Time Homogeneity of Syntax in the Songs of Cassin's Vireo (Vireo cassinii). *Plos One*, 11(4):e0150822.

Sanchez, H., Vallejo, E. E., and Taylor, C. E. (2015). PajaroLoco: A suite of programs to study the grammatical structure of bird songs. In *Proceedings of the Twentieth International Symposium on Artificial Life and Robotics*, volume 2015, pages 148–153.

Sasahara, K., Cody, M. L., Cohen, D., and Taylor, C. E. (2012). Structural design principles of complex bird songs: a network-based approach. *PloS one*, 7(9):e44436.

Taylor, C. E. and Cody, M. L. (2015). Bird song: A model complex adaptive system. In *Proceedings of the Twentieth International Symposium on Artificial Life and Robotics*, volume 2015, pages 2–7.

Artificial Chemistries

Jordan Algebra AChems:
Exploiting Mathematical Richness for Open Ended Design

Penelope Faulkner[1,3], Angelika Sebald[1,3] and Susan Stepney[2,3]

[1]Department of Chemistry, University of York, UK
[2]Department of Computer Science, University of York, UK
[3]York Centre for Complex Systems Analysis
pf550@york.ac.uk

Abstract

We identify some desired mathematical properties of bonds in an Artificial Chemistry (AChem) that promote complexity and open-ended behaviour (i.e. an AChem not designed to display particular behaviours). We identify the underlying structures created by different properties of mathematical products. We use these to exploit existing algebra to generate a potentially open-ended subsymbolic AChem (ssAChem). We give examples of how our approach leads to interesting behaviour, focused on the structure of composite particles within our system.

A Low Level Approach to Artificial Chemistries

Most Artificial Chemistries (AChems) seek to produce a system capable of displaying specific behaviours associated with abiogenesis, the transition from inorganic to organic (living) materials (Hutton, 2002; Lucht, 2012; Suzuki et al., 2003). Those systems succeed in generating their particular behaviours because that is what they are designed to do. Another approach is to consider that we are seeking open-ended behaviour in our systems. In order to design for open-ended behaviour we need to approach the problem in an open-ended way.

We need to design a system that is rich and complex, with properties that allow us to define all the reactions of our AChem implicitly. We can then start looking for, and finding, behaviours that are emergent from the design, rather than engineered explicitly. We need a set of building blocks and connectors that do not limit the structure we design. Think of this as the difference between a prefabricated house and a brick house. A prefab has pieces that are specifically designed to fit together and form a house, and have a limited capability to do anything else. A brick house is just the bricks and the mortar that joins them. The bricks are not limited to building a certain house, or even a house of a particular size. With enough bricks and mortar the possibilities are endless. Likewise in an open-ended AChem the only limit should be the material and the amount of energy in the system.

We need to consider desirable properties of the interactions of our particles, rather than of the whole system, while ensuring that we do not over- or under-constrain the AChem. Here we do this by taking a mathematical approach, and taking advantage of existing mathematical theory and structures. This allows us to discuss not just the properties and behaviours of the particles, but also the different links and linking structures between them. We can then use established mathematics that has many emergent properties with interesting forms of interactions. We can also expand our view to talk about the effects of these properties on the system as a whole.

Terminology

Dittrich et al. (2001) define an AChem as a triple (S, R, A), where S is the set of possible molecules, R the set of rules for binding molecules, and A an algorithm describing the dynamics of the environment.

Rather than talking of 'molecules', we refer to the members of S as *particles*; these are either atomic particles (*atoms*) or composite particles (*composites*). Rather than talking of 'bonds', we say that the rules R say how particles can be joined together with *links*; links can be broken to decompose composite particles. We use this terminology to help prevent confusion between the properties of real chemical molecules and our AChem particles, and to prevent the abuse of chemistry terminology.

Faulconbridge et al. (2010) introduce the concept of *subsymbolic* AChems (ssAChem), with an example based on RBN-world. Such AChems have an implicit rule set where the properties used by the rules emerge from the internal structure of the particles. RBN-world was further developed in (Faulconbridge et al., 2010; Faulconbridge, 2011).

Here we demonstrate how the algebraic properties of the chosen rule set can be exploited to help obtain rich structures, and demonstrate this with an ssAChem based on a Jordan algebra of Hermitian matrices.

Mathematical Properties and Structure of Composite Particles

Our ssAChem rules have two parts: a set of mathematical products (or mathematical operations) for forming links and composites, and a set of probabilities used to determine probability of a reaction. In this section we discuss the properties of the mathematical product.

In mathematics there are two properties of a product on a set that are easily defined, and that can be indicative of many further properties of an algebra. These are associativity and commutativity.

$$\text{Associativity:} \quad (a \circ b) \circ c = a \circ (b \circ c) \quad (1)$$
$$\text{Commutativity:} \quad a \circ b = b \circ a \quad (2)$$

When we have a binary product, thereby linking two particles, combinations of these properties lead to four distinct structures, Table 1.

For an associative, commutative binary product we can change the order of evaluation and the ordering within any evaluation. No matter how we link a given set of particles, we get the same result. The structure is a bag. For an associative, non-commutative binary product we can change the order of evaluation such that there is no ordering on the products, but we cannot change the ordering within the product; the structure is a string.

Associativity is an assumed property of most algebras. Non-associative algebras, while rare, normally appear in an applied setting. They have been used in connection with genetics (Reed, 1997) and physics (McCrimmon, 1978) as well as a broad range of applications to mathematical theory (Gonzalez and Martinez, 2003). One of their main attractions is that with their enforced evaluation order they can embody a loose form of time, or at least an ordering of interactions.

For a non-associative, commutative product we can re-order particles in a product, but we have an enforced order of products. The structure is a binary tree, with unordered child nodes. For a non-associative, non-commutative product, we have an enforced order of products and ordering of particles within the products. The structure is a graph, with complicated directionality restrictions requiring labelling on both edges and nodes; these are not simple structures and do not conform to any of the normally used graph subtypes.

Let us consider these four structures in terms of an AChem. A bag has no internal structure, and limits us to a set of composite particles with the cardinality of the power set of the component particles. In real chemistry there are *isomers*: molecules with different inherent properties despite containing the same atoms in different arrangements (Muller, 1994). Isomers add complexity and increase the size of the combinatorial space. An AChem with a bag structure has no equivalent of isomers, so we do not want to base ours on an associative commutative product.

Associativity	Commutativity	Structure
Yes	Yes	Bag
Yes	No	String
No	Yes	Tree
No	No	Graph

Table 1: Summary of structure provided by different mathematical properties

Strings are structures that have received a lot of attention in the computing community, but they are rather simple mathematical objects that lack room for expansion. They have very simple combinatorial power of

$$C_{n-1} = \frac{(2n-1)!}{n!(n-1)!} \quad (3)$$

Strings support analogues of isomers, but there are not many of them. There is also no ordering of operations, so how they are formed does not affect the result. So we reject associative non-commutative products.

The tree structure given by the non-associative commutative product not only has more room for expansion to larger trees, it also has an implicit ordering. Because we cannot change the order of operations we get a variety of structures, and a system in which structure is as important as the building blocks themselves. This gives us a system with greater intrinsic flexibility.

The graph structure of a non-associative non-commutative product provides yet more structure, but makes it hard for the product to have any regularity to exploit as it allows so many possible structures. It is not necessary that we work with a structure this complex so we stick to trees.

Larger products

We can look beyond binary products to products that take more arguments, combining multiple particles with a common link.

This does not affect the structure of the system if we have an associative non-commutative product: since it is associative this changes nothing and we still have a string. However in the case of the non-associative commutative product as we expand from a binary product to a larger product we move from a binary tree to a general tree.

There are a larger number of possible trees with $n \geq 4$ leaves than strings with $n \geq 4$ elements. For $n = 3$ we have $s_3 = 6$ and $t_3 = 4$, where s_n is the number of strings with n elements and t_n is the number of trees with n leaves. For $n = 4$ we have $s_4 = 24$ and $t_4 = 31$ using products of any size, see Figure 1. We can show that from this point onward there is a larger number of possible trees than strings.

The number of possible strings increases with n such that $s_{n+1} = s_n(n+1)$. For trees we have a faster growth. We can

(a) $1\times$ (b) $6\times$
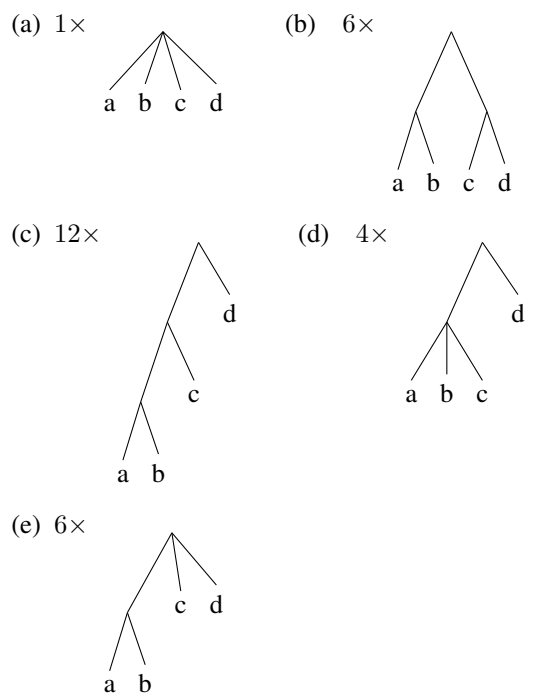
(c) $12\times$ (d) $4\times$

(e) $6\times$

Figure 1: Tree structures with four leaves with multipliers indicating the number of relevant rearrangements of leaves, giving an indication of all possible trees with four leaves with in this system.

show that if we link the extra element to the result of each of the graphs with n nodes with a binary link then the new element can be swapped with any of the other elements to give at least $t_{n+1} \geq t_n(n+1)$. We also always have more graphs as this does not include the graph of the $(n+1)$-product (see Figure 2) making t_{n+1} strictly greater than $t_n(n+1)$. Thus as we have more trees at $n = 4$ and a faster growth in the trees than in the strings, for $n \geq 4$ we always have more possible trees than strings.

In terms of an AChem, these properties show that we have a more interesting selection of possibilities in a non-associative system than otherwise, and these possibilities are controlled by the order in which reactions occur. Hence we focus on non-associative commutative products for our ssAChem design.

Jordan Algebras

Having established that these mathematical properties are desirable, we need to find a system in which we have these properties. Mathematics as a field has already found and studied systems with such properties, in the case of non-associative commutative systems we have Jordan Algebras (McCrimmon, 2006).

Jordan Algebras were originally conceived to find a solution to describing observables in quantum mechanics, but
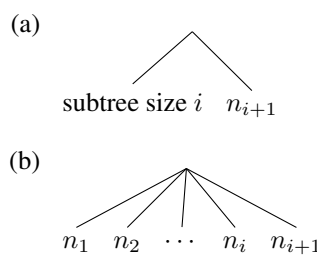

Figure 2: Trees showing greater growth than strings

were later discarded for that purpose because none of the Jordan Algebras were able to solve the problem. They have two important properties which define them:

Jordan identity: $\quad (x \bullet y) \bullet x^{\bullet 2} = x \bullet (y \bullet x^{\bullet 2})$ (4)

where $x^{\bullet n} = x \bullet x \bullet \cdots \bullet x$ (n times)

Power associative: $\quad x^{\bullet m} x^{\bullet n} = x^{\bullet(m+n)} \quad \forall m, n \geq 0$ (5)

Power associativity tells us what happens when we work with just one kind of particle.

There are several Jordan Algebras (McCrimmon, 2006). Here we take the most accessible Jordan Algebra that exists over the Hermitian matrices (a matrix is Hermitian if it equals its Hermitian conjugate, see Equation 14).

With this Jordan Algebra we start with a binary product formed of familiar matrix multiplication and addition to define the Jordan product:

$$X \bullet Y := \tfrac{1}{2}(XY + YX) \quad (6)$$

As one can see $X \bullet Y = Y \bullet X$. It is also non-associative:

$$(X \bullet Y) \bullet Z = \tfrac{1}{2}(XY + YX) \bullet Z \quad (7)$$
$$= \tfrac{1}{4}(XYZ + YXZ + ZXY + ZYX) \quad (8)$$
$$\neq \tfrac{1}{4}(XYZ + XZY + YZX + ZYX) \quad (9)$$
$$= \tfrac{1}{2}X \bullet (YZ + ZY) \quad (10)$$
$$= X \bullet (Y \bullet Z) \quad (11)$$

One of the advantages of a non-associative algebra is the ability to expand from the binary product and the binary tree it creates to a general product and its general tree. We can expand the binary product linearly to give the Jordan triple product:

$$\{X, Y, Z\} = (X \bullet (Y \bullet Z) + (X \bullet Y) \bullet Z - (X \bullet Z) \bullet Y)$$
$$= \tfrac{1}{2}(XYZ + ZYX) \quad (12)$$

We can further extend this to an arbitrary length n product, called an n-tad in Jordan theory (McCrimmon, 2006):

$$\{X_1, X_2, \cdots, X_n\} = \tfrac{1}{2}(X_1 X_2 \cdots X_n + X_n \cdots X_2 X_1)$$
$$(13)$$

Using the n-tad notation, $(X \bullet Y) = \{X, Y\}$.

Commutativity of this product means that we can fully reverse the order of the elements in the product, but not freely rearrange the order completely. So there is a large number of possible n-tad products for a particular set of n objects, increasing our combinatorial power and the ability of our system to exploit some properties of composite particles. Thus Jordan Algebras equip us with products that are open-ended, and are applicable to the open set of Hermitian matrices.

Mathematical Objects

Other AChems have used 'matrices' as the basis of their set S. In particular, the binary string chemistry (Banzhaf, 1993), dubbed the matrix-multiplication chemistry by Dittrich et al. (2001), makes use of matrix multiplication. However, it does not treat its particles as mathematical objects; rather, it folds binary strings into a matrix in order to give a simpler definition of a function over the binary strings. This is common for the use of 'matrices' in systems that use 'matrix' to mean a two dimensional storage array rather than the mathematical object that we use here.

All of the previous discussion in this paper has been building towards creating a system that uses mathematical objects for both the particles *and* links of our system. This is the beauty of a mathematical product: it is in some ways an object with properties in its own right.

Additionally, the matrices themselves are rich in emergent properties that might be exploited by our system.

Hermitian Matrices and Subsymbolic Artificial Chemistries

The atoms in the Jordan ssAChem used here are 3×3 Hermitian matrices.

Hermitian matrices use the Hermitian conjugate of a complex matrix:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}^\dagger = \begin{pmatrix} \bar{a_{11}} & \bar{a_{21}} & \bar{a_{31}} \\ \bar{a_{12}} & \bar{a_{22}} & \bar{a_{32}} \\ \bar{a_{13}} & \bar{a_{23}} & \bar{a_{33}} \end{pmatrix} \quad (14)$$

The elements a_{ij} are complex numbers, and $\bar{a}$ is the complex conjugate of a. A matrix M is Hermitian if $M = M^\dagger$. Hermitian matrices are closed under the Jordan product (McCrimmon, 2006).

Hermitian matrices provide a rich variety of properties such that we can use them as prime material for creating a subsymbolic AChem (ssAChem) where emergent properties of the matrices dictate the linking capabilities/probabilities of a particle, and the algebra gives the structure of the composite particles.

In this work we use the eigenstates of the Hermitian Matrices, chosen for their dimensionality and spatiality, and because they are a well studied mathematical object. A fully worked example of linking, probabilities and strengths generated using Hermitian matrices is given in the Appendix.

Subsymbolic Link

We make use of the eigenstate of the matrix to define linking probabilities. For a matrix M, consider

$$Mv = \mu v \quad (15)$$

The solution vectors v_i are the eigenvectors; the corresponding scalars μ_i are the eigenvalues. Here we choose these unit eigenvectors and the corresponding normalised eigenvalues λ_i as our emergent properties of interest to define our linking probabilities:

$$\lambda_i = \mu_i / \sum \mu_j \quad (16)$$

We normalise the eigenvalues to ensure sensible linking probabilities of larger composites.

The probability of two particles A and B linking, based on a given pair of eigenvalues and eigenvectors, is defined to be:

$$p_{A_i B_j} = \mathcal{N}(\lambda_{A_i} - \lambda_{B_j}) \left(1 - \tfrac{1}{2}((v_{A_i} \cdot v_{B_j}) + 1)\right) \quad (17)$$

This has two parts. The first term $\mathcal{N}(\lambda_{A_i} - \lambda_{B_j})$ is the probability density of the normal distribution ($\mu = 0, \sigma = 1$) at the point given by the difference in the normalised eigenvalues. This means the probability of linking is larger for more similar normalised eigenvalues. The normal distribution is not the only option; we simply need a symmetric distribution centred on zero, and the normal distribution is a well-known such one.

The second term $(1 - \tfrac{1}{2}((v_{A_i} \cdot v_{B_j}) + 1))$ uses the dot product between the corresponding unit eigenvectors. The dot product between two unit vectors is the cosine of the angle between them. The overall term has a value between 0 and 1, and is 0 if the vectors are perfectly aligned and 1 if the vectors are anti-aligned.

The probability of two particles linking, p_{AB}, is defined as the maximum probability of all the possible pairs:

$$p_{AB} = \max\{p_{A_i B_j} \ \forall i, j\} \quad (18)$$

The other property we define from the eigenstate is the strength of the link (probability that the link does not decompose). This is based solely on the difference in the normalised eigenvalues:

$$l_{A \bullet B} = \mathcal{N}(\lambda_{A_i} - \lambda_{B_j}) \quad (19)$$

Both these properties are based on the binary product. We define the linking probability of a triple link to be:

$$p_{ACB} = \min\{p_{AC}, p_{CB}\} \quad (20)$$

and we use the same set of eigenvalues to generate the links strength as the minimum of the strengths for each of the pairs $[\lambda_{A_p}, \lambda_{C_q}]$ and $[\lambda_{C_r}, \lambda_{B_s}]$:

$$l_{\{ACB\}} = \min\left\{\mathcal{N}(\lambda_{A_p} - \lambda_{C_q}), \mathcal{N}(\lambda_{C_r} - \lambda_{B_s})\right\} \quad (21)$$

This can be extended to the n-tad case in a similar fashion.

Composite Particles Probability

In our system the links have properties of their own. Equation 18 is the probability of the link forming given the presence of its components and that we choose that many reactants.

We need a further two probabilities to work out the probability of the resulting composite of the link existing, f_A. We need a probability of a particle existing, e_A:

$$e_A = \begin{cases} 1 & \text{if } A \text{ is an atom} \\ f_A & \text{if } A \text{ is a composite} \end{cases} \tag{22}$$

And we need a probability that each particle takes part in the reaction. All reactions that form a link require us to select at least two components. We define the probability of selecting further components for the reaction as 0.1 for each component. This choice discourages our system from constantly forming large links and quickly becoming one large composite.

$$r_{\{A_1, \cdots, A_n\}} = 0.1^{n-2} \tag{23}$$

So together the probability of a composite forming in terms of its last link is:

$$f_X = f_{\{A_1, \cdots, A_n\}} = p_X r_X \prod_{i=1}^{n} e_{A_i} \tag{24}$$

Composite Particles Strength

Link strength in our system is truly a property of the link rather than the composite. The composite particle contains a series of links all with different properties; each link has the link strength given in Equation 19. The overall composite strength is given by the probability of each of the links not decomposing, and is the product of all the link strengths in the system. The result is that larger links are stronger than a series of smaller links as there is less chance for the composite to break down. This is given by:

$$s_X = \prod_{Y \subset X} l_Y \tag{25}$$

where l_Y is the link forming Y a sub-particle of X.

Structure

The Jordan Algebra underlying this system means that the structure of the composite is as important as the particles that make it up. Through this we can see that not only does the structure add to the properties of the composite, but also we can find behaviour in the structure independent of the particles. By this we find the analogue of an *isomer* from real chemistry.
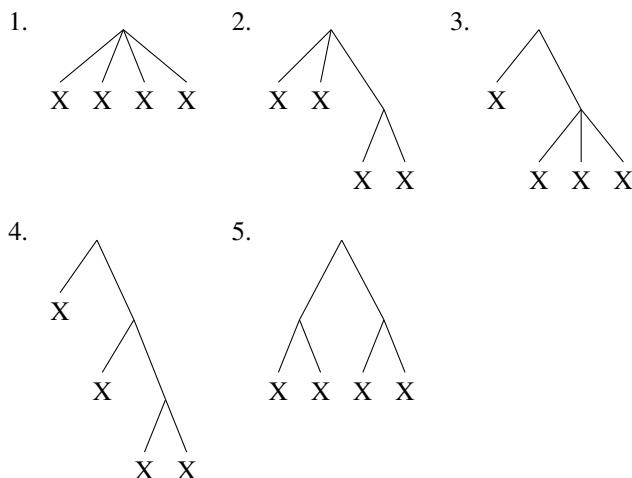

Figure 3: The set of structural isomers of four identical atoms

Isomer	Probability	Strength	Maximum reaction size	No. links
I1	0.0020	0.3989	4	1
I2	0.0040	0.1592	3	2
I3	0.0040	0.1592	3	2
I4	0.0079	0.0635	2	3
I5	0.0079	0.0635	2	3

Table 2: Particle probability and particle strength for the isomers of the identity atom (isomers 1 to 5 with X=I).

An Example of Emergent Richness

Throughout we have talked about the richness of the Jordan Algebra basis for this system and how it allows us to create a system in which we can have meaningful isomers. We can see this most clearly when we consider the homomers (isomers containing only one kind of particle) generated by the identity matrix, I, which is also an identity under the Jordan product (and hence all these composites are also represented by I).

There are five possible structural isomers using just four atoms I (Figure 3). Each of these has a probability of forming and a strength (Table 2).

From these results we can see that in the simplest of circumstances (when we are looking only at the structure of the isomer), structures with larger links are harder to form but once created are also harder to destroy. Thus we have emergently created a system that has stronger and weaker structures depending on the way the composite particle forms.

These multiple link isomers can form links with particles of different matrices, which have different eigenvalues. We give a second example of a base particle for which the five

Isomer	Probability	Strength	Maximum reaction size	No. links
M1	0.0019	0.3774	4	1
M2	0.0037	0.1476	3	2
M3	0.0037	0.1496	3	2
M4	0.0073	0.0585	2	3
M5	0.0070	0.0557	2	3

Table 3: Particle probability and particle strength for the isomers of the M atom (isomers 1 to 5 with X=M).

resultant homomers behave differently. Consider the matrix M:

$$M = \begin{pmatrix} 1 & i & 0 \\ -i & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (26)$$

The isomers formed from M all result in the same final matrix, as is true of all homomers due to the power associativity law (Equation 5). Since all the component matrices have the same eigenvectors, we never have a product occurring with the same eigenvalue position, as they are perfectly aligned and so the linking probability is zero (Equation 17). This increases the strength and probability of larger links as all pairs in the link can form across the strongest link because there is no case in which both possible link positions are occupied.

These results show that isomers are all unique: the structure is not defined by size or number of links (otherwise isomers M2, M3 or M4, M5 would be identical). This also shows that the structure has a strong effect on the system. The link properties are changing because we are no longer working with the identity matrix, and the composites are different from M. Thus when we link with the larger composite particles, eg $((M \bullet M) \bullet M)$, we are in fact linking with a different set of eigenvalues (if not necessarily a different set of eigenvectors as discussed previously). This still gives us a pattern of decreased probability of creation and increased strength for a smaller number of larger links.

We can also see that this behaviour does not indicate a universal pattern of higher probability causing lower strength regardless of other properties. Isomer M4 has a higher probability than isomer M5 but also has a higher strength. We can also see that these do not stretch across different homomers as I isomer I1 is more probable than M isomer M1 and is also stronger. This means that it is not the relationship between probability and strength that causes this behaviour, it is a relative effect caused by differently structured homomers of the same size.

While the probabilities given for the existence of composites existing here are small and tending towards zero for larger composites we must remember that this system is intended to operate over a very large number of interactions.

Thus while the probability of any particular composite existing is small particularly for larger composites the chance of generating a (large) composite is relatively high given the number of possible (large) composites.

Other Possible Behaviours

This is not the only interesting behaviour we might find which results directly from non-associativity and a mathematical focus.

Firstly we may consider looking at the isomers of our system in order to understand the more general behaviours of isomers. We can look at the probabilities of large molecules forming and their ability to act as information storage and transfer. We can be certain from the design of the system that the formation and replication of large composite particles is possible. However we cannot be sure of how stable or regular these large composites would be. If they are not stable then they cannot act as an information storage and transfer mechanism as the data would have too high a probability of corrupting. If they are not sufficiently regular then the information stored in them cannot be read in any useful manner.

The concepts of catalysis and substitution are fairly well established behaviors looked for in AChems (Hutton, 2003; Faulconbridge et al., 2010; Hickinbotham et al., 2010; Suzuki et al., 2003). Many other systems implement these concepts in addition to their basic reaction mechanism. We could create additional capability in our system to enable catalysis and substitution, but it is not necessary. This is because we have a probabilistic system. If we have two particles, A and B, that have a low probability of linking then we can use a composite particle C to generate a larger particle $((A \bullet C) \bullet B)$ (Figure 4). When we decompose in the correct manner this can leave us with a composite $(A \bullet B)$ whose total probability of forming is much higher than the original probability of A, B linking directly. It is even possible that this would allow objects with perfectly aligned eigenvectors, which would normally have a linking probability of 0, to connect and have a strong resultant link.

Another well-established desirable behaviour is replication. We have not eliminated self-replication. In this system it would look like a composite forming and then it being used much like C in Figure 4 to help the formation of an identical composite. Interesting instances of this would be cases in which the copy has a higher probability of existing than the original. This would mean that the composite encourages the creation of copies of itself.

Considering the analogy to real chemistry, we will at some point want to consider adding a temperature analogue to our system. This should modify the probability of linking in the system. In this case it could modify either the eigenvalues or the eigenvectors of the matrix. There are very few ways to do this that do not effect both of these, so as well as changing the probability of linking we would be caus-
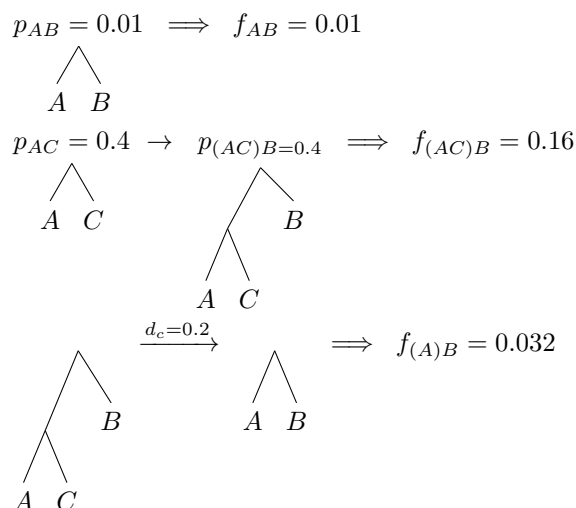
$p_{AB} = 0.01 \implies f_{AB} = 0.01$

$p_{AC} = 0.4 \rightarrow p_{(AC)B=0.4} \implies f_{(AC)B} = 0.16$

$\xrightarrow{d_c=0.2} \implies f_{(A)B} = 0.032$

Figure 4: A general example of what the concept of catalysis would look like in our system where d_C is the probability of C decomposing in the correct manner.

ing the composite to "rotate". The structure of the matrices provide for a direct connection between an analogue of thermal energy and its effects on the energetic states (such as rotational or kinetic energy) of our composite particles. In addition, in our system variation of temperature should lead to increasing and decreasing probabilities, similar to the effects in ensembles in real chemistry as described by statistical thermodynamics.

Summary and Conclusions

Creating an AChem using a mathematical basis such that links and ordering exists gives us a chance to exploit the open set of Hermitian matrices and their already well-studied emergent properties. It also provides us with structure that exists and is capable of displaying emergent behaviours that cannot be predicted purely from the mathematical roots of the system.

We have shown that even within the power associativity provided by the Jordan Algebra the fact that the underlying system is non-associative allows for these strong structural influences to become more prevalent and vary the system. We have also discussed that this may not remain the case when expanded to general isomers but will remain true in homomers.

We have demonstrated that we can find interesting important behaviors in this sort of ssAChem that were in no way designed into our system. This suggests that these systems have much greater potential in that they are not being limited by the intentions and goals of their creators. In terms of our starting metaphor, we are starting to create bricks and mortar, rather than a prefabricated AChem.

Appendix: worked example
Linking Hermitian Matrices

Consider the three Hermitian matrices:

$$X = \begin{pmatrix} 1 & -i & i \\ i & 0 & 0 \\ -i & 0 & 0 \end{pmatrix} \quad Y = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$Z = \begin{pmatrix} -1 & i & 0 \\ -i & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{27}$$

These have the following eigenvector matrices and eigenvalues:

$$X : v_X = \begin{pmatrix} 0.58i & 0 & 0.82 \\ 0.58 & 0.71 & -0.41 \\ -0.58 & 0.71 & 0.41 \end{pmatrix} \lambda_X = \begin{pmatrix} -1 & 0 & 2 \end{pmatrix} \tag{28}$$

$$Y : v_Y = \begin{pmatrix} 0.41 & 0.71 & 0.58 \\ 0.41 & -0.71 & 0.58 \\ -0.82 & 0 & 0.58 \end{pmatrix} \lambda_Y = \begin{pmatrix} 0 & 0 & 3 \end{pmatrix} \tag{29}$$

$$Z : v_Z = \begin{pmatrix} -0.71i & -0.71i & 0 \\ 0.71 & 0.71 & 0 \\ 0 & 0 & 1 \end{pmatrix} \lambda_Z = \begin{pmatrix} -2 & 0 & 0 \end{pmatrix} \tag{30}$$

We can form three links over these matrices, using the relevant products:

1. $(X \bullet Y) = \begin{pmatrix} 1 & \frac{1}{2} - \frac{1}{2}i & \frac{1}{2} + \frac{1}{2}i \\ \frac{1}{2} + \frac{1}{2}i & 0 & 1i \\ \frac{1}{2} - \frac{1}{2}i & -1i & 0 \end{pmatrix}$

2. $\{X, Y, Z\} = \begin{pmatrix} -1 & 1i & -\frac{1}{2} - \frac{1}{2}i \\ -1i & -1 & \frac{1}{2} - \frac{1}{2}i \\ -\frac{1}{2} + \frac{1}{2}i & \frac{1}{2} + \frac{1}{2}i & 0 \end{pmatrix}$

3. $((X \bullet Y) \bullet Z) = \begin{pmatrix} -\frac{3}{2} & -\frac{1}{2} + 1i & -\frac{3}{4} - \frac{1}{4}i \\ -\frac{1}{2} - 1i & -\frac{1}{2} & \frac{1}{4} - \frac{3}{4}i \\ -\frac{3}{4} + \frac{1}{4}i & \frac{1}{4} + \frac{3}{4}i & 0 \end{pmatrix}$

Link Properties

For each of these links we can calculate the probability of the link, L, forming, p_L, and the strength of the link, l_L. In order to calculate the probability of L we need to calculate the probability using each possible choice of eigenvalues. Taking link 1 we have the probabilities as given in Table 4 the maximum of which occurs using the second eigenvalues

p	X_1	X_2	X_3	Z_1	Z_2	Z_3
Y_1	0.035	0.257	0.041	0.086	0.142	0.362
Y_2	0.170	0.299	0.019	0.182	0.299	0.200
Y_3	0.027	0.242	0.121	0.118	0.072	0.051

Table 4: Probability of linking with Y for X and Z for each eigenvalue

p	Z_1	Z_2	Z_3
$(X \bullet Y)_1$	0.0270	0.1210	0.0354
$(X \bullet Y)_2$	0.0782	0.2700	0.0997
$(X \bullet Y)_3$	0.1638	0.0175	0.0135

Table 5: Probability of $(X \bullet Y)$ linking with Z for each eigenvalue

of X and Y, this gives us a probability $p_{XY} = 0.2992$ and a strength of $l_{XY} = 0.3989$. These along with the strength and probability of the other two links are summarized in Table 6.

For link 2 we take the minimum of the probabilities p_{XY} and p_{YZ}. The probabilities for p_{YZ} are also given in Table 4 and the maximum occurs with Y's first eigenvalue and Z's third eigenvalue. This gives $p_{YZ} = 0.3623$ which has a strength of $l_{YZ} = 0.3989$. We then have that the overall $p_{XYZ} = 0.2992$ and it has strength $l_{XYZ} = 0.3989$.

For link 3 we need to work out a third set of probabilities between $(X \bullet Y)$ and Z, these will be based on the eigenstate of $X \bullet Y$ (Table 5):

$$(X \bullet Y) : V_{(X \bullet Y)} = \begin{pmatrix} 0 & -0.5 - 0.5i & 0.5 + 0.5i \\ -0.71i & 0.5i & 0.5i \\ 0.71 & 0.5 & 0.5 \end{pmatrix}$$

$$\lambda_{(X \bullet Y)} = \begin{pmatrix} -1 & 0 & 2 \end{pmatrix} \qquad (31)$$

This means the largest probability comes from using the second eigenvalues of each component and so $p_{(XY)Z} = 0.2700$ which makes the strength $l_{(XY)Z} = 0.3989$.

Composite Properties

We next consider the f and s values for the composite resulting from each link. The first link has only atoms and is binary so $f_{XY} = p_{XY}$ and since there is only one link the strengths are also the same: $s_{XY} = l_{XY}$. Similarly $s_{XYZ} = l_{XYZ}$ as there is only one link but $f_{XYZ} \neq p_{XYZ}$ as the size of the link reduces the probability to $f_{XYZ} = 0.1 p_{XYZ} = 0.0299$.

Finally we have the probability of the third link: $f_{(XY)Z} = p_{XY}p_{(XY)Z} = 0.0808$. The base components are atoms and the links are all binary so it is simply the product of the probabilities of both links. The strength is the product of the strengths of each link: $s_{(XY)Z} = l_{XY}l_{(XY)Z} = 0.1591$.

Link	Probability	Strength	Composite Probability	Composite Strength
1	0.2992	0.3989	0.2992	0.3989
2	0.2992	0.3989	0.0299	0.3989
3	0.2700	0.3989	0.0808	0.1591

Table 6: Summary of links probabilities and strengths

References

Banzhaf, W. (1993). Self-replicating sequences of binary numbers. Foundations II: Strings of length $N = 4$. *Biol. Cybern.*, 69(4):275–281.

Dittrich, P., Ziegler, J., and Banzhaf, W. (2001). Artificial Chemistries—A review. *Artificial Life*, 7(3):225–275.

Faulconbridge, A. (2011). *RBN-World: Sub-Symbolic Artificial Chemistry for Artificial Life*. PhD thesis, University of York.

Faulconbridge, A., Stepney, S., Miller, J. F., and Caves, L. S. D. (2010). RBN-World: The hunt for a rich AChem. In *ALife XII*, pages 261–268. MIT Press.

Gonzalez, S. and Martinez, C. (2003). Nonassociative algebras: Some applications. *Rev. Mat. Iberoamericana*, 19(2):385–392.

Hickinbotham, S., Faulconbridge, A., and Nellis, A. (2010). The blind watchmaker's workshop: three artificial chemistries in the context of Eigen's paradox. In *ALife XII*, pages 82–89. MIT Press.

Hutton, T. J. (2002). Evolvable self-replicating molecules in an artificial chemistry. *Artif. Life*, 8(4):341–356.

Hutton, T. J. (2003). Information-replicating molecules with programmable enzymes. In *Proc. Sixth International Conference on Humans and Computers*, pages 170–175.

Lucht, M. W. (2012). Size selection and adaptive evolution in an artificial chemistry. *Artif. Life*, 18(2):143–163.

McCrimmon, K. (1978). Jordan algebras and their applications. *Bull. Am. Math. Soc.*, 84(4):612–627.

McCrimmon, K. (2006). *A taste of Jordan algebras*. Springer.

Muller, P. (1994). Glossary of terms used in physical organic chemistry (IUPAC recommendations 1994). *J. Macromol. Sci. Part A Pure Appl. Chem.*, 66(5).

Reed, M. (1997). Algebraic structure of genetic inheritance. *Bull. Am. Math. Soc.*, 34(2):107–130.

Suzuki, H., Ono, N., and Yuta, K. (2003). Several necessary conditions for the evolution of complex forms of life in an artificial environment. *Artif. Life*, 9(2):153–174.

Generalized Stochastic simulation algorithm for Artificial Chemistry

Hedi A. Soula[1,2]

[1] INRIA EPI Beagle LYON, FRANCE
[2] INSERM INSA U1060 LYON, FRANCE
hedi.soula@inria.fr

Abstract

Artificial chemistries (AC) are useful tools and a simple shortcut for the study of artificial life. In many works, AC's are quite straightforward or simplistic or highly unrealistic (or all combined) but in several works AC are extremely complex. Among them, we focus of Hutton Artificial Chemistry HuAC where reactions act on the nodes of a graph (so-called the atoms) where the connected components composed the actual molecules of the environment. The main works from Hutton are based on a 2D simulator (squirm) with auto-replication and several other properties. This paper proposes a computation framework and software that cancel the need for 2d space simulation in the HuAC while keeping a lot of the features of this chemistry. It relies on the Stochastic Simulation Algorithm that has been here adapted to work on graph structure. In order to test it, we simulated Hutton's auto-replication – which relies heavily on strong spatial interactions – in a spaceless environment. In addition, due to the increase in performance, we develop some preliminary work on Random Chemical Worlds where reactions are randomly selected. We showed on simple metrics that the fraction of reactions among all possible is a general parameter that acts on the system similarly to a phase transition.

Introduction

Most of the artificial life questions and problematic revolve around understanding and providing clues to the origin of life and evolution of organism starting from scratch (Hutton (2003)). Because of the difficulties of real-life (or 'wet') experiments to address these questions in 'real' life, simulations and artificial modelling seems to be a most adequate tools (Dorin and Korb (2007)). Unfortunately, reproducing completely life-like systems are still not an option for understanding general features used by living system to adapt and develop. Few artificial systems can be designed without the need to fix rules between building blocks components of living organisms. It usually necessitates to design an artificial chemistry scheme (Dittrich et al. (2001)) and this need extends obviously to problems on artificial life and evolutionary strategies. Indeed put it simply, since real life rests on chemistry *artificial* life should rely on artificial chemistry (Suzuki and Dittrich (2009)).

What is done (usually) is that first most chemistry is prescribed: it has small dimension (small # of reactions), straightforward that is the chemistry graph is either extremely simplistic or small (Knibbe and Parsons (2014)) and it is somewhat unrealistic for example most transition energies are ignored. Several AC have been recently developed to tackle specifically this issue of energy transition (Benkö et al. (2005); Ducharme et al. (2012); Benkö et al. (2009)). Additionally, problems like mass conservation can arise e.g. $A + B \mapsto C$ and $C \mapsto A$. It is expected that any evolvable scheme will exploit this kind of easy shortcut.

So what properties an AC should possess for more general life-like and evolution-based framework experiments? AC must be complex, rich and generic. More precisely, AC should be large and have a huge number of reactions. Second, and this is related to energy transitions, all reactions should not be possible. This amounts to require that components (molecules) cannot be reactive with all others. Obviously, mass conservation is required. This problem usually arises when the chemistry of the system is procedurally generated – for example using artificial genome (Rocabert et al. (2015)). In other words, reversibility should not be hacked. Finally, AC should allow some form of open endedness: we don't know all reactions/molecules (Lenaerts and Bersini (2009)).

Several frameworks have been published that address some of the properties described above Tominaga et al. (2009); Oohashi et al. (2009)).

One of them in particular, Hutton's Artificial Chemistry (HuAC) own several of these properties (Hutton (2007, 2004, 2002)). The central feature (cf Model) is to describe the chemistry as reactions between atoms with a fixed type and changing state while describing molecules as connected graphs of atom. Note that the term atoms refers to the smallest structure in the system and can describe structures – or domains – that are larger than actual atoms. The chemistry is a set of reaction on pairs of atoms and strikingly *not* on molecules. Like in the classical sense, these atoms need to make an encounter to react with each others (as a classical bimolecular reaction). However, the originality of HuAC

relies on so-called conformation reactions: reaction that occurs between bounded atoms. HuAC possesses all the required elements to be of a wider use. It has some element of open-endedness but part of the chemistry reaction set has been finely tuned to obtain the expected outcome.

Understandably, hand designing was part of the proof of concept of the chemistry (mainly to display self-replication) but lacks of generalisations. The simple questions what are the 'best' set of chemistry rules to obtain a life-like chemical system where for example artificial evolution can be tested is still open. Also overall, the main drawback is that it is in essence a 2D construction with - without any explicit mention to it - a strong diffusion-limited component.

This is a drawback for two reasons: 2D systems are nice but long to simulate – most of the time time between reactions is juste to simulate diffusion. Also, 2D simulations are simple to simulate but introduce unnecessary topology constraints like chirality that cannot be overcome without strong tuning. Also, as we will see more in details afterward, all the tuning relied on strong spatial assumptions: correlation of positions ignoring mixing effects. Finally, and to for sake of completeness, HuAC does not introduce really reaction rates in a biochemical sense: bimolecular reactions occurred immediately upon collision - and conformation reaction occurred instantly. Thus, there were absolutely no notion of reactions rates (and affinity etc...).

The aim of this article is to provide a framework – a stochastic simulator that simulate Hutton's artificial chemistry with several properties. First it will describe this chemistry in a well stirred medium (which amounts to 'infinite diffusion') using stochastic methods known as the Gillespie algorithm. This algorithm will be modified to suit Hutton's artificial chemistry that keep track of graphs of atoms and not atom abundance only. Our framework also introduce reactions rate and other reactions that were not included in the descriptions of Hutton's original work. Also we will examine the difference with HuAC's dependence on 2D structural constraints and specially the impact of spatial correlation for the main Hutton's algorithm: auto-replication. Finally since this framework can handle a huge number of reactions and/or molecules we will present preliminary works on randomly generated Artificial Chemistry and study their properties on simple metrics.

Model: Hutton's Artificial Chemistry

We start this section with a brief refresher on Hutton's chemistry (HuAC). HAC has been published in several papers (Hutton (2009, 2007, 2004, 2002)) but to our knowledge only one follow-up (Lucht (2012)). The main advantage of HuAC is that the description is really straightforward. The chemistry's rules are very simple. However, it can grow extremely complex due its graph structure. Briefly, Hutton's chemistry is composed of molecules that are graphs of *atom* (in Hutton's words). The atom term do not encompass actual

atomic structure but merely describe the smallest compound in the system. As described, atoms have a type and a state that are both integers and molecule are simply connected graphs whose nodes are atom and with connections (bonds). Atoms have a fixed type and a changing state that are both integers and described by a pair $(t|s)$ with $t, s \in \mathbb{N}$ (note: in Hutton's papers type is a letter and state is an integer e.g $a0, e8\ldots$). Any atom can have any numbers of connections. Therefore the chemistry is composed of fully connected subgraphs which is the set of molecules. The chemistry relies on a physical simulator in 2D. All atoms are spatially resolved and each atom has its own id and position. Atoms are hard spheres (of equal size) that undergo some kind of brownian motion in viscous environment and links are coded using springs $-k(p - q)$.

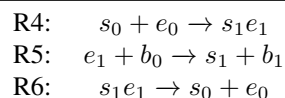Reactions are based only on atoms and are of the form:

$$(t_1|s_1)(.|+)(t_2|s_2) \rightarrow (t_1|s_3)(.|+)(t_2|s_4) \tag{1}$$
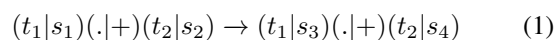
where . design a link i.e one atom of $(t_1|s_1)$ and one of $(t_2|s_2)$ must be linked for the reaction to occur as they design a conformational change in the pair. Also $+$ design an encounter reaction (therefore $(t_1|s_1)$ and $(t_2|s_2)$ *must* not be linked together and the reaction occurs whenever the two atoms collide (whether they are otherwise linked to other atoms). Note that one way to ensure mass conservation is to never allow the type to be changed in the equations. All reactions occur locally so other links are not modified (they can be later if there is a reaction matching the new links in the graph). In the original papers, conformationnal reaction noted . is performed instantaneously and when several are possible they "are chosen at random" – presumably with uniform distribution.

This chemistry is extremely general and encompass very easily several classical equations such an enzyme-substrate-product $S + E \rightleftharpoons C \rightarrow P + E$ using

R1:	$s_0 + e_0 \rightarrow s_1 e_1$
R2:	$s_1 e_1 \rightarrow s_2 + e_0$
R3:	$s_1 e_1 \rightarrow s_0 + e_0$
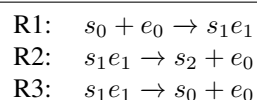
and also the so-called *Hit & Run* reactions :
$S + E \rightleftharpoons C$ with $C + A \rightarrow C + B$:

R4:	$s_0 + e_0 \rightarrow s_1 e_1$
R5:	$e_1 + b_0 \rightarrow s_1 + b_1$
R6:	$s_1 e_1 \rightarrow s_0 + e_0$

note that in the last equation the complex the atom e_1 is the functional equivalent of the complex C. Technically any e_1 in the reactor can react with a b_0 to yield a b_1.

However, due to its atom/graph organisation, this chemistry can quickly give rise to complex structure. For example the simple reactions:

R7:	$a0 + a1 \mapsto a1.a2$
R8:	$a0 + a1 \mapsto a2.a1$
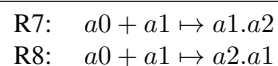
(where the only difference is a swap between target states) for a given number of $a0$'s and $a1$'s in the reactor yields two different graph structures as shown on Fig. 1.
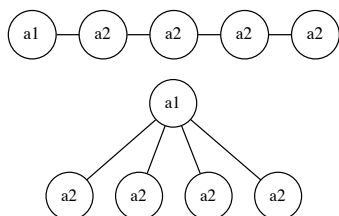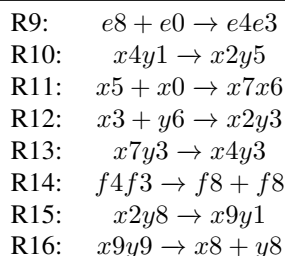

Figure 1: Examples of graph structure obtained using only R7 (top) and R8 (bottom) starting with 4 $a0$ and one (1) $a1$.

Hutton's original papers also come with an auto replication scheme where a given arbitrary single strand molecule $e_8.x_1^1 \ldots x_1^k.f_1$ can automatically be duplicated using a given set of reactions:

R9:	$e8 + e0 \rightarrow e4e3$
R10:	$x4y1 \rightarrow x2y5$
R11:	$x5 + x0 \rightarrow x7x6$
R12:	$x3 + y6 \rightarrow x2y3$
R13:	$x7y3 \rightarrow x4y3$
R14:	$f4f3 \rightarrow f8 + f8$
R15:	$x2y8 \rightarrow x9y1$
R16:	$x9y9 \rightarrow x8 + y8$

Here is introduced the wild-card reaction using x and y. For example, $x4y1$ refers to linked atoms of *any* type but in state 4 and 1 whereas $x4 + x0$ refers to the collision of two atoms of the same type (but of any type) in state 4 and 0. One can immediately see that this does lead to a replication of any sequence $e_8.x_1^1 \ldots x_1^k.f_1$. Elements are added and linked using reaction R11 and the splicing is initiated by R14 and propagated via R15/R16 (see 2 for a walkthrough).

This AC has several wanted features for artificial chemistry. It is very general and allows for very complex feature emergence (as attested by Fig. 1). Namely, molecules can be very complex. Also this comes with default embedded mass conservation. Due to a complicated graph geometry with 2D features, there are some chirality issues (due to 2D) and of course this AC is computationally demanding. Also as mentioned, from a chemical standpoint, there is no reaction rates.

The final remark is that most results that have been published on this AC has used hand-crafted and well designed set of reactions. In particular, there was no clear thought process described that explained how the replication system could work. It seems to us that it was drawn by hands and graph evolution was constructed with implicit bias due to proximity. However cells and more generally biological medium are disordered and can be highly diffusive. And of course processes takes places, mostly, in three dimensions.
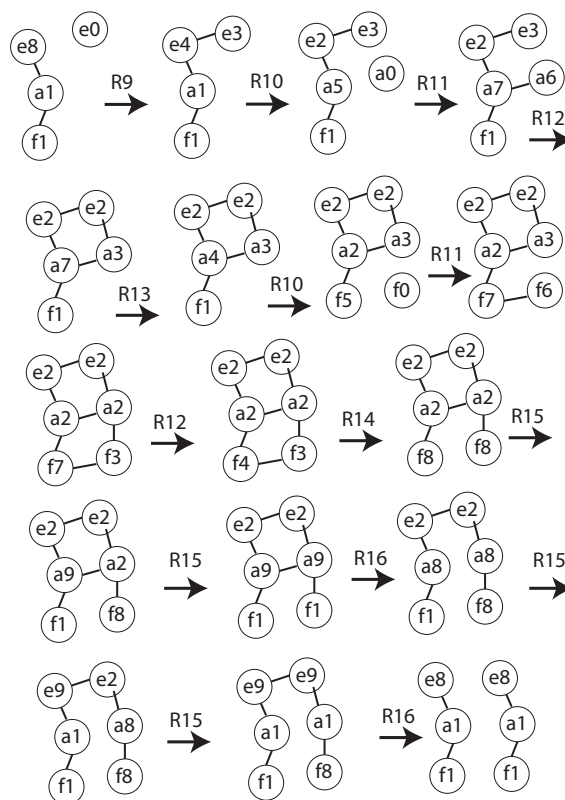

Figure 2: Walkthrough of the replication R9-R16 with an initial seed of $e8a1f1$ reproduced from Hutton (2002)

The very complexity of the HuAC lead us to design a system that would be more tractable and keep the essence of the original chemistry. We are also able to experiments with a large number of particles and compounds and deal with a large number of possible reactions to obtain more reliable results on the potential of HuAC.

STAARC: STochastic Atom-based ARtificial Chemistry

We developed the STAARC framework to use the HuAC in a virtual reactor that completely eliminate spatial location. Since due to the particle-based nature of HuAC and its graph structure, reactions are identified and occur concurrently but asynchronously at a given speed. Therefore, the ODE formalism seems not realistically tractable for this problem. This framework is not based on differential equations but on the Stochastic Simulation Algorithm (SSA) as it was originality developed by Gillespie Gillespie et al. (2013); Gillespie (2007).

Stochastic Simulation Algorithm

The important feature of the Gillespie algorithm is that we simulate the world only at times when a reaction occurs and not in between. Contrary to ODE formalism where description can occur for arbitrarily small time step, in SSA, all that is needed is to estimate which will be the next reaction, when it will occur and to actually implement the reaction. Since both the time and the reaction will be drawn according to a given diffusion, it is a simple mean to obtain noise and variability on molecular reactions. Note that, however, both the ODE and its SSA counterpart describe the same system with the same hypotheses and therefore yield the same results: the ODE being the description of the dynamics of the average.

To function, Gillespie algorithm introduces propensities a_i: the average rate at which the reaction i can occur in the medium. For a bimolecular reaction $X + Y \xrightarrow{k}$ this rate is equal to kxy when x and y are the *number* of molecule X and Y respectively in the reactor. Similarly, unimolecular reaction $X \xrightarrow{k}$ propensity is kX. Note that if $X = Y$ i.e bimolecular reaction involving the same type of molecule the propensity is equal to $kx(x - 1)$. Propensities are the speed for one reaction to occur and are therefore of unit $time^{-1}$. The algorithm is as follow: When all reactions propensities are computed we compute the propensity of the system :

$$a = \sum_{i \in Reactions} a_i$$

and Gillespie showed that the time for the next reaction to occur is exponentially distributed with this parameter a.

Since reactions occur proportional to their rate, a simple random selection biased by relative rate proportion of rate yields the next reaction. Once the reaction is selected, it is applied i.e the reactants are removed and the products are added. This in turn modifies propensities - that need to be reevaluated and so forth. Note that in this algorithm, propensities are only calculated when reactions are occurred, propensities depends only on the number of the reactants involved in actual reactions. Only one set of reactants is updated at each step of the algorithm and finally it involves only the drawing of two random numbers: One to find the next reaction time and another to find the reaction itself.

This description of Gillespie can be applied as this to the HuAC scheme with few tweaks. First we simply add to each reaction a rate k_r for each reaction r.

Now the main hurdle is to obviously have a data structure that contains all the information about the atoms and molecules (which are the connected components). Therefore the first – and important – difference with the SSA scheme is that we need to keep track of each individual particles $p_i = (t_i, s_i)$ for $i \in [1, N]$ (N being the number of particles) and we keep track of all the edges (link) between particles $\mathcal{E} = \{(p_i, p_j) where p_i linked p_j\}$. Note that in the SSA,

we normally keep track of the *number* only of each reactive particles.

For a given pair $(t|s)$ let's set $\mathcal{A}(t, s) = \{p_i | t_i = t, s_i = s\}$ the set of particles of this given type and state. Let's note $a(t, s) = |\mathcal{A}(t, s)|$. Also let $\mathcal{B}(t_1, s_1, t_2, s_2) = \{(p_i, q_j) | p_i \in \mathcal{A}(t_1, s_1), p_j \in \mathcal{A}(t_2, s_2), (p_i, p_j) \in \mathcal{E}\}$ and finally let $b(t_1, s_1, t_2, s_2) = |\mathcal{B}(t_1, s_1, t_2, s_2)|$.

For the bimolecular reactions of the type:

> Ra: $(t_1|s_1) + (t_2|s_2) \xrightarrow{k_a} \ldots$
> Ra': $(t_1|s_1) + (t_1|s_1) \xrightarrow{k_{a'}} \ldots$
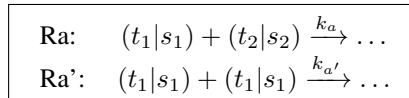
the rates will be equal to

Ra: $k_a \left(a(t_1, s_1) a(t_2, s_2) - b(t_1, s_1, t_2, s_2) \right)$
Ra': $k_{a'} \left(a(t_1, s_1)(a(t_1, s_1) - 1) - b(t_1, s_1, t_1, s_1) \right)$

We basically need to remove atoms that are already link for the equation since it concerns only unlinked atoms. We also need to take care of the case when atoms have the same type and state.

Also for a reaction of the type:
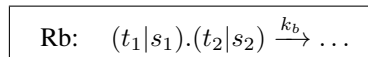
> Rb: $(t_1|s_1).(t_2|s_2) \xrightarrow{k_b} \ldots$

the rate will be equal to

Rb: $kb(t_1, s_1, t_2, s_2)$

We compute the total propensity as the sum of all the rates of all reactions using the formulas above and draw our two random numbers. The first one $r \in [0, 1]$ uniformly to compute the next time $\tau = -\log(r)/a$ and the other s for determining the reaction i such that:

$$\sum_{k \in reaction, k \leq i} a_i \leq s < \sum_{k \in reaction, k \leq i+1}$$

Once a reaction is selected we apply the modification to a given pair (selected at random uniformly). Due to the SSA scheme only one reaction is applied between each step so only one update to a maximum.

Properties

Akin to a more realistic chemistry, all reactions now have rate. This simulates a well mixed 3D reactor without any chirality issue that was only related to 2D. Almost all the HuAC properties are conserved in particular complex graphs. We can mimic diffusion to a certain point by modifying the actual rate of bimolecular reactions. Indeed, since Gillespie is the limit at infinite diffusion but reaction rate can be modified by diffusion using Smoluchowski equation (Szabo (1989)):

$$k = \kappa \frac{D}{A + D} \qquad (2)$$

κ being the thermodynamic rate (when $D \mapsto \infty$).

We obtain an extremely fast computation where the only simulated moments are whenever a reaction occurs. We also

update the data structure that keeps track of the graph for only one pair at a time. The addition allows to simulate also a wider variety of reactions. Indeed, several reactions can be added to the simulator to allow even more realism. Equation such as production $\varnothing \xrightarrow{k_p} (t|s)$, degradation $(t|s) \xrightarrow{k_d} \varnothing$ and 'auto-conformation' $(t|s_1) \xrightarrow{k_c} (t|s_2)$ with respective rates as k_p, $k_d a(t,s)$ and $k_c a(t,s_1)$ respectively. The simulator was written in C# and is available with MIT licence at `https://github.com/hsoula/staarc`.

Results

We provide in this section two examples of the output of the simulator: we reproduce the auto-replication scheme of Hutton and we create Random Chemical Worlds to study their properties. Because of its flexibility it allows us to test several possibilities in the set of equations rules and rates and also compare with the 2D case of HuAC. Due to the local nature of the replication process, we expected it to fail completely in a infinite diffusion medium.
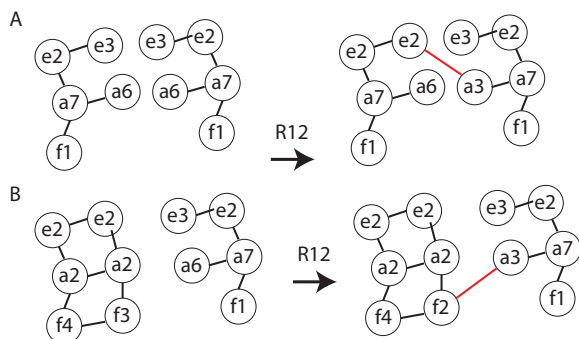

Figure 3: Examples of replication errors A) Due to no local constraints and concurrent replication bimolecular reactions can occur from other replication process B) Due to race condition – conformation change – occurring too late to prevent another bimolecular reaction.

Vanilla Replicator

We submitted our simulator the restricted set of equations (R9-R16) described above to check how replication occurs. When drawn by hands the replication seems it can occur flawlessly as shown on Fig. 2 but inspection of equation R13 shows a race condition. For the replication to continue atoms $a6$ must encounter (any) atome $x3$ (here $e3$). In a 2D and diffusion limited environment, the closest possible atom would be the $e3$ but it could theoretically be any atom from a concurrent replication elsewhere (see Fig.3A). This feature happened in the original Hutton's simulation occasionally and as this is the case here does not impact the stability of the replication it only changes the sequence in the replicated molecules. This deviation in replication is expected because
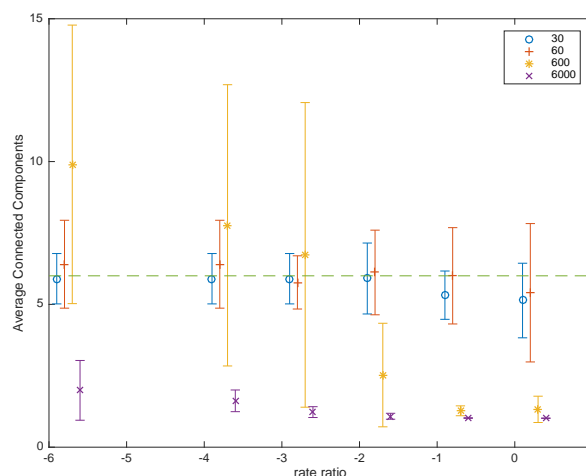

Figure 4: Number of connected components – molecules – at the end of the simulation of the replicator R9-R16. Two parameters were varied; first the rate of conformation versus bimolecular reaction (rate ratio) to simulate the impact of diffusion. The other parameter is the number of particles. The simulator was tested with an initial replicator seed: $e8a1b1c1d1f1$ of length 6 in addition with a number $x0$ with $x \in \{a,b,c,d,e,f\}$ varying from 60 to 6000 (from 5 to 1000 for each type). Results are displayed as mean $\pm$ standard-deviation (computed on 20 runs). For a perfect replication, molecules size average should be 6 – plotted as a dash line to guide the eye. A small jitter has been added on x values for clarity.

local interaction and spatial correlation can strongly modify bimolecular reactions either transiently (Van et al. (2014)) or at equilibrium (Caré and Soula (2011, 2013)) by modifying encounter probabilities.

In addition, there is another race condition on reaction R13. If R3 does not occur fast enough the $a3$ molecule can become linked with another $x6$ atom and turned into an $a2$ blocking the replication (see Fig.3B). This second race condition occurs because in the original scheme conformation reaction where instantaneous which is not the case anymore.

As we mentioned, this replication scheme is highly local and space dependent and can probably fail when confronted to well stirred and infinite crowding medium. In order to test the resilience of the replication, we created a reactor with an initial molecule $e8a1b1c1d1f1$ of size 6 and provided the medium initially with N of each type at state 0. This should start the replication process. All conformation reactions had given rate κ and bimolecular reaction of rate $\lambda\kappa$ (with $\lambda < 1$ the rate ratio to take into account diffusion).

We computed the average size of the molecules at the end of the process: when no reactions can occur i.e when $a = 0$. The results are displayed on Fig.4 with mean $\pm$ standard
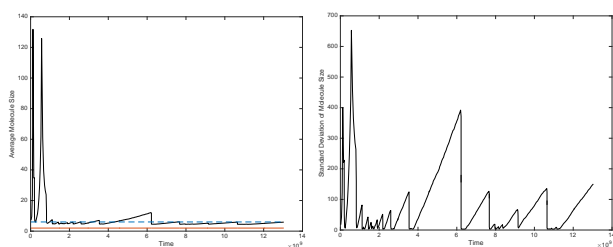
Figure 5: Results of long replication. Left: Average molecules size throughout time. Right: Standard-deviation of molecules size.

deviation (on 20 runs) for various N (so $6N$ total atoms) and various λ.

Normally when replication is occurring correctly the average size of the molecules at the end of the process should be 6 (dashed line displayed to guide the eye). For low values of N everything works correctly - except if λ is close to 1 and race conditions occur frequently stopping the replication earlier. This race condition 'error' occurs more and more frequently with increasing N because bimolecular reactions occurs more frequently with increasing concentrations. When λ is very small however most errors come from intertwined replication creating either big molecules ($N = 100$ and $\lambda = 10^{-6}$) or the minimal replicator $e8f1$ of size 2 ($N = 100$ and $\lambda = 10^{-6}$). Minimal replicators are the only stable replication seed that will ignore other replication interferences. In Hutton's original papers, he performed environmental 'wash' by setting all atoms to a zero state and unlinks them in a quadrant of the environment. He showed that the minimal replicator was 'selected' by this wash. We show here that it is enough to have a high population and ideal mixing to achieve the same result.

Long Replicator

We tested a longer episode of replications with initially 10 replicator seed $e8a1b1c1d1f1$ (of length 6) and no other particules. We've added the reactions of production/degradation for particles $x0$:

R17:	$(x\|0) \rightarrow \varnothing$
R18:	$\varnothing \rightarrow (x\|0)$
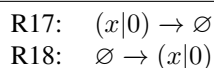
and let the system go for 450,000 reactions to occur (around $1.2 10^9$ time steps). Here conformational rate were equal to 1 and bimolecular reaction's rate was 10^{-4}. Due to both degradation and production (note that once bound, an atom does not degrade anymore), the system slowly feeds atoms to the various replications seed. Since λ is low, most 'errors' are from entwined replications that create bigger and bigger molecules as seen on Fig.5. The average size increases at the start of the experiment and slowly settled to the actual size of the initial replication seeds: 6 (a dashed line is displayed

to guide the eye; another line is for the size 2 of the minimal replicator). Since the number of particle grows through time averages are deceiving. Also on Fig.5, the standard deviation of the molecules size is displayed and showed that, at times, some molecules becomes extremely big compounds. As long as the replication process goes on they decreases to smaller size via separation. This alternation is particularly illustrated on Fig.6 which shows the number of replication according to time. High variation corresponds to big compounds slowly building – few replications per time unit – followed by quick disintegrations.

These results suggest that, when fed with atoms in a steady state manner, the replication scheme is fairly stable and in the end produces molecules whose sizes are the same as the initial replication seed.
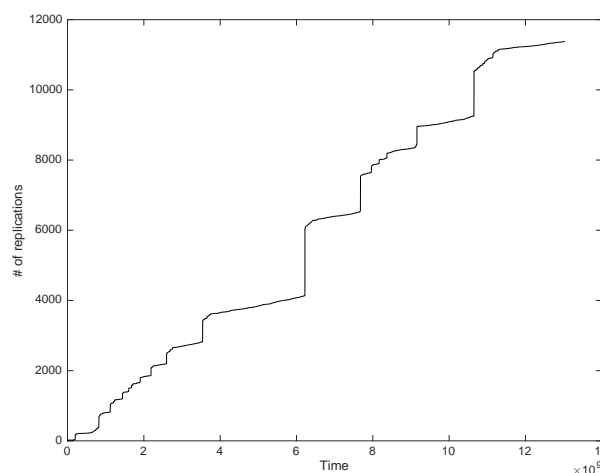

Figure 6: Number of replication through time (MC time) for the long replication events. Replication are obtained when rule R16 happened with an atom of type 'e'. Alternation of periods between extreme and slow replications are clearly visible.

Random Chemical World

The simulator allows us to compute several thousands of atoms and thousands of reactions. It is therefore possible to test use-cases where the reactions are drawn and chosen randomly and vary the amount of reactions available. Let the set of available set be $\{a, b, c\}$ and the maximum state being 5. We can compute all the possible rules in reactions, conformations, with no production nor degradation. We start with N particles $(t|s)$ with $t \in \{a, b, c\}$ and $0 \leq s \leq 4$ and $p \in [0, 1]$ describe the fraction of the reactions kept.

We simulate for a maximum of 2,000 reactions. We chose to have a rate of $1 time^{-1}$ for conformation reactions and $1^{-2} time^{-1}$ r collision reactions. We simulated 20 different AC's for increasing fraction of p simulating the first 2,000 reactions for $N = 10,000$ initial atoms. In addition, we

provided larger molecule by adding random links to atoms (with increasing probability). We used a very simple metric by computing the ratio of the number of molecules between the start and the end of the experiment. These results are displayed on Fig. 7. When the number of chemical reactions available is low, the AC does not modify the structure of the particles graphs - even when it is already structured (orange star). Whereas a big number of reactions yields standardised particles graph with very low variability among chemistries. The interesting behaviour occurs at the transition where variability is at the highest between 10^{-4} and 10^{-3} fraction of possible reactions.
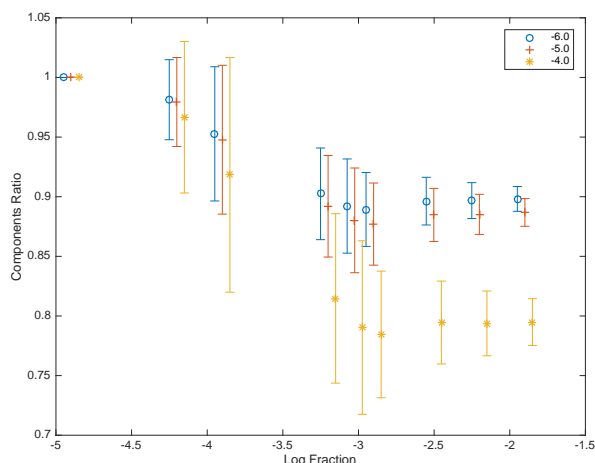

Figure 7: Results on the ratio of the number of connected components in the particles graph between the start and the end of the simulation as a function of the fraction (in log) of the reactions. Values are mean $\pm$ standard deviation. A small jitter has been added on x values for clarity.

The same results hold when looking at the dynamics i.e the actual time (in time steps) it took to complete the 2,000 reactions (see Fig 8). Conformation are the fastest reactions but must occur when molecules are already formed. Therefore interesting situation occurs when there is a mix of bimolecular and conformation to keep the system going.

Discussion

Artificial chemistries are extremely useful to understand and develop artificial life simulations. It will prove to be undoubtedly interesting in the future in the context of metabolic networks – either for theoretical considerations or for artificial recreation of entire cells. In this context, intricate and complex chemistries will be needed to create complex and open-ended simulated environment that could yield non trivial and emergent properties.

Among complex chemistries, we used the Hutton Artificial chemistry (HuAC) that, while very general, can generate complex chemistries by acting on nodes on molecules
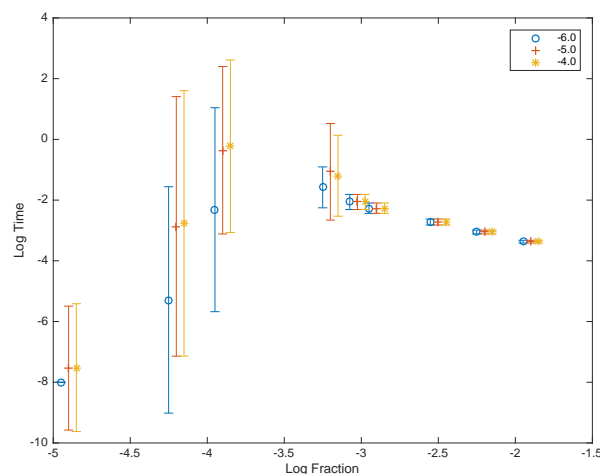

Figure 8: Total time (in log) at the end of the simulation (2,000 reactions) as a function of the fraction (in log) of the numbers of reactions. Values are mean $\pm$ standard deviation. A small jitter has been added on x values for clarity.

considered as connected graphs. The drawbacks of HuAC is linked to its spatial nature dependency – both in computational power and the unrealistic dependence on 2D constraints to function.

We provided here a Stochastic Simulation Algorithm that recreates HuAC in a fully stirred 3D medium. HuAC has problems of being locally constrained and had spatial correlation dependencies that do not exist in well- stirred medium. Even though we have argued and showed elsewhere that spatial correlations should not be ignored in the context of real cell signalling we contend that the loss of spatial correlation in the context of HuAC is a net gain due to its strongly 2D dependence.

We tested the impact on the Hutton's original replicator. We show that simple diffusion in a high concentration medium achieves the same result as his environmental 'wash'. We showed also that race conditions between reactions can stop the replication process altogether.

Our simulator allows to perform long simulation and we could test the replication process to its limit. By adding atoms stochastically, we showed that the result replication process was able to stabilise and provide correct replication size (on average) provided we waited long enough for episode of enormous compound to subside.

We finally tried to build Random Chemical World and see some of their simple properties when the reactions are chosen randomly. However this work is preliminary and random selection of reaction will not be susceptible to provide complex behaviours. A higher number of reaction that are linked together (e.g by being a chain between reactants and products) will probably yield more reactions time and

more molecules graphs modifications. In addition, the starting 'soup' of randomly connected atoms is also unlikely to provide interesting situations and we should investigate the impact on a initial set of bigger molecules. Not surprisingly, both these requirements are indeed met in the replicator situation. Therefore future works should include the study of random 'graphs' of reaction instead of random reactions.

The simulator, which that we called STAARC, can be used to scale up to very general and procedural chemistry. Scale up in terms of size and time but also in the number of reactions that can be handled. Our hope here is that it will prove to be well suited for study of artificial evolution.

References

Benkö, G., Centler, F., Dittrich, P., Flamm, C., Stadler, B. M., and Stadler, P. F. (2009). A topological approach to chemical organizations. *Artificial Life*, 15(1):71–88.

Benkö, G., Flamm, C., and Stadler, P. F. (2005). Explicit collision simulation of chemical reactions in a graph based artificial chemistry. In *Advances in Artificial Life*, pages 725–733. Springer.

Caré, B. R. and Soula, H. A. (2011). Impact of receptor clustering on ligand binding. *BMC Syst Biol*, 5:48.

Caré, B. R. and Soula, H. A. (2013). Receptor clustering affects signal transduction at the membrane level in the reaction-limited regime. *Phys Rev E Stat Nonlin Soft Matter Phys*, 87(1):012720.

Dittrich, P., Ziegler, J. C., and Banzhaf, W. (2001). Artificial chemistries—a review. *Artificial life*, 7(3):225–275.

Dorin, A. and Korb, K. B. (2007). Building virtual ecosystems from artificial chemistry. In *Advances in Artificial Life*, pages 103–112. Springer.

Ducharme, V., Egli, R., and Legault, C. Y. (2012). Energy-based artificial chemistry simulator. In *Artificial Life*, volume 13, pages 449–456.

Gillespie, D. T. (2007). Stochastic simulation of chemical kinetics. *Annual review of physical chemistry*, 58:35–55.

Gillespie, D. T., Hellander, A., and Petzold, L. R. (2013). Perspective: Stochastic algorithms for chemical kinetics. *J Chem Phys*, 138(17):170901.

Hutton, T. J. (2002). Evolvable self-replicating molecules in an artificial chemistry. *Artificial life*, 8(4):341–356.

Hutton, T. J. (2003). Simulating evolution's first steps. In *Advances in Artificial Life*, pages 51–58. Springer Berlin Heidelberg.

Hutton, T. J. (2004). A functional self-reproducing cell in a two-dimensional artificial chemistry. *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9)*, pages 444–449.

Hutton, T. J. (2007). Evolvable self-reproducing cells in a two-dimensional artificial chemistry. *Artificial life*, 13(1):11–30.

Hutton, T. J. (2009). The organic builder: A public experiment in artificial chemistries and self-replication. *Artificial life*, 15(1):21–28.

Knibbe, C. and Parsons, D. P. (2014). What happened to my genes? insights on gene family dynamics from digital genetics experiments. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 33–40.

Lenaerts, T. and Bersini, H. (2009). A synthon approach to artificial chemistry. *Artificial life*, 15(1):89–103.

Lucht, M. W. (2012). Size selection and adaptive evolution in an artificial chemistry. *Artif Life*, 18(2):143–63.

Oohashi, T., Ueno, O., Maekawa, T., Kawai, N., Nishina, E., and Honda, M. (2009). An effective hierarchical model for the biomolecular covalent bond: An approach integrating artificial chemistry and an actual terrestrial life system. *Artificial life*, 15(1):29–58.

Rocabert, C., Knibbe, C., and Beslon, G. (2015). Towards a Integrated Evolutionary Model to Study Evolution of Evolution. In *EvoEvo Workshop (Satellite workshop of ECAL 2015)*, York, United Kingdom.

Suzuki, H. and Dittrich, P. (2009). Artificial chemistry. *Artificial life*, 15(1):1–3.

Szabo, A. (1989). Theory of diffusion-influenced fluorescence quenching. *The journal of physical chemistry*, 93(19):6929–6939.

Tominaga, K., Suzuki, Y., Kobayashi, K., Watanabe, T., Koizumi, K., and Kishi, K. (2009). Modeling biochemical pathways using an artificial chemistry. *Artificial life*, 15(1):115–129.

Van, A. L., Soula, H. A., and Berry, H. (2014). Space-induced bifurcation in repression-based transcriptional circuits. *BMC Syst Biol*, 8:125.

Thresholds in Messy Chemistries

Nathaniel Virgo[1]

[1]Earth-Life Science Institute (ELSI), Tokyo, Japan
nathanielvirgo+alifexv@gmail.com

Many of the chemistries studied by chemists as being relevant to the origins of life produce a combinatorial explosion of products. Such chemistries include Miller-Urey chemistry, HCN polymerisation, Fischer-Tropsch synthesis and the Formose reaction, to name just a few. As well as producing a huge variety of products, these reactions can produce some of the basic molecules that comprise living systems, such as amino acids or sugars. The combinatorial explosions occur because the basic building blocks of organic chemistry can be put together in a huge variety of ways; the number of possible molecules that can be made in these systems grows exponentially (or faster) with their size. Moreover, these networks do not have obvious symmetries or other forms of structure that would make them easy to model and analyse.

Researchers in the origins of life have traditionally regarded such messy combinatorial explosions as a problem that needs to be overcome (e.g. Schuster, 2000). However, technology is now progressing to the point where we can explicitly map out the products produced by these chemistries and the networks of reactions that lead to them (e.g. Andersen et al., 2013). The emerging field of systems chemistry has begun to shift the focus away from static descriptions of complex chemical systems and towards an understanding of their dynamics. With this increased interest comes the realisation that the products of messy prebiotic chemistries are not necessarily just an inert "tar" but may be extremely complex dynamic systems in their own right, which we have simply lacked the right tools to study up to now.

This raises a number of important questions regarding the dynamics of such complex, messy chemistries. One such question is whether a sufficiently large and complex reaction network can behave fundamentally differently from what is possible in smaller, cleaner networks. Here I demonstrate an example of this, with the aid of a simple toy example. The example shows the existence of a phase transition, leading to a threshold effect of a kind that cannot occur in a small reaction network. This threshold is closely related to Eigen's error threshold (Eigen, 1971). The difference is that while Eigen's threshold occurs in a "clean" chemistry (template replication) with plenty of symmetry, ours occurs in a model designed to represent a messy chemistry with little structure.

Let us imagine that that there are N chemical species, each of which can react to form some of the other species from some surrounding milleu. The probability that species i causes the formation of species j is assumed to be constant (with value p) and independent of i and j.

This is essentially the same as Kauffman's (1986) model of the formation of autocatalytic sets, except that we have simplified it further to remove the specific structure associated with cleavage and ligation of peptides, since we are interested in a much broader class of chemistries. We interpret our model in terms of small-molecule organic chemistry rather than peptides, as described in (Virgo and Guttenberg, 2015; Virgo et al., 2016). As in Kauffman's model, this model has a percolation transition, meaning that in the large-N limit, if $p > 1/N$ there will be a single giant autocatalytic set consisting of most of the species in the system. In the simulations below we set the kinetic rates of all the catalysis reactions to be equal, with the value k, but one can show analytically that this makes no qualitative difference to the results of the model, as long as the kinetic rates are chosen from a distribution with bounded variance.

We are interested in the case where p is above the percolation threshold. In this regime, if any amount of any species is added to the system, it will eventually produce some amount of almost every species. We are using this simple model to stand in for a much more complex autocatalytic system with a combinatorial explosion. As the amount of matter in the system grows, it samples more and more of the combinatorially huge space of species available to it.

Let us now suppose that within this giant autocatalytic set there exists a smaller subset that can collectively produce its own members at a greater rate. For ease of exposition, we take this set to be a single species that catalyses its own production at a rate k^*, although similar results follow in the more general case. We can ask the following question: can this small subset produce itself faster than the "messy" autocatalytic set that contains it? If so, then if we were to perform this reaction experimentally, we might find that once the "fast" set was discovered it would come to dominate
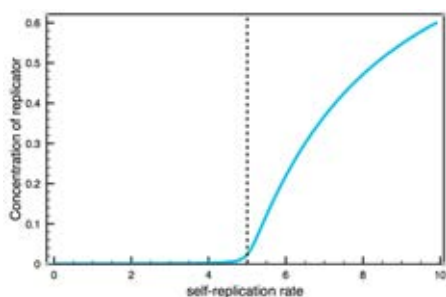
Figure 1: The relative concentration of the self-replicating molecule, plotted as a function of its replication rate k^*. For this plot the values $N = 4000$ and $p = 5/N$ were used. With a greater value of N the threshold becomes sharper.

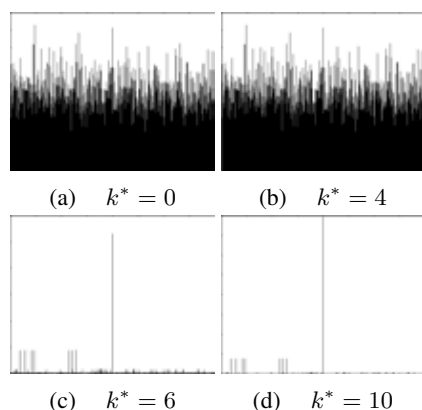(a) $k^* = 0$ (b) $k^* = 4$

(c) $k^* = 6$ (d) $k^* = 10$

Figure 2: Relative concentration profiles for various values of the replicator's growth rate k^*. In each plot, the species are enumerated along the x axis, with the relative concentration shown on the y axis. The self-replicating species is in the centre of the x axis. Note the stark difference between plots (a) and (b) with k^* below the threshold, versus (c) and (d) above it.

the composition, so that mass spectroscopy would reveal a highly peaked concentration profile, rather than a heterogeneous sampling of all possible products.

Figure 1 shows the relative concentration of a rapidly self-replicating species as a function of its replication rate. (That is, we normalise the vector of concentrations by its sum.) The threshold occurs when $k^* = kp/N$, with k being the rate at which the other species catalyse one another's formation. Below the threshold, all species grow at similar rates. Increasing k^* leads to only a slight relative increase in the amount of the replicator molecule and no change in the relative concentrations of the other species. However, above the threshold the concentration profile changes, becoming dominated by the replicator, and to a lesser extent, by the species whose production can be directly catalysed by it, as shown in figure 2. (This is closely analogous to the "mutational halo" in the case of Eigen's error threshold.)

The significance of this result lies not in the naïve scenario presented, but in showing that such threshold phenomena can exist in messy chemistries at all. Previously two distinct phase transitions were known in prebiotic chemistry models. The error threshold is a phase transition in a chemistry, but not a messy one. Kauffman's model exhibits a phase transition in a messy chemistry, but it is in the opposite direction from ours (i.e. from clean to messy) since once the threshold is passed the system explodes out into the whole combinatorial space. This new model shows a phase transition in which an initially messy chemistry can spontaneously constrain itself to a specific part of its accessible phase space.

We suspect that many more threshold phenomena of this nature will be found, in models much closer to chemical reality than the toy one we have presented. Indeed we suspect that the phase "more is different" (Anderson, 1972) applies to the size of reaction networks as much as it does to the size of a physical system, with phase transitions being a common and generic phenomenon in the dynamics of complex chemistries. Perhaps in time we will find transitions that lead not just to replication of a fixed set of species but to a complex, metabolism-like network of molecular interactions, or to replication with heritable variation, and thence to natural selection.

Acknowledgements.

References

Andersen, J. L., Andersen, T., Flamm, C., Hanczyc, M., Merkle, D., and Stadler, P. F. (2013). Navigating the chemical space of HCN polymerization and hydrolysis: Guiding graph grammars by mass spectrometry data. *Entropy*, 15:4066–4083.

Anderson, P. W. (1972). More is different. *Science*, 177(4047):393–396.

Eigen, M. (1971). Selforganization of matter and evolution of biological macromolecules. *Naturwissenschaften*, 58(10):465–523.

Kauffman, S. (1986). Autocatalytic sets of proteins. *Journal of Theoretical Biology*, 119:1–24.

Schuster, P. (2000). Taming combinatorial explosion. *PNAS*, 97(14):7678–7680.

Virgo, N. and Guttenberg, N. (2015). Heredity in messy chemistries. In Andrews, P. et al., editors, *Proceedings of the European Conference on Artificial Life 2015*, pages 325–332. MIT Press.

Virgo, N., Ikegami, T., and McGregor, S. (2016). Complex autocatalysis in simple chemistries. *Artificial Life*, early access publication.

Emergent Bonding Properties in the Spiky RBN AChem

Mihail Krastev[1,3], Angelika Sebald[2,3], Susan Stepney[1,3]

[1]Department of Computer Science, University of York, UK
[2]Department of Chemistry, University of York, UK
[3]York Centre for Complex Systems Analysis
mk599@york.ac.uk

Abstract

We present a subsymbolic Artificial Chemistry (ssAChem) in which all properties relevant to bonding are emergent from the underlying dynamical system (an RBN). We explore this ssAChem by evolving a seed set of atomic particles and showing the type of composite particles the system can produce.

INTRODUCTION

The field of Artificial Chemistry (AChem) has produced numerous models of chemical systems over the years. These models often have a symbolic representation of atoms or molecules, a defined set of possible reactions, and rely on some form of environment such as a 2D lattice or a well mixed reactor (Dittrich et al., 2001).

In these approaches bonding occurs via explicitly defined reaction rules through a grammar-like notation. A reaction results in a new symbol replacing one in the reactor (McMullin, 1997; Varela et al., 1974; Ono and Ikegami, 2001; Madina et al., 2003) These systems are not always mass conserving. In mass-conserving systems bonds form links between two particles (Hutton, 2002, 2005, 2007) In both cases reaction rules are explicitly defined aiming to explore specific behaviours. Another approach (Fontana and Buss, 1994; Dittrich and Banzhaf, 1998; Banzhaf et al., 1999; Hickinbotham et al., 2010) uses one reactant as an operator and another one as an operand with the result being the reaction product. In such systems reactions are emergent from the properties of the particles themselves.

In subsymbolic Artificial Chemistries (ssAChems) (Faulconbridge et al., 2010, 2011; Faulconbridge, 2011) particles are defined as systems with internal properties. Bond formation is a result of these interacting in a predefined way. Reactions are an emergent property of interaction. Overall the subsymbolic system can be seen as being less complicated, since it does not define individual behaviours like most symbolic approaches do. However, because bonding properties are now emergent and the reaction algorithm universal for all possible particles, the system is significantly more expressive with a huge combinatorial search space of possible reactions.

Random Boolean Networks (RBNs) (Kauffman, 1969; Kauffman et al., 2003) are an attractive dynamic system on which to base ssAChems. They provide a large number of exploitable properties, are computationally inexpensive and can easily be combined to produce analogues to molecular structures. RBN-World (Faulconbridge et al., 2011) itself, while having emergent bonding properties, still has many of its properties externally defined. Here we introduce the *Spiky RBN* model, and a simple reaction mechanism which allows for bond formation and decomposition. Like in RBN-World we use an RBN as the subsymbolic system. However, our mapping of subsymbolic to atomic particle properties is significantly different. Our aim is to build an ssAChem where all properties relevant to bonding are fully emergent from the underlying structure and dynamics.

The Spiky RBN Model

Here we first introduce the Spiky RBN, which defines our atomic particles and their properties. Next we give the collision and stability criteria, which determine if a link can be formed and if a link is stable. Then we explain the linking mechanism that describes what a link consists of and how to form one. To emphasise that we are not simulating real atoms or chemicals, we use the following terminology. Particles are connected by *links*. An *atomic* particle is a particle with no links (in our model it is a single Spiky RBN); no operations within our system can break down an atomic particle. A *composite* particle consists of two or more atomic particles connected via link. References to a *particle* mean either an atomic or a composite particle.

The boolean networks are random since our interest is in emergence of properties and dynamics as oppose to engineering of properties towards specific dynamic behaviour. Engineering specific boolean networks would limit the systems dynamics to only those that we have encoded.

Atomic Particle

An atomic particle is a small RBN, and its properties emerge from the dynamics. Particles are linked to form larger RBNs, with their own emergent dynamics. In the original RBN-
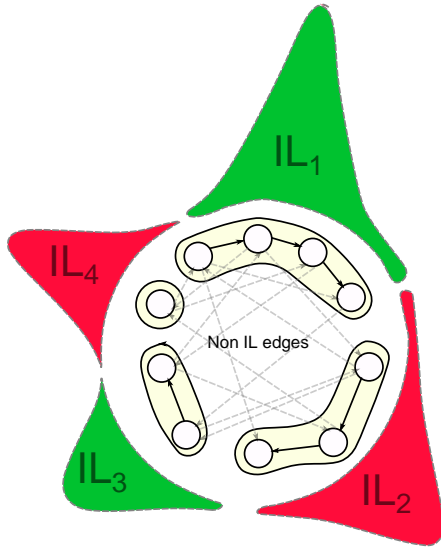
Figure 1: Diagram of the Spiky RBN. Nodes are separated into Interaction Lists (IL) of varying size based on Algorithm 1. Edges in solid lines are those which are part of the IL and can be used in linking. Dashed line edges are part of the RBN topology that cannot be changed due to linking. Note that there is an IL with only one node and no edges; it cannot form a link. Each IL spike can be seen, with colour denoting sign and size denoting magnitude.

Data: N: list of all nodes in the RBN ordered by least influential first
while N *is not empty* **do**
    Remove first node n from N;
    Create new Interaction List IL_i;
    Add n to IL_i ;
    while $\exists\, n' \in N$ *where* n *is an input to* n' **do**
        Remove n' from N;
        Add n' to IL_i;
        $n \leftarrow n'$;
    end
    i++ ;
end

Algorithm 1: Building Interaction Lists

world, particles had two special 'bonding nodes' added arbitrarily; in our Spiky RBN model, the number, location, and properties of these nodes are emergent.

The core of the model is the subsymbolic representation of the atomic particle. The RBN is split into Interaction Lists (IL) made up of RBN nodes as shown in Fig. 1. Each IL is a list of nodes where each subsequent node takes direct input from the previous node in the list. The first node is the only one with unspecified inputs. ILs are constructed by following connected nodes where the next node is chosen based on the number of its outgoing edges, as defined in Algorithm 1. We attempted a few methods of picking first and subsequent nodes least influential gave a good balance between number of nodes per IL and number of ILs.

The ILs partition the nodes: every node belongs to exactly one IL, and every IL has at least one node in it. ILs have no effect on the topology of the RBN; they are a logical grouping.

ILs form the basis for interaction between particles in the Spiky RBN model, replacing the arbitrary binding site in RBN-World (Faulconbridge et al., 2011). The size of the IL (the number of nodes it contains) and the number of ILs in a particle are all derived purely from the topology of the RBN. Each IL has a numerical property referred to as the *spike* (Fig. 1), which determines if a link will form and remain stable. The spike value is calculated over the attractor cycle

as follows.

The value of node x at RBN state s, $x_{s_{value}}$ is 1 if it is in a 'true' state and -1 if it is in a 'false' state.

$$x_{s_{value}} = \begin{cases} 1, & \text{if } x_{s_{state}} = T \\ -1, & \text{if } x_{s_{state}} = F \end{cases} \quad (1)$$

The value of node x over one cycle of the attractor x_{value} is

$$x_{value} = \sum_{s=t}^{s=t+c} x_{s_{value}} \quad (2)$$

where t is the first state of the attractor and c is the attractor length.

The spike for IL$_1$ of particle A, S_{A1}, is the sum of all node values for nodes in that IL.

$$S_{A1} = \sum_{x \in IL_1} x_{value} \quad (3)$$

This gives us a spike with both a magnitude and a sign. It is constrained by the attractor length c and the number of nodes in $IL_{A1_{size}}$:

$$-IL_{A1_{size}} c \leq S_{A1} \leq IL_{A1_{size}} c \quad (4)$$

We calculate the attractor of a RBN from an initial state of all 'false'.

An atomic particle has three properties: the number of ILs, the size of each IL, and the spike of each IL. The first two are a function of the RBN topology and the third is a function of the RBN dynamics. Because all are deterministically calculated, two identical RBNs will produce two identical atomic particles with identical behaviours. The number of ILs gives the maximum number of links that a particle can form. The size of each IL determines if it can be part of a link and how severely a link will change the topology of the particle. The spike dictates which specific set of bonds is possible.

Collision and Stability Criteria

The second component of the model are the collision and stability criteria. The collision criterion dictates what must be true in order for a link to form. The stability criterion dictates what must be true in order for a link to continue to exist.

A link can form between ILs chosen from two different particles. The ILs must have size >1 in order for a link to be possible. How particles and ILs are chosen depends on the reactor type. In an aspatial well mixed reactor two random particles can be chosen and a random IL on each. In a 2D lattice reactor particles in adjacent sites and the nearest ILs can be chosen.

The collision criterion states that:

$$S_{iA} + S_{jB} = 0 \qquad (5)$$

where S_{iA} is the spike of the ith IL of particle A. If the collision criterion is not met, the collision is considered elastic and the two particles do not form a link. If the collision criterion is met then a link forms as described below. Link formation results in a change in RBN topology and consequently in a possible change to partial linking properties. After the bond construction all partial linking properties are recalculated and used to check against the stability criterion.

Like collision criterion, the stability criterion states that:

$$S'_{iA} + S'_{jB} = 0 \qquad (6)$$

where S'_{iA} is the spike of the ith IL of particle A after the bond has been formed. The stability criterion is checked not only for the newly formed link, but for every pair of ILs that are part of a link in the new composite particle. Decomposition results in a particle splitting into two or more fragments. Each IL that was part of a now broken link is free again. This means the topology has changed such that stability criterion is checked recursively until all links in all fragments meet the criterion. Because the stability criterion is checked for all links it holds true for every composite particle that is not currently attempting to link.

Link Structure and Formation

If the collision criterion holds then a link is formed. This is done by swapping pairs of nodes inputs between the two ILs as shown in Fig. 2. Edges which are part of the IL are swapped, starting with the edge outputting from the first node in the IL.

The maximum number of swaps possible is $n - 1$, where n is the size of the smaller of the two ILs.

A link can be constructed between any two IL of size >1. ILs of size 1 cannot link because they have no edges to swap. (It is possible to have a node that takes input from itself. We do not consider these.)

After link formation the spikes of all ILs in the new composite particle are recalculated, since link formation results
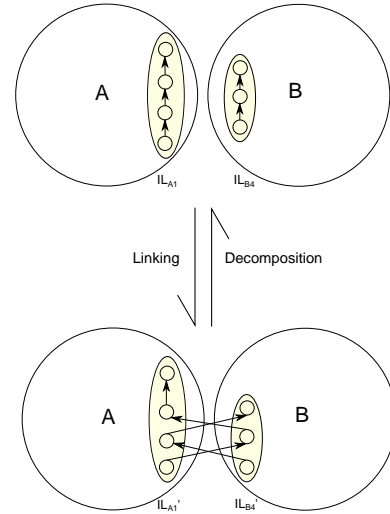

Figure 2: An example of the structure of a link between particles A and B. Note that two pairs of edges are swapped since IL_{B4} has a size of 3.

in a change in the underlying RBN topology. For any links that do not meet the stability criterion the link is decomposed by reversing the input swaps. When a link decomposes the break results in two new particles; again the spikes of the ILs are recalculated, and any further decomposition needed is performed. This process continues until the products are stable (meet the stability criterion), hence a single interaction between two reactants can result in multiple product particles. The algorithm for two atomic particles is shown in fig. 3.

Our links have a richer structure than those in the original RBN-World (Faulconbridge et al., 2011). Small ILs mean fewer swaps to form a link, resulting in less perturbation to the linking particles. This implies a higher chance that the spikes do not change and the link is stable. Larger ILs produce a larger change in topology and are therefore more likely to result in different spikes and so bond instability.

Experiment: Growing a Seed Set

The aim of the experiment is to use an evolutionary approach to generate interesting seed sets of atomic particles. One of the long term goals of the project is to add further parameters such as kinetics, variable link strengths, and geometry to the model, in order to see the effects these properties have on the dynamics of the system. In order to understand and compare the effects of these parameters we need an exemplar dynamic system: a set of atomic particles with nontrivial dynamic properties. Finding such a set can be difficult since the search space is vast and there is no way to predict dynamic behaviour without simulation. We can state undesirable characteristics:
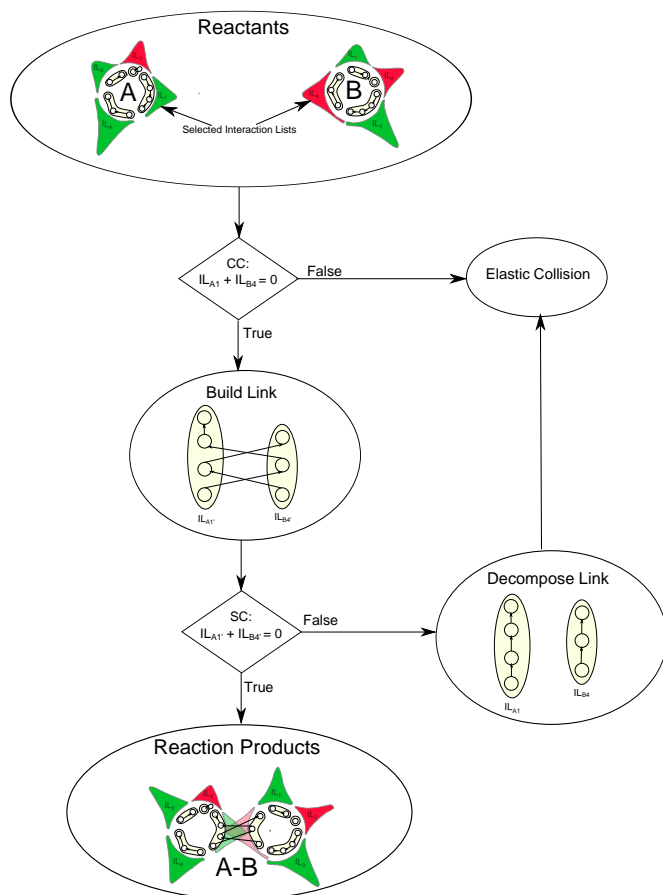
Figure 3: Reaction algorithm between two atomic particles. First an IL is selected on each. The collision criterion is checked. If it passes then a link is formed (product particle AB). The stability criterion is checked for the ILs that are part of the link. If it fails then the link decomposes. Note that ILs not involved in links may have had their spikes changed due the effect of the link on the new composite RBN.

- Overly restrictive : most reactions are elastic and do not result in larger composite particles. The system quickly reaches a stable state with no reactions occurring.

- Overly permissive : almost all reactions result in stable links. The system quickly congeals into a single large composite particle.

- Chaotic : almost all reactions between composite particles result in decomposition of links. The system is reactive but larger composites are quickly destroyed.

Reactor

For this experiment we use an aspatial reactor initialised with 20 unique atomic particles. The reactor attempts 1,000 links by picking two particles at random, picking an IL on each at random, and attempting a link. If the reaction is successful all reactants *and* products are added to the reactor.

At any time the reactor contains one copy of each composite particle that has been generated so far, plus the initial 20 atomic particles. In effect our system is a well stirred reactor with equal concentrations of all particles. This is a rough exploration of the possible behaviour the seed set can produce.

Fitness Metrics

We use a fitness function to describe the type of system we are looking for. We calculate a fitness per seed set in the population, as well as a fitness per atomic particle. There are five measures on which we base our fitness function:

C The number of unique composite particles that the system creates after a set number of reaction attempts.

V Variance in observed composite particle size.

L Variance in number of links per atomic particle in a composite particle.

R Percentage of attempted reactions for which the new bond is stable.

P Number of unique links that an atomic particle has been observed as forming. For example if atomic particle A has only ever formed bonds with atomic particle B and itself, then $P = 2$.

These characteristics form the basis of our fitness function. C, V, L and R provide an overall fitness for our seed set. P provides an individual fitness for each atomic particle within the seed set. We use a rank based approach, which removes the need to provide weights for the components of the fitness function.

Fitness Functions

The fitness of reactor i is:

$$fr_i = Rank(C_i) + Rank(V_i) + Rank(L_i) + Rank(R_i) \tag{7}$$

where $Rank(C_i)$ is the rank of reactor i when the reactors are ordered by lowest to highest. Since our population is made of 20 individuals and there are four ranks, fr_i is constrained to

$$4 \leq fr_i \leq 80 \tag{8}$$

The fitness of atomic particle j in reactor i is the number of unique bonds it can form, P_{ij}

$$f_{ij} = P_{ij} \tag{9}$$

With a seed set of 20 atomic particles f_{ij} is constrained to

$$0 \leq f_{ij} \leq 20 \tag{10}$$

The mutation function replaces the atomic particle j in reactor i with a new random one with a probability proportional to M_{ij}, where

$$M_{ij} = \frac{84 - fr_i}{1 + f_{ij}} \tag{11}$$

That is, fitter reactors and fitter particles are mutated less.

$$0.19 < M_{ij} \leq 80 \tag{12}$$

Exploratory Algorithm

To generate atomic particle sets we use an algorithm similar to clonal selection (De Castro and Von Zuben, 2000, 2002) (see Fig. 4). Our population is made of 20 reactors. Unlike normal clonal selection, each population member produces exactly one clone by mutating atomic particles based on M_{ij}. This is because our aim is not specific optimisation but rather exploration for possible seed sets with favourable behaviours. In order to ensure that good seeds propagate through the generations we include a low 5% crossover chance. The crossover function replaces the three lowest participation particles in a set with the three highest participation particles from another set. When crossover occurs we ensure that the resultant seed set has 20 unique atomic particles by making sure the incoming particles are not already in the seed set. The crossover probability is kept low because again we are interested in diversity. Also, due to the nature of the system, high fitness of a particle in one reactor does not necessarily imply high fitness in another reactor. This is a desirable trait since high fitness in all reactors would suggest the particle is overly permissive and can bond with almost everything.

Results

The experiment was run with RBNs of $K = 2, N = 12$ forming atomic particles. We first look at the behaviour of the reactors over the generations and then give an example particle from the best reactor at the end of the run.
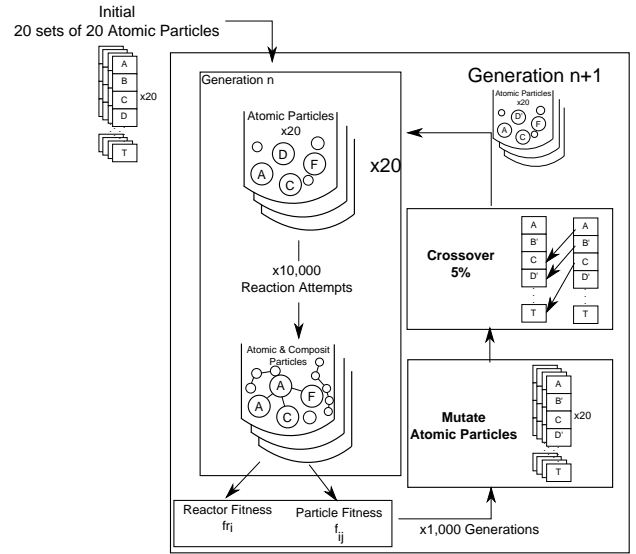

Figure 4: Exploration algorithm. The algorithm is initialized with 20 sets of 20 atomic particles each. Each reactor then attempts 1,000 reactions. We then calculate reactor fitness and particle fitness, mutate and perform crossover to get the new population for gen $n + 1$. This repeats for 100 generations.

Exploratory Algorithm Performance

Fig. 5 shows how the values of C, V, L and R change over the generations.

Over the generations there is an increase in the distribution's upper quartiles suggesting some reactors are improving and finding better seed sets.

The median values in each graph fluctuate, which is to be expected: even if no mutation or crossover is experienced there is no guarantee that a successful reactor will reproduce its behaviour in the following generation. Because reactions are randomly chosen it is possible that a very reactive composite particle is not generated even if one is possible.

The median variance in size (Fig. 5b) stays very low for most of the experiment. This is due to reactors producing only composite particles of one size (most commonly size 2) giving a variance of 0.

Generation 68 shows a large increase in variance in particle size (Fig. 5b) compared to the previous generation. For that generation there is also an increase in median number of unique particles (Fig. 5a), and number of bonds formed (Fig. 5d) compared to the previous generation. The median variance in number of bonds per particle (5c) is lower then the previous generation however. This suggests that gen 68 produced many large composites which where mostly straight ribbons of particles with low branching.

The large number of outliers shows that while these low-reactivity systems are common we can also find more interesting examples. The reduction in outlier numbers towards
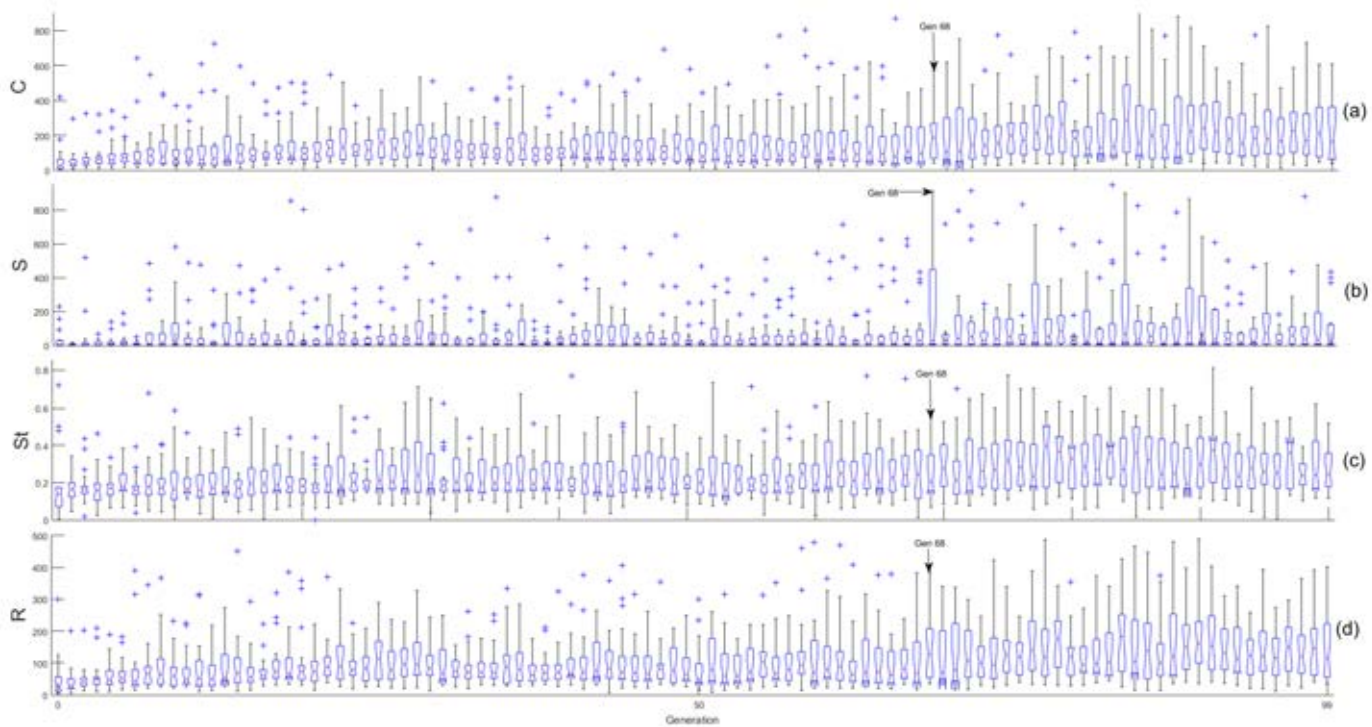
604

Figure 5: Distribution of the reactor measures per generation; some outliers are omitted. (a) C, number of unique particles; (b) V, variance of particle size; (c) L, variance of number of links; (d) R, stability. Generation 68 is highlighted for reference.
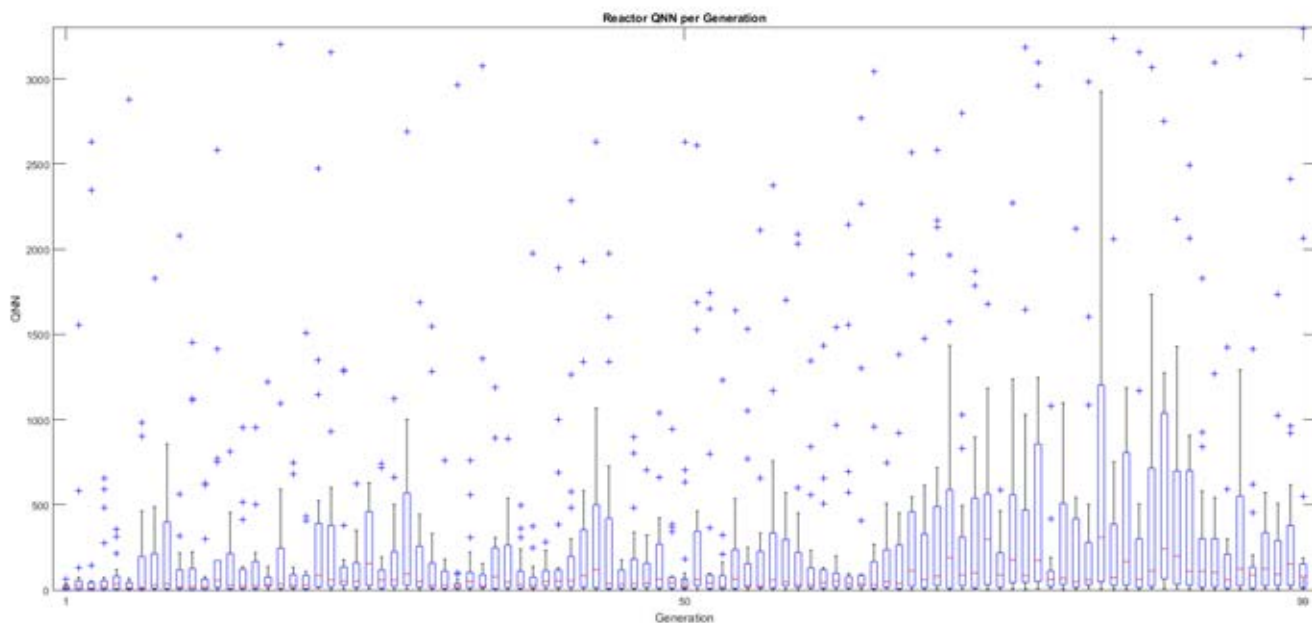

Figure 6: Distribution of the evolutionary activity measure QNN per generation for each reactor.

| Reactor	C	V	L	R
0	46	2.3667	0.18255	40
1	63	4.3401	0.19970	59
2	129	8.3091	0.20915	83
3	586	430.3	0.51866	307
4	76	7.6058	0.17437	58
5	229	21.694	0.38975	137
6	59	1.8248	0.17602	54
7	139	6.8823	0.27666	116
8	501	1607.6	0.36968	402
9	453	128.54	0.27755	298
10	149	16.221	0.20390	93
11	39	2.5063	0.12235	34
12	566	116.41	0.36923	280
13	274	398.26	0.11454	166
14	608	368.44	0.29997	338
15	266	31.143	0.28607	158
16	183	25.329	0.34497	140
17	58	4.797	0.18159	45
18	179	61.056	0.48978	114
19	50	2.2724	0.12439	38

Figure 7: Gen 99 Reactors
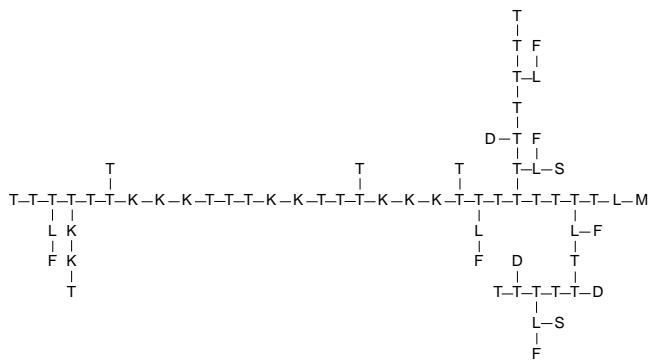
Figure 8: A composite particle produced by reactor 3, made of 65 atomic particles from 7 different species (T,K,L,S,F,M,D).

the end of the run together with an increase in distribution variance suggests that fit individuals have a positive influence on the population.

In order to check if the exploratory algorithm is producing positive evolutionary activity we use the QNN measure (Droop and Hickinbotham, 2012). Fig. 6 shows the QNN distribution per reactor over the generations. Large increases in the QNN distribution, especially towards the latter third of the experiment, suggest a period of strong evolutionary activity for most reactors. Large positive outliers are single reactors which are showing high activity, likely due to mutation. Again due to the random nature of the reactor we see fluctuations between generations.

Example product

Fig. 7 shows the reactor metrics for each reactor at the end of the last generation. In the final run four of the 20 reactors produced over 500 unique composite particles. Of these the best (based on f_r) was reactor 3 producing 586 particles of which 307 had newly created stable bonds.

Fig. 8 shows one of the largest generated composite particles. The main chain consists of T and K atomic particles. We see branching along the chain showing that the T atomic particle is capable of three links. Interestingly we also see that particle L allows other, non-K or -T particles to join the chain (specifically F, M and S). This gives the product compositional diversity. Like the T particle, L is capable of forming up to three links. However the reactor does not contain any long chain L composites, suggesting that L-L links are unstable. While the particle in Fig. 8 is stable there are still T atoms with only one link, suggesting that it could grow even further.

This particle is a product of 47 unique reactions. Most produce exactly one new unique composite particle. However three of the reactions produce two unique particles. This suggests that most reactions are combinatorial in nature. While reactions can produce multiple products, only previously unobserved products are recorded and added back to the reactor.

Branching is common in the final reactors; 15 out of 20 reactors have at least one composite where an atom has three links. However we have not observed a particle that can form four or more links. It is possible that changing the way ILs are constructed to ensure that each particle has at least 4 linking sites of size >2 would give more branching. However engineering particles in such a way is contrary to the core principle of having emergent properties. A more consistent approach would be to find naturally occurring particles of that nature and introduce them into the seed set.

Conclusions and Outlook

We have presented a new ssAChem based on the Spiky RBN model of an atomic particle. The core design principle of the model is to derive all properties relevant to linking from

the emergent properties of the RBN. We have shown that the method is capable of producing large composite particles with varying structures. Seed sets of atomic particles have been found which are reactive and produce a variety of possible composite particles, as well as a range of reaction paths which can be further explored.

Overall the Spiky RBN model seems to be a viable option for a fully emergent ssAChem. Future work will focus on expanding the available mechanisms beyond simple bonding and stability.

Firstly energetics will be introduced. Bonding and decomposition will depend on meeting collision and stability criteria as well as a probability proportional to reactor temperature. This could result in a relaxation of the stability criteria allowing for more composite species to exist for a short time.

Secondly a spatial 2D reactor will be introduced. Geometry of composites will be determined by number of bonds per particle as well as angles between bonds. The values will again be emergent from the sRBN properties.

As well as bonding, weaker inter-particle interactions could be considered. This could allow the emergence of organisation within a spatial reactor.

The sRBN model provides us with the flexibility to introduce the above mechanisms in a way that is fully emergent from the underlying organisation.

Acknowledgment

References

Banzhaf, W., Dittrich, P., and Eller, B. (1999). Self-organization in a system of binary strings with spatial interactions. *Physica D: Nonlinear Phenomena*, 125(1):85–104.

De Castro, L. N. and Von Zuben, F. J. (2000). The clonal selection algorithm with engineering applications. In *Workshop on Artificial Immune Systems and their Applications, Workshop Proceedings of GECCO*, pages 36–39.

De Castro, L. N. and Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, 6(3):239–251.

Dittrich, P. and Banzhaf, W. (1998). Self-evolution in a constructive binary string system. *Artificial Life*, 4(2):203–220.

Dittrich, P., Ziegler, J. C., and Banzhaf, W. (2001). Artificial chemistries—a review. *Artificial Life*, 7(3):225–275.

Droop, A. and Hickinbotham, S. (2012). A quantitative measure of non-neutral evolutionary activity for systems that exhibit intrinsic fitness. *Artificial Life*, 13:45–52.

Faulconbridge, A. (2011). *RBN-World: sub-symbolic artificial chemistry for artificial life*. PhD thesis, University of York.

Faulconbridge, A., Stepney, S., Miller, J. F., and Caves, L. S. (2010). RBN-World: The hunt for a rich AChem. In *ALife XII*, pages 261–268. MIT Press.

Faulconbridge, A., Stepney, S., Miller, J. F., and Caves, L. S. (2011). RBNWorld: Sub-symbolic artificial chemistry. In *ECAL 2009*, pages 377–384. Springer.

Fontana, W. and Buss, L. W. (1994). What would be conserved if "the tape were played twice"? *Proceedings of the National Academy of Sciences*, 91(2):757–761.

Hickinbotham, S. J., Clark, E., Stepney, S., Clarke, T., Nellis, A., Pay, M., and Young, P. (2010). Diversity from a monoculture: Effects of mutation-on-copy in a string-based artificial chemistry. In *ALife XII*, pages 24–31. MIT Press.

Hutton, T. J. (2002). Evolvable self-replicating molecules in an artificial chemistry. *Artificial Life*, 8(4):341–356.

Hutton, T. J. (2005). Replicators that make all their own rules. In *ECAL Workshop on Artificial Chemistry*.

Hutton, T. J. (2007). Evolvable self-reproducing cells in a two-dimensional artificial chemistry. *Artificial Life*, 13(1):11–30.

Kauffman, S., Peterson, C., Samuelsson, B., and Troein, C. (2003). Random boolean network models and the yeast transcriptional network. *Proceedings of the National Academy of Sciences*, 100(25):14796–14799.

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437–467.

Madina, D., Ono, N., and Ikegami, T. (2003). Cellular evolution in a 3d lattice artificial chemistry. In *ECAL 2003*, pages 59–68. Springer.

McMullin, B. (1997). SCL: An artificial chemistry in Swarm. Working paper 97-01-002, Santa Fe Institute.

Ono, N. and Ikegami, T. (2001). Artificial chemistry: Computational studies on the emergence of self-reproducing units. In *ECAL 2001*, pages 186–195. Springer.

Varela, F. G., Maturana, H. R., and Uribe, R. (1974). Autopoiesis: the organization of living systems, its characterization and a model. *Biosystems*, 5(4):187–196.

A Precarious Existence: Thermal Homeostasis of Simple Dissipative Structures

Stuart Bartlett[1,2] and Seth Bullock[2,3]

[1]Earth-Life Science Institute, Tokyo Institute of Technology, Tokyo, Japan
[2]Institute for Complex Systems Simulation, University of Southampton, Southampton, UK
[3]Department of Computer Science, University of Bristol, Bristol, UK
stuart.bartlett@elsi.jp

Abstract

We demonstrate the emergence of spontaneous temperature regulation by the combined action of two sets of dissipative structures. Our model system comprised an incompressible, non-isothermal fluid in which two sets of Gray-Scott reaction diffusion systems were embedded. We show that with a temperature dependent rate constant, self-reproducing spot patterns are extremely sensitive to temperature variations. Furthermore, if only one reaction is exothermic or endothermic while the second reaction has zero enthalpy, the system shows either runaway positive feedback, or the patterns inhibit themselves. However, a symbiotic system, in which one of the two reactions is exothermic and the other is endothermic, shows striking resilience to imposed temperature variations. Not only does the system maintain its emergent patterns, but it is seen to effectively regulate its internal temperature, no matter whether the boundary temperature is warmer or cooler than optimal growth conditions. This thermal homeostasis is a completely emergent feature.

Introduction

Life is a quasi-miraculous panoply of controls, feedbacks, interactions and diversity. Every time we think we have discovered its final limits, we stumble upon hidden surprises that propel us to once more rip up the rulebook. It remains arguably the greatest intellectual challenge of our time to understand how evolutionary forces have picked, crafted and re-worked physical and chemical mechanisms such that life wins. What makes this quest so difficult is that we still struggle to define the game that life is playing, and it also seems that some players have found ways to modify the rules mid-play and perhaps even cheat.

In order to break into the vault of life's mysteries, we must seek plausible trajectories. Trajectories that life may have taken here on Earth, that it could have taken, or even could take in other scenarios, to get from dead chemistry to open-ended complexity. One of the most characteristic features of organisms is their ability to carry out regulation and stabilisation in the face of external change. The reactions and interactions that life depends on cannot take place in an arbitrary range of conditions; in fact many have strict limitations.

The idea of homeostasis conjures ideas of regulating cell salinity, glucose concentration, body temperature or cell membrane lipid composition, to name but a few. In fact, it is a fundamental and necessary feature of all living organisms that some minimal set of internal variables are maintained within viable ranges.

It is easy to imagine many forms of proto-life, early in the history of our planet (and perhaps also other worlds), which exhibited only some, not all of the characteristics of life as we know it today. For example, one could think of an early metabolic reaction set being driven by a particular geochemical gradient. Out of the space of possible such reaction sets, there are a large number which are very sensitive to the exact details of the local conditions. So if such conditions were to change, many of those possible metabolisms would cease to function.

One can also imagine early forms of life that produced some kind of waste product that was toxic in sufficiently high concentrations. In this scenario early life could have easily poisoned itself into oblivion. This of course brings to mind the example of free oxygen production by photosynthetic organisms, and the danger posed to many forms of life from that oxygen.

Broadly speaking, part of the solution that life stumbled upon was cooperation. If there is a growing excess of a particular substance and there is the possibility that a new organism can somehow make use of that substance and move the system towards being materially cyclic, then it's likely that such an arrangement will emerge and persist. To be thermodynamically consistent, the new organism may have to use a novel energy source to carry out its recycling stage, and there are many examples of this in both modern and past life.

Once there are several ecological interactions taking place between a set of organisms, there will inevitably be feedbacks. Feedbacks between population sizes, concentrations of key chemical species, environmental variables, and many other factors. Despite the fact that such feedbacks did not arise from any intentional design process, those that persisted over evolutionary time now appear to be very well

tuned. Many negative feedbacks that we can identify appear to allow organisms to survive in a range of conditions, particularly those that are normally unfavourable to their emergence or survival. We can use hindsight and place evolutionary explanatory frameworks upon these observations, making arguments such as "groups of self-regulating organisms would have had a significant selective advantage over those groups that were unable to control key environmental parameters." However, the true chronology of life's discovery of these clever techniques, is veiled behind a shadow of a meandering story of change, and lost historical information.

Thus we are compelled to seek systems in which such biological functions emerge spontaneously, and perhaps get a window on possible narratives for life's ascension. Whether such sequences of events are similar to how life as we know it arose is very difficult, if not impossible to truly know. However, if we find that amongst the space of complex driven systems, a large number produce life-like dynamics, we at least can get a first foothold on the inevitability of life in the universe.

Our previous research sought to elucidate conditions in which non-living systems express life-like characteristics (Bartlett, 2014; Bartlett and Bullock, 2015; Bartlett et al., 2010), and in this work we continue in that endeavour. Having discovered systems in which non-living patterns spontaneously compete with one another for a common free energy source (Bartlett and Bullock, 2015), we questioned whether cooperation or symbiosis, might also readily emerge in simple physico-chemical systems.

Such dynamics can indeed be observed, and in this paper we will explore the spontaneous emergence of a temperature regulation mechanism in a shockingly simple non-living system.

We will first define the system in question and summarise its normal characteristic features. Then the modelling framework used to carry out our simulations will be briefly described. The next sections document the phenomenology of thermal Gray-Scott reaction diffusion (GSRD) structures and their resilience (or sensitivity) to temperature changes. We then illustrate a robust thermal homeostasis mechanism that emerges from the combined action of two sets of GSRD systems, before drawing conclusions in the final section.

The Quintessential Pattern-Former

The original GSRD system consists of two chemical species A and B, which are free to diffuse and react within a two-dimensional domain (Gray and Scott, 1985, 1994; Lee et al., 1994; Mahara et al., 2008; Pearson, 1993; Virgo, 2011). Species A is fed into the system at all points via porous walls, at a rate equal to $F(1 - \psi_A)$, where F is a constant and ψ_A is the concentration of A at that location. There is a non-linear autocatalytic reaction between the two species: $A + 2B \rightarrow 3B$. Substance B is removed at a rate $(F+R)\psi_B$, where R is a positive parameter (removal rate) which speci-

fies the rate of removal of substance B (over and above the feed rate F).

Despite its bare simplicity, this system is capable of exhibiting a myriad of dynamic chemical structures. Of course there are two trivial, non-structured attractors as well: one in which the reaction rate drops to 0 because B has all but disappeared from the system (perhaps if the reaction rate was too low to keep pace with the rate of removal of B and hence $\psi_A \rightarrow 1$ due to the boundary conditions), and one in which the reaction occurs at a constant rate homogeneously across the domain and both species exist at finite concentrations which do not vary with space or time.

Nonetheless, under a small but finite range of conditions, and when the system is initialised in the correct way, a broad range of stable structures can be observed (structures in the concentration fields of the two species). These patterns are the emergent result of a frustration between the inward diffusion and supply of A, the transformation of A into B, and the outward diffusion and ubiquitous removal of B. If there is a point at which ψ_A is relatively low, ψ_B is relatively high, and the reaction turns A into B faster than B is lost from that region, then a small structure, or soliton, can persist. There are patterns of many different morphologies, also emerging from a similar balance of physical effects. In this paper we are interested purely in the well-known, self-reproducing spot patterns (Lee et al., 1994; Pearson, 1993; Virgo, 2011).

Traditionally, these systems are modelled under an isothermal assumption, so temperature plays no role in their dynamics. However the role of temperature in chemical kinetics could not be more essential. In recent years we have carried out a range of studies on the effects of adding thermal dependencies and interactions to GSRD systems (Bartlett, 2014).

In the investigation documented here, the system in question was a duo of GSRD systems. Having observed a competitive dynamic between GSRD spots and convection cells in our previous work (Bartlett and Bullock, 2015), we intended to go further and explore the possibility of cooperation between dissipative structures. Thus in this paper, there are two GSRD systems placed within the same domain. They are dissolved in a solvent fluid which flows and advects any passive scalars according to conventional incompressible flow dynamics. The temperature is of course also free to vary as a function of space and time. We simply need to control the temperature at the upper and lower boundaries of the system. Chemical reactions can also release or absorb a certain amount of heat (per unit reaction rate), quantified by the enthalpy change ΔH (negative for exothermic reactions and vice versa).

The next section will describe the numerical algorithm that allowed us to re-create these model systems *in silico*.

Modelling Framework

The simulation of non-isothermal fluids advecting chemical species, which themselves are undergoing temperature-sensitive reactions is not a lightweight undertaking. While various 'traditional' numerical methods (e.g. methods based on numerical integration or discretisation, such as multi-physics solvers like COMSOL) exist for the solution of the governing equations of these systems, they come with a significant computational burden and tend to lack transparency. Since we were not concerned with absolute predictive accuracy, but instead with carrying out "opaque thought experiments" (Di Paolo et al., 2000), we made use of an extended Reactive Thermal Lattice Boltzmann Model RTLBM, developed by Bartlett (2014).

The basic LBM has been extensively used and evaluated for a variety of fluid flows, and its accuracy and efficiency for simple flows is now well established (see e.g. Wolf-Gladrow, 2000; Chen and Doolen, 1998, and references therein). The LBM has also been extended to include buoyancy driven convection of non-isothermal fluids, and its effectiveness for such flows is also well established (He et al., 1998; Shan, 1997; Peng et al., 2003).

LBMs capable of simulating reactive flows were less common until recent years, but these models are now steadily reaching maturity (Ayodele et al., 2011, 2013; Frouzakis, 2011). The specification, development and testing of the exact RTLBM used in this paper is described fully in Bartlett (2014); Bartlett and Bullock (2015), and hence we will not go into any further modelling details here.

Instead we will simply re-state the governing equations of the system in question. The flow field obeys the standard Navier-Stokes equations for an incompressible fluid that can experience buoyancy forces (under the Boussinesq approximation). The temperature field follows the advection-diffusion equation of a passive scalar (with sources and sinks from chemical reactions and boundary conditions):

$$\frac{\partial T}{\partial t} = \chi \nabla^2 T - \nabla \cdot (\mathbf{u}T)$$
$$-Ae^{-E_a/T(\mathbf{x},t)}(\psi_{A_1}\psi_{B_1}^2 \Delta H_1 + \psi_{A_2}\psi_{B_2}^2 \Delta H_2) \quad (1)$$

Finally, the four chemical species concentration fields obey reaction-diffusion-convection equations:

$$\frac{\partial \psi_{A_1}}{\partial t} = D_{A_1}\nabla^2 \psi_{A_1} - \nabla \cdot (\mathbf{u}\psi_{A_1})$$
$$-Ae^{-E_a/T(\mathbf{x},t)}\psi_{A_1}\psi_{B_1}^2 + F(1 - \psi_{A_1}) \quad (2)$$

$$\frac{\partial \psi_{B_1}}{\partial t} = D_{B_1}\nabla^2 \psi_{B_1} - \nabla \cdot (\mathbf{u}\psi_{B_1})$$
$$+Ae^{-E_a/T(\mathbf{x},t)}\psi_{A_1}\psi_{B_1}^2 - (F+R)\psi_{B_1} \quad (3)$$

$$\frac{\partial \psi_{A_2}}{\partial t} = D_{A_2}\nabla^2 \psi_{A_2} - \nabla \cdot (\mathbf{u}\psi_{A_2})$$
$$-Ae^{-E_a/T(\mathbf{x},t)}\psi_{A_2}\psi_{B_2}^2 + F(1 - \psi_{A_2}) \quad (4)$$

$$\frac{\partial \psi_{B_2}}{\partial t} = D_{B_2}\nabla^2 \psi_{B_2} - \nabla \cdot (\mathbf{u}\psi_{B_2})$$
$$+Ae^{-E_a/T(\mathbf{x},t)}\psi_{A_2}\psi_{B_2}^2 - (F+R)\psi_{B_2}, \quad (5)$$

where the diffusion coefficients are $D_{A_1} = 2D_{B_1} = D_{A_2} = 2D_{B_2}$, the supply and removal parameters are fixed at $F = 0.03$ and $R = 0.061$ and $\mathbf{u}$ is the local fluid velocity. Note that in contrast to the standard GSRD system, these equations have temperature dependent rates (incorporating the Arrhenius equation), which all share the same activation energy $E_a = 1.7$ and frequency factor $A = 3.1$.

Fragility

We are now in a position to explore the stability of a broad class of chemical patterns. Of course sustained structure formation on its own is grossly insufficient as an analogue for life. Many chemical systems show spatial structure (indeed the first that come to mind are the microscopic crystalline structures of many solid materials). However, if that structure is purely a function of external conditions, and has no means to actively respond to them, then it certainly cannot be considered life-like.

In this investigation we considered two GSRD systems in the same domain (they could only interact with each other implicitly through their thermal influence, there was no cross-diffusion or cross-reactions). We will begin at this stage with the assumption that the two reactions are neither exo- nor endothermic (they are thermally neutral, but their rates are sensitive to temperature). The steady states of such systems are shown in Figure 1 for three different temperatures. Note that in all figures, the temperature colourmaps are normalised by the same values, with red corresponding to $T \approx 3$ and pale blue corresponding to $T \approx 1.3$.

We can see that at $T = 1.5$, the standard self-replicating spot dynamic is reproduced. However there is a strong sensitivity of the emergent patterns upon temperature. At $T = 1.4$, the rate of the reaction is so slow that the spots struggle to reproduce and proliferate (Figure 1(a)). At temperatures of $T = 1.3$ and lower, the pattern formation all but ceases.

If the temperature is raised to, for example $T = 1.6$, a different phase of pattern emerges: more of an amorphous lamellar structure (Figure 1(c)). This is due to the thermal reaction rate enhancement causing the round spot structures to be unstable compared to the worm-like formations that arise instead.

Overall, we see that the delicate spot patterns cannot really tolerate changes of temperature and so their viable thermal range is $\Delta T_{nt} < 0.1$.
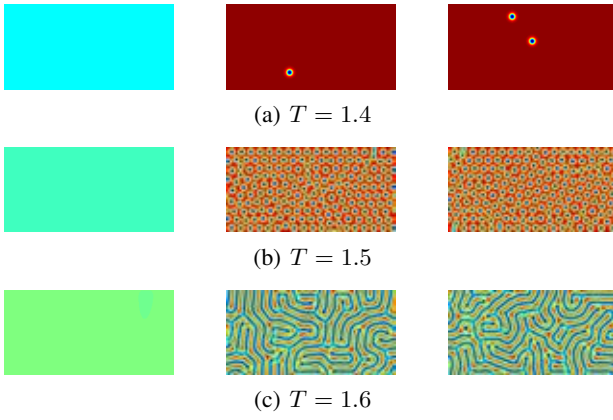
(a) $T = 1.4$

(b) $T = 1.5$

(c) $T = 1.6$

Figure 1: Steady state snapshots of the temperature (left column) and chemical order parameter fields, $\phi_{1,2} = \psi_{A_{1,2}} - \psi_{B_{1,2}}$, (centre column for system 1 and right column for system 2), for a double GSRD system. In these simulations, both reactions are thermally neutral ($\Delta H_{1,2} = 0$).

Self-destruction

In the previous section, we saw structures that could not exert any influence on their environment, and the result was a high degree of sensitivity to the key environmental variable: temperature. Perhaps if we add an extra coupling between the spot patterns and that crucial parameter, they will be less susceptible to thermal variations. If the reaction which drives their existence could interact with the temperature field, a self-stabilising feedback might be induced. An exothermic reaction might allow the spots to create more favourable local conditions if the background temperature was low and becoming a limiting factor. However, there is no guarantee that such a coupling would confer stability. What if it eroded the viability of the environment rather than enhanced it? This section will explore such a possibility.

As explored by our previous work (Bartlett, 2014; Bartlett and Bullock, 2015), exothermic spots exert a very strong positive feedback effect on the temperature of their surroundings. We illustrate the effect in Figure 2.

At the beginning of the simulation the spots begin to replicate as normal. However the additional heat released from the reaction begins to warm up the surroundings, which further augments the reaction rate. This then takes the system through different phases of pattern until eventually it is swamped and the temperature begins to diverge. The simulation in Figure 2 was carried out at $T = 1.4$. As one might imagine, higher temperatures simply reduce the time taken for the system to 'explode'.

If the boundary and initial temperature is taken down to $T = 1.3$, the runaway feedback does not occur but pattern formation is also completely suppressed (the low initial temperature and concomitantly low reaction rate mean that small initial fluctuations are unable to grow into stable

(a) $t = 1000$
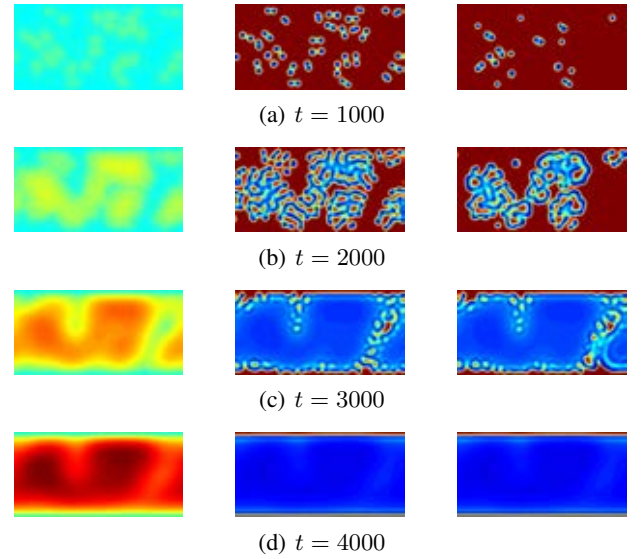
(b) $t = 2000$

(c) $t = 3000$

(d) $t = 4000$

Figure 2: Snapshots of the temperature and chemical order parameter fields for a double GSRD system. One species has an exothermic reaction ($\Delta H_1 = -20 \times 10^{-3}$, centre column), and the other has a thermally neutral reaction ($\Delta H_2 = 0$, right column). The boundary and initial temperatures were all fixed at $T = 1.4$.

structures). Thus exothermic spots are highly unstable, in fact more so than thermally neutral spots.

What then, of endothermic spots? Our general finding has been that these patterns suffer from self-inhibition (Bartlett, 2014; Bartlett and Bullock, 2015). By reducing the temperature of their surroundings, they reduce the rate of the reaction that drives them and this often proceeds until the patterns themselves are all but extinguished. An example of this is shown in Figure 3.

We see that the endothermic spots rapidly damp themselves out. The temperature reductions are quite local so the neutral spots (shown in the central column) only experience a limited amount of destructive cooling. However as the simulation proceeds, the endothermic structures eventually cause their own demise.

At higher temperatures this self-damping still persists. At $T = 4$, for example (Figure 4), there is no initial pattern formation, and the reaction proceeds at a high rate homogeneously across the domain. However small fluctuations near the boundaries eventually lead to a division of the system into two layers next to the upper and lower walls, where a string of stable spots form.

Within these two layers, in the bulk of the system, the reaction ceases because the reacting layers at either boundary extract most of the thermal energy from the interior. Thus two films of spots persist, making use of the large heat flux from the boundaries to prevent their own self-destruction.

(a) $t = 500$

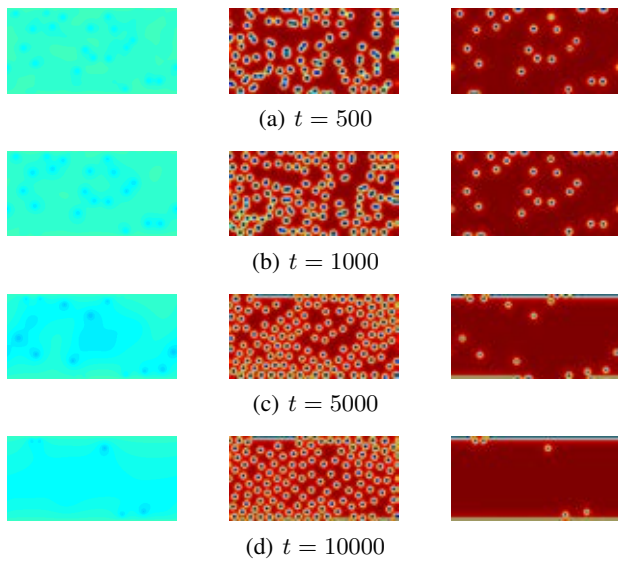(b) $t = 1000$

(c) $t = 5000$

(d) $t = 10000$

Figure 3: Snapshots of the temperature and chemical order parameter fields for a double GSRD system. One species has a thermally neutral reaction ($\Delta H_1 = 0$, centre column), and the other has an endothermic reaction ($\Delta H_2 = 25 \times 10^{-3}$, right column). The boundary and initial temperatures were all fixed at $T = 1.5$.

(a) $t = 4000$

(b) $t = 5000$

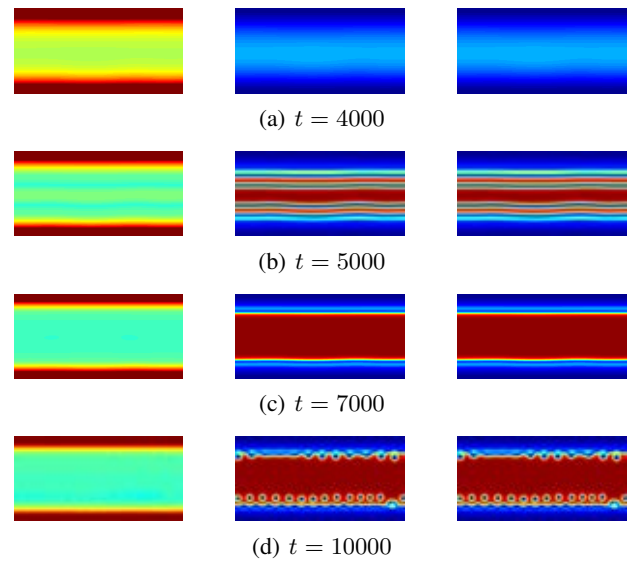(c) $t = 7000$

(d) $t = 10000$

Figure 4: Snapshots of the temperature and chemical order parameter fields for a GSRD system with two different spot species. One species has a thermally neutral reaction ($\Delta H_1 = 0$, centre column), and the other has an endothermic reaction ($\Delta H_2 = 25 \times 10^{-3}$, right column). The boundary and initial temperatures were all fixed at $T = 4$.

Despite this effect, the heat extraction from the spot persistence prevents them from spreading throughout the domain, so they do not proliferate. Thus even at very high temperatures, endothermic spots impose a fundamental limit on their own existence.

Harmony

In the previous sections we illustrated the fundamental fragility of self-reproducing spot patterns in thermal GSRD systems. If the reaction neither releases nor absorbs heat, the type of emergent pattern has a strong temperature dependence, and self-replicating spots only occur within a narrow thermal window. If the reaction of one of the spot species is exothermic (and the other is neutral), there is a runaway positive feedback and the temperature diverges, taking all chemical structures with it. Conversely, when one of the two reactions is endothermic (and the other neutral), the effect is one of self-limitation. The cooling from the reaction has the effect of reducing its own rate, which in turn leads to the damping out of patterns.

At this stage we might naturally wonder whether some combination of exo- and endothermic spots might be able to mutually stabilise one another. To harmonise means to agree, to complement, and so we will now experiment with systems comprised of complementary spot systems. They will consist of an exothermic spot species with $\Delta H_1 = -20 \times 10^{-3}$, and an endothermic species with $\Delta H_1 = $

25×10^{-3}. The slightly higher magnitude of the enthalpy change of the endothermic reaction is necessary to ensure that the exothermic reaction cannot push the system into a positive feedback cycle from a small fluctuation (e.g. a temporary period where there are a smaller than average number of endothermic spots).

If we proceed as before with initial and boundary temperatures of $T = 1.5$, we indeed observe stable behaviour, as shown in Figure 5(b). There is stable self-replication and the heat emitted by species 1 (substances A_1 and B_1) is compensated for by the heat absorption of species 2 (substances A_2 and B_2).

These symbiotic patterns can also provide a stable environment for themselves at higher temperatures, such as $T = 2.0$. After an initial transient phase, the system settles into a steady state with a stable population of both spot species (Figure 5(c)). The reason the structures survive is that they are carrying out a form of temperature regulation. Despite the boundaries being warmer than ideal, the combined spot system is able regulate the bulk temperature to $T_{in} \sim 1.5$. If the boundary and initial temperatures are less than $T = 1.45$, the endothermic spots extinguish themselves before the exothermic ones can provide compensatory heating.

In conclusion, this combined synergy of an exothermic with an endothermic spot species yields a viable temperature range of $\Delta T_{sym} \sim 1$, much greater than the equivalent
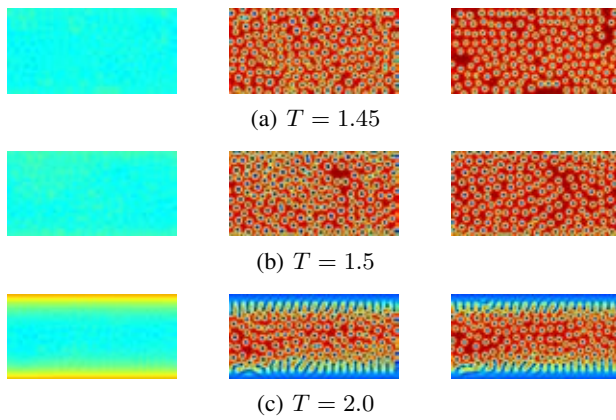
612

(a) $T = 1.45$

(b) $T = 1.5$

(c) $T = 2.0$

Figure 5: Snapshots of the temperature and chemical order parameter fields for a double GSRD system with three different values for the boundary and initial temperatures. One species has an exothermic reaction ($\Delta H_1 = -20 \times 10^{-3}$, centre column), and the other has an endothermic reaction ($\Delta H_2 = 25 \times 10^{-3}$, right column).

Figure 6: Population dynamics of the spot patterns within a two-species thermal GSRD system. The number of spots of the two types are shown as N_{exo} (red curve) and N_{endo} (blue curve). Also shown is the imposed boundary temperature T_B (cyan curve), and the mean bulk temperature, T_{in} (green curve). Note that this is the mean temperature of the inner region of the domain, for which $0.2H \leq z \leq 0.8H$.

thermally neutral system (note also that when one species was exothermic or one was endothermic, there was essentially no viable range over which spots stably formed).

Having established the temperature window within which the symbiotic system can persist, we can also consider the influence of initial conditions. Perhaps it is possible that once established, a stable population could endure thermal perturbations beyond the range of static temperatures that we have already observed.

Resilience

In this section we will carry out stress tests in which a stable population of spots are subjected to temperature variations that go beyond their viable range. Note that with initial **and** boundary temperatures outside of the range $1.45 < T < 2.5$, the spots are either destroyed by themselves or the phase of pattern transforms and the spots are mostly diminished. While this range is much greater than the small range of the neutral patterns, there are circumstances under which it can be extended yet further.

Here we will initialise the system at $T_0 = 1.5$ and then vary the temperature in linear ramps over the range $0.8 \leq T \leq 2.5$. The results of this experiment are illustrated in Figure 6 and can also be viewed as an animation (Bartlett, 2016).

The most striking feature of this graph is how small the variations in the internal temperature are compared to the variations of the boundary temperatures. The bulk of the system stays within the range $1.4 \leq T_{in} \leq 1.5$, despite the fact that the boundary temperatures range between $0.8 \leq T_B \leq 2.5$. It seems that the two sets of interacting chemical patterns are able to regulate their local temperature such that
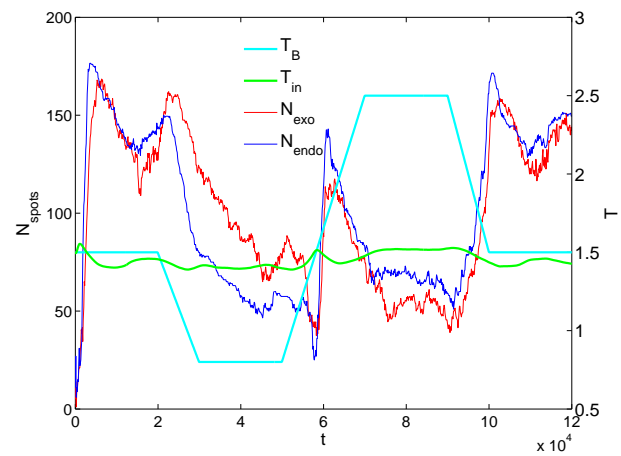
it remains close to a suitable value for their own persistence.

From Figure 6, we can start to assess the mechanism behind this thermal homeostasis. When the temperature is lower than ideal, the exothermic spots tend to outnumber their endothermic counterparts. It makes sense that colder conditions make it harder for heat absorbing structures to survive but easier for those that are heat emitting.

Conversely, when the temperature is raised high enough to destroy the structures under normal conditions, we see that the endothermic spots instead have a greater population than those that are exothermic. In this case the damping effect of the endothermic spots is reduced by the warmer boundary temperature and hence they are able to increase in number, so much so that they outnumber the exothermic spots and contribute to the maintenance of the bulk temperature close to $T_{in} \approx 1.5$.

Note that even with ideal initial temperatures, the spot patterns cannot persist when the boundary temperatures are held below $T_B = 0.8$. Likewise above $T_B = 2.5$, the system begins to bifurcate to a different GSRD phase, and there are few spot structures left. Hence the viable range of the symbiotic system was $\Delta T_{sym} \approx 1.7$, when the initial temperature was close to the ideal value of 1.5.

What we have seen here is a prime example of precariousness (Virgo, 2011; Froese et al., 2013). These dissipative structures do not persist if the initial temperature is lower than ~ 1.45, but if a population is established, it can then stabilise its environment even in the face of deadly external changes.

Conclusions

In this paper we have demonstrated how a regulatory mechanism, akin to homeostasis in living organisms, can emerge in a very simple non-living system. We carried out numerical simulations of systems with two GSRD systems. We found that when the rate constant takes on a standard kinetic dependence upon temperature, the emergent patterns in the system are readily destroyed when the temperature is perturbed away from the ideal pattern-forming value.

When enthalpy changes are introduced, making only one of the reactions either exo- or endothermic, the emergent structures are destroyed by strong positive feedback in the case of heat release, or damping in the case of heat uptake. Hence when the spot patterns have a singular influence on the temperature field, there is still no thermal resilience.

However, with a combined system in which one of the two reactions is exothermic and the other endothermic, we find that robust self-stabilisation occurs. Even when the boundary temperature was raised to $T_B = 2.5$, the combined action of the spot patterns maintained the internal temperature of the system very close to the ideal growth value of $T = 1.5$. Also in colder than optimal conditions, the two sets of structures were able to retain the ideal temperature in the midst of the system. Note that for the very low temperatures, $T < 0.8$, it was necessary to initialise the system at a higher temperature, such that a population of spots could first establish themselves, before surviving externally imposed temperature reductions. Overall the internal variation in temperature was only $\sim 10\%$ of the boundary temperature variation (see Figure 6).

We observed that the regulation arose purely from changes in the number of spot patterns of the two different types. In cool conditions, the exothermic spot population rose above that of the endothermic spots. In hot conditions the opposite occurred. It is clear that with a simple set of feedbacks between a control variable and the strength of the objects carrying out the control, robust regulation of that variable can emerge spontaneously.

There are many parallels between the dynamics of our system and the characteristic Daisyworld family of models (Harvey, 2015; Watson and Lovelock, 2011; Wood et al., 2008). In both cases, there exist 'active' components of a system which somehow influence external forcings or parameters that have a fundamental impact on their persistence. In the case of Daisyworld, black daisies tolerate cold temperatures well because they have a warming effect (due to their lower albedo) and white daisies persist at high temperatures because they cause cooling. When combined together on Planet Daisyworld (described through a set of coupled, non-linear differential equations), the combined system appears to be able to regulate its temperature in the face of changes in external radiative forcing, just like our combined GSRD system.

However, there are several key differences between Daisyworld and our GSRD world. The two species of Daisyworld do not destroy themselves when placed in isolation. In fact there is some residual stabilisation effects even when only one species of daisy is present (Watson and Lovelock, 2011). So in Daisyworld, one species confers some regulation, and two species confers much more. In the model presented in this paper, one species with thermal effects destroys structure, whereas two species with thermal effects permits strong stabilisation.

The common feature of both models is of course that two rein controls within a system allows a strong degree of negative feedback around a certain point in phase space. The original forms of our model system were not created to demonstrate the spontaneous emergence of homeostasis, as Daisyworld was. Furthermore, the parameter range over which daisies can survive is explicitly linked to the width of the (prescribed) growth rate-temperature curve. In our model, the viability range of the patterns is not prescribed, but emergent.

Observations of our system may have implications for ways in which primordial life might have been able to start to influence its own survival chances by acquiring simple feedback mechanisms with its environment. Perhaps transitions such as the prokaryote to eukaryote transition were heavily influenced by the additional feedback or control conferred on a larger organism when a smaller organism infected it. Perhaps many of the characteristic homeostasis mechanisms that we see in extant life started off as simple push-pull feedback combinations such as those illustrated in this work.

Further Work

In this work we have explored an interesting class of dynamics in emergent pattern formation. There remain many interesting extensions to this work that could further reveal new phenomena. The reaction enthalpies of the two spot systems could be further increased. In fact it would be desirable to explore the scaling of thermal resilience as a function of the reaction enthalpy magnitudes. It could be that with extremely (thermally) strong reactions, small fluctuations could favour one or the other spot species, yielding a collapse of the system (through e.g. temperature divergence from dominance of the exothermic species).

In Section 7.4 of Bartlett (2014) (wherein the reaction enthalpies were lower in magnitude than those used in the current work), it was observed that pairs of thermally interacting spots (one from each species) locked together spatially. It would be useful to establish the point at which this becomes unnecessary (i.e. at what level of reaction enthalpy) for the survival of the two-spot species system. Furthermore, one could vary the diffusion constants of the two species such that, e.g. several spots of one species were embedded within the envelope of one spot of the other species. Perhaps several small exothermic spots with the correct enthalpy value can stabilise one much larger endothermic spot.

Such a dynamic takes inspiration from ideas concerning the Prokaryote-Eukaryote evolutionary transition.

References

Ayodele, S., Raabe, D., and Varnik, F. (2013). Lattice boltzmann modeling of advection-diffusion-reaction equations: Pattern formation under uniform differential advection. *Communications in Computational Physics*, 13:741–756.

Ayodele, S. G., Varnik, F., and Raabe, D. (2011). Lattice boltzmann study of pattern formation in reaction-diffusion systems. *Phys. Rev. E*, 83:016702.

Bartlett, S. (2014). *Why is life? An assessment of the thermodynamic properties of dissipative, pattern-forming systems*. PhD thesis, University of Southampton.

Bartlett, S. (2016). Symbiotic gray-scott reaction diffusion simulation. https://youtu.be/_fguKE9Xxio.

Bartlett, S. and Bullock, S. (2015). Emergence of competition between different dissipative structures for the same free energy source. In *Proceedings of the European Conference on Artificial Life 2015*, pages 415–422. MIT Press, Cambridge, MA.

Bartlett, S., Bullock, S., and Attard, G. (2010). Challenging the robustness of simulated protocells (abstract). *Artificial Life XII: Twelfth International Conference on the Synthesis and Simulation of Living Systems, Odense, Denmark*.

Chen, S. and Doolen, G. D. (1998). Lattice boltzmann method for fluid flows. *Annual Review of Fluid Mechanics*, 30(1):329–364.

Di Paolo, E. A., Noble, J., and Bullock, S. (2000). Simulation models as opaque thought experiments. In Bedau, M. A., McCaskill, J. S., Packard, N. H., and Rasmussen, S., editors, *Artificial Life VII*, pages 497–506. MIT Press, Cambridge, MA.

Froese, T., Virgo, N., and Ikegami, T. (2013). Motility at the origin of life: Its characterization and a model. *Artificial Life*, 20(1):55–76.

Frouzakis, C. (2011). Lattice boltzmann methods for reactive and other flows. In Echekki, T. and Mastorakos, E., editors, *Turbulent Combustion Modeling*, volume 95 of *Fluid Mechanics and Its Applications*, pages 461–486. Springer Netherlands.

Gray, P. and Scott, S. (1994). *Chemical Oscillations and Instabilities: Non-linear Chemical Kinetics*. International Series of Monographs on Chemistry. Clarendon Press.

Gray, P. and Scott, S. K. (1985). Sustained oscillations and other exotic patterns of behavior in isothermal reactions. *The Journal of Physical Chemistry*, 89(1):22–32.

Harvey, I. (2015). The circular logic of gaia : Fragility and fallacies , regulation and proofs. In *Proceedings of the European Conference on Artificial Life 2015*, pages 90–97. MIT Press, Cambridge, MA.

He, X., Chen, S., and Doolen, G. D. (1998). A novel thermal model for the lattice boltzmann method in incompressible limit. *Journal of Computational Physics*, 146(1):282 – 300.

Lee, K.-J., McCormick, W. D., Pearson, J. E., and Swinney, H. L. (1994). Experimental observation of self-replicating spots in a reaction-diffusion system. *Nature*, 369(6477):215–218.

Mahara, H., Suzuki, K., Jahan, R. A., and Yamaguchi, T. (2008). Coexisting stable patterns in a reaction-diffusion system with reversible gray-scott dynamics. *Phys. Rev. E*, 78:066210.

Pearson, J. E. (1993). Complex patterns in a simple system. *Science*, 261(5118):189–192.

Peng, Y., Shu, C., and Chew, Y. T. (2003). Simplified thermal lattice boltzmann model for incompressible thermal flows. *Phys. Rev. E*, 68:026701.

Shan, X. (1997). Simulation of rayleigh-bénard convection using a lattice boltzmann method. *Phys. Rev. E*, 55:2780–2788.

Virgo, N. D. (2011). *Thermodynamics and the structure of living systems*. PhD thesis, University of Sussex.

Watson, A. and Lovelock, J. (2011). Biological homeostasis of the global environment: the parable of daisyworld. *Tellus B*, 35(4).

Wolf-Gladrow, D. (2000). *Lattice-Gas Cellular Automata and Lattice Boltzmann Models: An Introduction*. Number no. 1725 in Lattice-gas Cellular Automata and Lattice Boltzmann Models: An Introduction. Springer.

Wood, A. J., Ackland, G. J., Dyke, J. G., Williams, H. T. P., and Lenton, T. M. (2008). Daisyworld: A review. *Reviews of Geophysics*, 46(1).

A Self-Replicating System of Ribosome and Replisome Factories

Lance R. Williams[1]

[1]Department of Computer Science, University of New Mexico, Albuquerque, NM 87131
williams@cs.unm.edu

Abstract

An artificial chemistry with composition devices borrowed from object-oriented and functional programming languages was introduced in prior work. *Actors* in *object-oriented combinator chemistry* are embedded in space and subject to diffusion; since they are neither created nor destroyed, mass is conserved. This paper further develops these ideas and applies them in significant ways. First, it introduces the concept of a self-replicating system's *normalized complexity*. Normalized complexity permits comparisons between artificial organisms defined in different virtual worlds by explicitly accounting for the relative complexities of both organism and world. Second, object-oriented combinator chemistry is used to define a parallel, asynchronous, spatially distributed self-replicating system modeled in part on the living cell. This system is strongly constructive since interactions among its parts results in the construction of more of these same parts; constructed parts are assembled from elements of a few primitive types. The system's high normalized complexity is contrasted with that of a simple composome, which is also defined.

Introduction

Much as Turing (1936) had done when motivating his abstract computing machine by comparing it to a human 'computer' executing programs with paper and pencil, von Neumann (1966) began his study of *self-replication in the abstract*, by thinking about a *concrete* physical machine. As imagined, von Neumann's *kinematic automaton* assembled copies of itself from a supply of components undergoing random motion on the surface of a lake. The components consisted of girders, sensors, effectors, logic gates and delays, together with tools for welding and cutting. It is unlikely that von Neumann ever intended to actually *build* a physical self-replicating machine. More likely, he regarded the kinematic automaton as a thought experiment, and abandoned it when he understood how the problems of self-reference, control and construction that truly interested him could be rigorously formulated in the abstract domain of *cellular automata (CA)*.

By abandoning his kinematic automaton, von Neumann became the first 'player' of a sometimes abstruse 'game' that many others have played since (Sipper, 1998). This 'game' has two parts and two pitfalls. Roughly speaking, the parts are: *define a model of computation*; and *define a self-replicating object (or system) in the model.* The two pitfalls, which must be avoided if the 'game' is to be non-trivial are: *making the computational model too abstract*; and *making the primitives too complex.* For example, it is trivial for a self-replicating object defined as a 1 to 'replicate' in an array of 0s if physics is defined to be a Boolean 'or' operation in neighborhoods. It is equally trivial for a self-replicator comprised of a robotic arm, a camera, and a computer to make copies of itself given a supply of robotic arms, cameras and computers. Von Neumann himself was very conscious of the parts and pitfalls and discusses the tradeoffs the 'game' presents at length.[1] His ingenious solution was (characteristically) a saddle point, combining a fiendishly simple model of computation and an enormously complex self-replicating object.

Nearly sixty years after von Neumann's death, no one has yet constructed a kinematic automaton of the kind he imagined. However, because the 'game' seems to many of us to still afford the best prospect by which to address the twin problems of the origin of life on Earth and its evolution into forms of increasing complexity, there is no shortage of new 'players.' Fortunately, current 'players' are the beneficiaries of a wealth of biological science unknown to von Neumann (Watson and Crick, 1953), of significant advances in the science of computing that von Neumann and Turing played seminal roles in founding, and of a growing body of work in the interdisciplinary fields of artificial life and complex systems.

Unsurprisingly (given the preceding), this paper contains descriptions of both a new model of computation and of a self-replicating system defined using that model. The design of *our* model is strongly influenced by the belief that something important was lost when von Neumann adopted cellular automata as *his* model. More specifically, we believe that *conservation of mass*, a law which all machines that assemble copies of themselves from parts must obey, was (in effect) the "baby thrown out with the bath water."

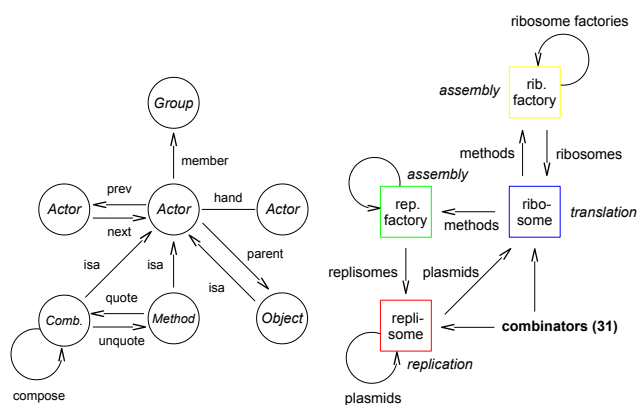[1]See von Neumann (1966) p. 76-77 and Arbib (1966) p. 179.

Figure 1: *Actor* datatype in *object-oriented combinator chemistry (OOCC)* (left). Self-replicating system of ribosome and replisome factories (right).

Our starting point is *artificial chemistry* (Dittrich et al., 2001), the study of the population dynamics of systems of constructible objects, which Fontana and Buss (1996) called *constructive dynamical systems*. Like them, we looked to the field of computer science for inspiration, hoping to re-purpose elements of modern *object-oriented* and *functional* programming languages as primitives and composition devices in an artificial chemistry. From object-oriented programming we borrowed the ideas of object composition and the association of programs with the data they operate on; from functional programming, we borrowed the idea of construction of programs by composition of 'program fragments' or *combinators.*

The first pitfall (excess abstraction) is avoided using a twofold strategy. First, we make our artificial chemistry concrete by embedding its constructed objects in space and relying solely on diffusion for dynamics. Significantly, to make this tractable, aggregates are treated as *masses* (unlike Arbib (1966) who treated aggregates as *areas*), and mass is conserved, which makes it a more plausible host for a kinematic automaton of the sort imagined by von Neumann. Second, as a guarantee of a different kind of realism, we insist that our artificial chemistry must be a *bespoke physics interface* as defined by Ackley (2013). More specifically, it must function as an abstract interface to a physically realizable *indefinitely scalable* computational substrate. This is consistent with the notion that kinematic automata defined using such interfaces, and which replicate by assembly of conserved parts, are tantamount to physical machines.

We avoid the second pitfall (complex primitives) by using (admittedly) complex primitives to build a self-replicating system comprised of parts that are *still more complex*. However, these more complex parts are constructed by the system itself! It follows that the system is *strongly constructive* in the sense that interactions among its parts result in the construction of more of these same parts (Dittrich et al., 2001).

Our inspiration, the ribosome, allowed us to imagine programs as enzymes and to define a pair of representations for programs, one spatially distributed and inert, the other compact and metabolically active. The self-replicating system that resulted is a parallel, asynchronous, distributed computation modeled in part on the living cell. See Figure 1.

The model of computation described in this paper has features in common with prior work on automata and artificial chemistry. The idea of movable aggregates of complex automata has precedent in Arbib (1966). The idea that kinematic automata can be built in embedded artificial chemistries has precedent in Laing (1977), Smith et al. (2003) and Hutton (2004). The use of nested multisets for object composition in an artificial chemistry has precedent in Paun (1998). The use of sequences of combinators as molecules and conservation of mass has precedent in di Fenizio (2000). The idea of molecules as programs has precedent in Laing (1977), Fontana and Buss (1996), di Fenizio (2000) and Hickinbotham et al. (2011). Finally, Taylor (2001) has argued that embeddedness and competition for matter, energy and space are necessary in artificial life systems capable of open-ended evolution.

Normalized Complexity

Pattee (1995) has described the simulation of an organism in a virtual world as an initial value problem where organisms are contingent states subject to the non-contingent laws of physics. In the 'game' of inventing both, there is a tradeoff between the *non-contingent complexity* of models of physical law, and the purely *contingent complexity* of artificial organisms defined inside those models. If physical law is too powerful, self-replication becomes trivial; life is too easy. Conversely, if physical law is not powerful enough, self-replication becomes impossible; life is too hard. It's possible that the most interesting games, those resulting in a bootstrapping process that culminates in organisms capable of open-ended evolution, are grounded in physical law "just powerful enough." Von Neumann's universal replicator R_V and its cellular automata virtual world CA_V suggest that interesting solutions to the 'game' are saddle points, maximizing the ratio of contingent $K(R_V \mid CA_V)$ and non-contingent complexity $K(CA_V)$:

$$\frac{K(R_V \mid CA_V)}{K(CA_V)}$$

where K is Kolmogorov complexity (Kolmogorov, 1965).[2] To explore this idea, let's consider a hierarchy of computational models; each model is built using an interface exposed

---

[2]Kolmogorov complexity is correct only if the replicator contains no *untranslated information*. Indeed, a pure template replicator, *e.g.,* Smith et al. (2003), might have very high Kolmogorov complexity yet its normalized complexity is zero since it is composed entirely of information that (apart from being copied) is never used.

by a more fundamental model. For example, the problem of simulating a CA with a more fundamental CA is described by Smith (1971). Among many other things, he showed that any CA with a Moore neighborhood (8 neighbors) can be simulated by a CA with a von Neumann neighborhood (4 neighbors) with an increase in space and a slowdown in time by constant factors that depend only on the numbers of states in the CA being simulated:

$$CA_8(\mathbb{Z}^2) \leq_1 CA_4(\mathbb{Z}^2)$$

where CA_8 and CA_4 are CAs with Moore and von Neumann neighborhoods, $\mathbb{Z}^2$ is the integer lattice and $(\leq_1)$ is Smith's $O(1)$ reduction.

Asynchronous cellular automata (ACA) are much like cellular automata except that local state is updated asynchronously (Nehaniv, 2004). It is possible to demonstrate by construction that any CA can be simulated by an ACA with an increase in space (Nakamura, 1974; Nehaniv, 2004) and a slowdown in time (Berman and Simon, 1988) by constant factors that depend only on the number of states and neighborhood size of the CA. Consequently,

$$CA(\mathbb{Z}^2) \leq_1 ACA(\mathbb{Z}^2)$$

where $(\leq_1)$ is Nakamura's $O(1)$ reduction. Our approach is premised on the idea that models in the *object-oriented combinator chemistry (OOCC)* defined in this paper can be compiled to ACAs of one higher dimension. Objects in the artificial chemistry are instances of a recursive datatype grounded in a small number of primitive types and closed under two forms of composition. The extra dimension is used to represent the internal structure of composed objects and the size of this representation is defined as an object's *mass*:

$$OOCC(\mathbb{Z}^2) \leq_1 ACA(\mathbb{Z}^2 \times \mathbb{N})$$

where $(\leq_1)$ is the hypothesized compilation process. Unlike Arbib (1966) who assumed that arbitrarily large automata aggregates could be moved $O(1)$ distance in $O(1)$ time, we assume only that objects of mass m can be moved $O(1)$ distance in $O(m)$ time.

While the significance of our work does not depend on it, the hypothesized compilation process is intriguing because ACAs of dimension three or less can (in principle) be physically realized in hardware. Furthermore, this can be done such that the abstract dimensions of space and time in the ACA (and of all models that have been $O(1)$ reduced to it) are coextensive with physical dimensions of space and time:

$$ACA(\mathbb{Z}^3) \leq_1 U$$

where U is the physical universe. This is the basis for the claim that our artificial chemistry is a bespoke physics interface as defined by Ackley (2013) and that kinematic automata built with it are tantamount to physical machines.

Given a replicator R defined on top of a hierarchy of models reducible to U by $O(1)$ reduction $R \leq_1 M_N \leq_1 \cdots \leq_1 M_1 \leq_1 U$, the ratio of the contingent and non-contingent complexities of replicator R becomes

$$\frac{K(R \mid M_N)}{K(M_N \mid M_{N-1}) + \cdots + K(M_2 \mid M_1) + K(M_1)}.$$

The meaning of this quantity, which will henceforward be termed a replicator's *normalized complexity*, is best illustrated by an example. Codd (1968) was able to significantly simplify von Neumann's replicator and its host CA. Although the contingent complexity of the Codd replicator is significantly less than that of the von Neumann replicator, we speculate that (were they calculated) their normalized complexities would be closer in value.

Langton (1984) defined a much simpler 'loop' replicator L_L on top of the Codd CA substrate. Its contingent complexity, $K(L_L \mid CA_C)$, is much less than that of the Codd replicator, $K(R_C \mid CA_C)$. Nehaniv (2004) showed how the Codd CA substrate could be $O(1)$ reduced to an ACA and demonstrated the Langton 'loop' running on top of the Codd CA running on top of this ACA. These results allow us to compare the normalized complexities of the Codd replicator and the Langton 'loop' defined with respect to the same hierarchy of computational models:

$$\frac{K(L_L \mid CA_C)}{K(CA_C \mid ACA_N) + K(ACA_N)} < \frac{K(R_C \mid CA_C)}{K(CA_C \mid ACA_N) + K(ACA_N)}.$$

Hutton's work on "artificial cells" provides a second example (Hutton, 2004). In Hutton's virtual world, physical law takes the form of an artificial chemistry defined by a set of 34 graph rewrite rules. Hutton's artificial organism is a cell-like configuration of atoms $C_H + P_1$ containing a small (non-functional) information payload P_1. Significantly, Hutton demonstrated that both the 'cell' and its payload are replicated by the 'reaction' rules of the artificial chemistry. Because the payload P_1 is untranslated, the contingent complexity of Hutton's cell is $K(C_H \mid AC_{34})$.

Hutton subsequently extended AC_{34} by adding six rules for translating the payload P_1 into an 'enzyme' E_1 capable of 'catalyzing' an arbitrary reaction and used this enzyme to replace one of the graph rewrite rules, R_1. In doing so, the (non-functional) information payload becomes a (functional) partial genome and some part of the system's complexity moves from the non-contingent to the contingent category. However, this exchange is insufficient to offset the increase in non-contingent complexity that results from the addition of the six rules. Consequently,

$$\frac{K(E_1 \mid P_1, AC_{40} - R_1) + K(C_H + P_1 \mid AC_{40} - R_1)}{K(AC_{40} - R_1)} < \frac{K(C_H \mid AC_{34})}{K(AC_{34})}.$$

where $K(E_1 \mid P_1, AC_{40} - R_1)$ is zero since P_1 encodes E_1 using a process defined by the artificial chemistry $AC_{40} - R_1$.

Object-Oriented Combinator Chemistry

There are three types of actors: *objects*, *methods* and *combinators*. Objects and methods are like objects and methods in object-oriented programming. More specifically, objects are containers for actors, methods are programs that govern actors' behaviors, and combinators are the building blocks used to construct methods (see Figure 1). Like amino acids, which can be composed to form polypeptides, *primitive* combinators can be composed to form *composite* combinators. A method is just a composite combinator that has been repackaged or *unquoted*. Prior to unquoting, combinators do not manifest behaviors, so unquoting might correspond (in this analogy) to the folding of a polypeptide chain into a protein.

Objects are *multisets* of actors. They are of four immutable types constructed using $\{\ \}_1$, $\{\ \}_2$, $\{\ \}_3$ and $\{\ \}_4$. For example, $\{x, y, z\}_2$ is an object of type two that contains three actors, x, y and z. Combinators are composed with $(>\!=\!>)$ and quoted and unquoted using $(\ )^-$ and $(\ )^+$. Primitive combinators and empty objects have unit *mass*. The mass of a composite combinator is the sum of the masses of the combinators of which it is composed. The mass of an object is the sum of its own mass and the masses of the actors it contains. Since actors can neither be created nor destroyed, mass is conserved.

Actors are *reified* by assigning them positions in a 2D virtual world. Computations progress when actors interact with other actors in their Moore neighborhoods by running methods. Methods are sequences of combinators compiled from programs defined in a visual programming language. The programs in this language, *dataflow graphs*, serve as abstract specifications of actors' behaviors. Neither the visual programming language nor the combinator language are described in this paper since both were described at length in Williams (2015).

All actors are subject to *diffusion*. An actor's diffusion constant decreases inversely with its mass. This reflects the real cost of data transport in the (notional) $ACA(\mathbb{Z}^2 \times \mathbb{N})$ substrate. Multiple actors can reside at a single site, but diffusion never moves an actor to an adjacent occupied site if there is an adjacent empty site.

The object that contains an actor (with no intervening objects) is termed the actor's *parent*. An actor with no parent is a *root*. Root actors (or actors that have the same parent) can associate with one another by means of *groups* and *bonds*. Association is useful because it allows working sets of actors to be constructed and the elements of these working sets to be addressed in different ways.

The first way in which actors can associate is as members of a *group*. All actors belong to exactly one group and this group can contain a single actor. For this reason, the group relation is an *equivalence relation* on the set of actors. A group of root actors is said to be *embedded*. All of the actors in an embedded group diffuse as a unit and all

methods run by actors in an embedded group (or contained inside such actors) share a finite time resource in a zero sum fashion. Complex computations formulated in terms of large numbers of actors running methods inside a single object or group will therefore be correspondingly slow. Furthermore, because of its large net mass, the object or group that contains them will also be correspondingly immobile.

The second way in which actors can associate is by *bonding*. Bonds are short relative addresses that are automatically updated as the actors they link undergo diffusion. Because bonds are short (L_1 distance less than or equal to two), they restrict the diffusion of the actors that possess them. Undirected bonds are defined by the *hand* relation H, which is a *symmetric relation* on the set of actors, *i.e.,* $H(x,y) = H(y,x)$. Directed bonds are defined by the *previous* and *next* relations, P and N, which are *inverse relations* on the set of actors, *i.e.,* $P(x,y) = N(y,x)$. An actor can possess at most one bond of each type.

Apart from composition, containment, groups and bonds there is no other mutable persistent state associated with actors. In particular, there are no integer registers. Primitive combinators exist for addressing individual actors or sets of actors using most of these relations. Other primitive combinators modify actors' persistent states.

Composomes

Composomes are quasi-stationary molecular assemblies that preserve *compositional information* (Segré et al., 2000). As self-replicating entities, they possess very low normalized complexity because they do not construct the parts of which they are comprised, and individually, these parts are more complex than the composome itself. Nevertheless, a composome serves as a good first example, and we can construct one by defining a set of behaviors using dataflow graphs and reifying them as an embedded group of methods:

$$X = \{\text{cmpA, cmpB, cmpC}\}$$

where cmpA, cmpB and cmpC are the three methods and $\{\ \}$ denotes an embedded group. The composome's first two methods run in the mother group (the group being copied) while its third runs in the daughter group (the copy). Because methods contained in different embedded groups run in parallel, they do not compete for cycles; this decreases the time required for self-replication.

- If cmpA is in a group with others but no members of its group have bonds then it finds an unbonded actor in its neighborhood similar to itself with no others in its group and creates a *next* bond with it.

- CmpB first verifies it is in the mother group. If it also has an unbonded neighbor similar to a member of its group and if the neighbor is not already in a group then it adds the neighbor to the daughter's group.
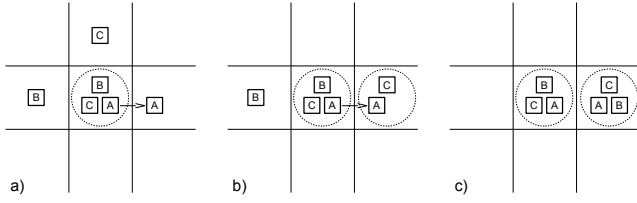
619

Figure 2: Self-replication by *composome*. a) CmpA forms a *next* bond with another cmpA instance in its neighborhood. b) CmpB finds a cmpC instance in its neighborhood and adds it to the daughter group. c) CmpC (in the daughter group) deletes the bond joining the mother and daughter groups after cmpB is added.

- CmpC first checks to see if it is in the daughter group. It does this by verifying that there is a group member with a *prev* bond (which can only be of type cmpA). It then verifies that there is also a group member not similar to either the cmpA instance or itself. By process of elimination, this group member must be of type cmpB. Since the daughter group contains the complete set of methods, it can delete the *prev* bond which joins the cmpA instances of the mother and daughter groups.

When placed into the virtual world with a supply of the methods that comprise it, the following reaction occurs

$$X + \text{cmpA} + \text{cmpB} + \text{cmpC} \rightarrow 2X.$$

Because cmpB does not check to see whether the composome already possesses a method before adding it to the daughter group, the fraction of reactants converted to complete composomes (self-replication efficiency) will be significantly less than 100%.

Ribosomes

Biological enzymes can be reified as chains of nucleotides or amino acids. The first can be read and copied but are spatially distributed and purely representational; the second are representationally opaque but compact and metabolically active. Dataflow graphs can be compiled into sequences of primitive combinators and reified in analogous ways: *genes* can be read and copied but do not manifest behaviors; *enzymes* manifest behaviors but cannot be read or copied. A *gene* is a spatially extended chain of actors of type combinator linked with *directed* bonds:

$$G_i = >_{j=1}^{|G_i|} c_i(j)$$

where $c_i(j)$ is combinator j of gene i and $(>)$ are directed bonds. As in the genomes of living cells, sets of genes that are expressed together can be grouped together. A *plasmid* is a sequence of genes joined with *undirected* bonds:

$$P = |_{i=1}^{|P|} >_{j=1}^{|G_i|} c_i(j)$$

where $(\,|\,)$ are undirected bonds. An additional undirected bond $c_{|P|}(|G_{|P|}|) \mid c_1(1)$ closes the chain. While plasmids are spatially distributed chains of multiple actors, *enzymes* are single actors of type method:

$$E_i = (>\!\!\Rightarrow_{j=1}^{|G_i|} c_i(j))^+$$

where $(>\!\!\Rightarrow)$ is Kleisli composition and $(\,)^+$ is the constructor for actors of type method. In addition to plasmids, comprised of genes, a minimum self-replicating system might contain objects of three types. *Ribosomes* translate genes into enzymes and *replisomes* copy plasmids. *Factories* are copiers of compositional information, namely, the sets of enzymes and objects that comprise ribosomes, replisomes and factories themselves. A self-replicating system like this would possess *semantic closure* (Pattee, 1995) because it would construct the parts that comprise it (enzymes) from descriptions contained within itself (genes). Unlike the cell (where enzymes are sequences of amino acids and genes are sequences of nucleotides) enzymes and genes are built from the same elementary building blocks, *i.e.,* combinators.

Biological ribosomes translate descriptions of proteins encoded as sequences of nucleotides into polypeptides, sequences of amino acids, the building blocks of proteins. A *computational ribosome* translates a plasmid into one or more enzymes by traversing genes while composing combinators from the neighborhood matching those comprising the gene. In functional pseudocode, the ribosome evaluates the following expression:

$$\text{map}_| \, ((\,)^+ \cdot (\text{fold}_> \, (>\!\!\Rightarrow))) \, P$$

where $map_|$ maps functions over the genes G_i that comprise plasmid P and $fold_>$ is right fold over the combinators $c_i(j)$ that comprise a gene. A computational ribosome can be constructed by defining four enzymes that perform these functions and placing them inside an actor of type object:

$$R = \{ \text{ribA, ribI, ribE, ribT} \}_0.$$

RibA first attaches R to the plasmid by adding it to the group of the initial combinator of some gene, $c_i(1)$. Afterwards, the (now unnecessary) ribA is expelled (and R becomes R'); see Figure 3 (a).

After ribosome attachment, ribI finds an actor in the neighborhood with type matching $c_i(1)$ and places it inside R'; see Figure 3 (b). When R' is at position j on the plasmid, ribE finds a neighbor with type matching $c_i(j+1)$ and composes it with the combinator contained in R', *i.e.,* with $c_i(1) >\!\!\Rightarrow \cdots >\!\!\Rightarrow c_i(j)$. It then advances the position of R' to $j+1$ by following the *next* bond; see Figure 3 (c). This process continues until R' reaches the last combinator in the gene, $c_i(|G_i|)$, which possesses a *hand* bond, at which point ribT promotes the combinator to a method, expels the method, and moves the ribosome across the bond.
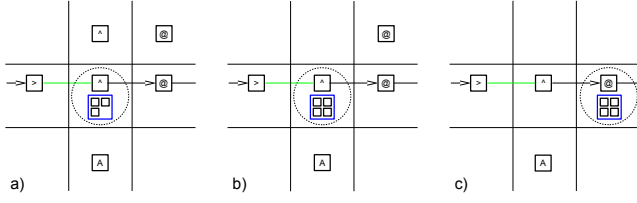
Figure 3: a) Ribosome attaches itself to plasmid at gene origin (marked by *hand* bond) and ejects ribA. b) Combinator from neighborhood matching initial combinator is placed inside ribosome. c) Combinator from neighborhood matching next combinator of plasmid is composed with combinator inside ribosome and ribosome advances.

If plasmid P and ribosome R are placed in the virtual world with a supply of primitive combinators $\sum_P \sum_C h(p,c)\, c$ then the ribosome manufactures the enzymes $\sum_P E_p$ described by the plasmid

$$P + R + \sum_P \sum_C h(p,c)\, c \rightarrow P + R' + \text{ribA} + \sum_P E_p$$

where C is the set of 42 primitive combinators and $h(p,c)$ is the number of combinators of type c in G_p and E_p, *i.e.,* the gene and enzyme reifications of behavior p.

Replisomes

We have already defined a computational ribosome, *i.e.,* an object that translates inert descriptions of behaviors encoded by a plasmid (genes) into behaviors reified as methods able to do actual work (enzymes). We now turn our attention to the problem of defining a *computational replisome*, an object that will replicate plasmids. In functional pseudocode, the replisome evaluates the following expression:

$$(\text{fold}_| \,(\,|\,) \cdot \text{map}_| \,(\text{fold}_> \,(>)))\, P$$

where $(\,|\,)$ and $(>)$ are functions that create undirected and directed bonds, $fold_|$ is right fold over the genes G_i that comprise plasmid P and $fold_>$ is right fold over the combinators $c_i(j)$ that comprise a gene.

Biological replisomes copy plasmids in pairs. Replication begins when two replisomes are assembled at the plasmid's *replication origin*. Each replisome manages one *replication fork*. The replication forks move away from the replication origin in opposite directions and replication is finished when the pair of replisomes reunite at a position on the plasmid opposite the origin. A computational replisome can be designed that works in a similar way. As in a cell, there are two replication forks. However, unlike a cell, only one moves; the other is stationary. The replisome manages the active replication fork. It is initially an object containing five enzymes and an empty object of the same type as itself:

$$Q = \{\text{repA, repE, repF, repY, repZ, } \{\,\}_2 \}_2.$$

RepA first causes the replisome to attach to the plasmid. It does this by adding the replisome to the group of one of
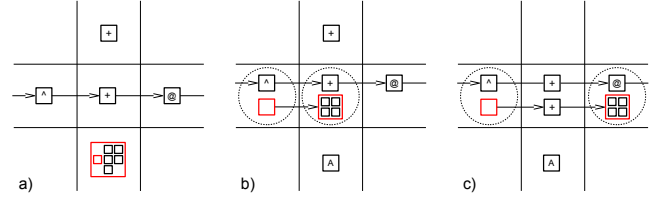
Figure 4: a) A short segment of a plasmid and a replisome containing five enzymes and an empty object *marker* of the same type as itself. b) Replisome attaches itself to the plasmid by joining the group of one of its combinators; attaches the marker to the combinator preceding its own attachment site; forms a directed bond with the marker; and ejects repA. c) Replisome advances along the plasmid after splicing a combinator of the correct type from the neighborhood into the directed bond that trails it. This is the first combinator of the daughter plasmid.

the plasmid's combinators. Afterwards, the stationary replication fork is marked by attaching the empty object $\{\,\}_2$ contained within the replisome to the combinator that precedes the replisome's own attachment site. RepA creates a directed bond from the marker to the replisome and then ejects itself since it is no longer needed; see Figure 4 (b).

RepE and repF govern the motion of the replication fork. RepE finds a combinator in the neighborhood matching the combinator attached to replisome Q'. It moves Q' in the increasing direction (by joining the group of the combinator that follows the replisome's own attachment site) and splices the neighbor into the growing chain (the incomplete *daughter* plasmid) that trails it; see Figure 4 (c). RepF is very similar except that it moves the replication fork through the *hand* bonds that mark the boundaries between genes.

Replication is complete when Q' encounters a marker, or more precisely, when it finds a marker attached to the combinator that follows its own attachment site. In the most common case, the replisome and marker are situated within a single gene. RepY recognizes this situation and creates the final *next* bond, completing the daughter plasmid. Very infrequently, the replisome and marker straddle a boundary between two genes. RepZ recognizes this situation and creates the final *hand* bond. In both cases, the replisome and marker are detached from the plasmid.

If plasmid P and replisome Q are placed in the virtual world with a supply of primitive combinators $\sum_P \sum_C h(p,c)\, c$ then the replisome copies the plasmid

$$P + Q + \sum_P \sum_C h(p,c)\, c \rightarrow 2\, P + Q' + \text{repA} + \{\,\}_2.$$

Self-Replicating Ribosome and Replisome Factories

Abstractly, factories are copiers of *compositional information*, which is heritable information distinct from the *genetic information* copied by replisomes and which ribosomes translate into enzymes. Concretely, factories are ob-
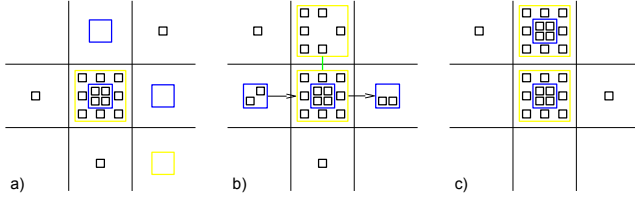
Figure 5: a) Self-replicating ribosome factory contains eight enzymes and a model ribosome. b) Directed bonds connect the factory to partially assembled ribosomes while an undirected bond connects it to a partially assembled daughter factory. c) One of the product ribosomes becomes the model for the daughter factory while the second is available to synthesize enzymes for the encompassing system.

jects containing a specific set of enzymes and a *model*, which can be either a ribosome or a replisome; see Figure 5 (a). A factory's enzymes can be grouped into four categories:

- FacP, facN and facH form *prev*, *next* and *hand* bonds with empty objects from the neighborhood. The order in which these three events occur is more or less random. The objects $\{\ \}_k$ bonded to the mother factory by *prev* and *next* bonds will become new instances of the model. The object $\{\ \}_{k+1}$ bonded to the mother factory by the *hand* bond will become the daughter factory; see Figure 5 (b).

- FacU moves enzymes with types matching contents of the model into the incomplete model instances. FacV moves enzymes with types matching contents of the mother factory into the incomplete daughter factory.

- FacX uses the generalized set difference operator to verify that a new model instance, *i.e.*, *product*, has all of the enzymes that the old model contains. If so, it marks the product as complete using a self-directed *hand* bond. FacY does the same thing for the daughter factory but uses a self-directed *prev* bond to indicate completeness.

- FacZ checks to see that both products have self-directed *hand* bonds and also that the daughter factory has a self-directed *prev* bond. If so, it 1) deletes the *prev* and *next* bonds connecting the mother factory and the products; 2) moves one of the completed products into the daughter factory (to serve as its model); and 3) deletes the *hand* bond connecting the mother and daughter factories; see Figure 5 (c).

Given the above enzymes, it is now possible to define a self-replicating ribosome factory:

$$F_R = \{facP, facN, facH, facU, facV, facX, facY, facZ, R\}_1.$$

When placed in the virtual world with a supply of empty objects and enzymes comprising ribosomes $\sum_R E_r$ and factories $\sum_F E_f$ the ribosome factory constructs a new ribosome

factory and a new ribosome:

$$F_R + 2\{\ \}_0 + \{\ \}_1 + 2\sum_R E_r + \sum_F E_f \rightarrow 2\,F_R + R.$$

A self-replicating replisome factory F_Q can be defined similarly. When placed in the virtual world with a supply of empty objects and enzymes comprising replisomes $\sum_Q E_q$ and factories $\sum_F E_f$ the replisome factory constructs a new replisome factory and a new replisome:

$$F_Q + 4\{\ \}_2 + \{\ \}_3 + 2\sum_Q E_q + \sum_F E_f \rightarrow 2\,F_Q + Q.$$

Note that the left hand side of the reaction contains four empty objects $\{\ \}_2$ instead of two; the extras are the markers contained by the new replisome and replisome model.

We now have all of the components needed to build a self-replicating system of ribosome and replisome factories. Unlike the composome, which copied itself solely by reflection, the self-replicating system is a *quine* that translates and replicates a self-description reified as a data structure within the virtual world itself:

$$P_{17} = ribA \mid ribI \mid ribE \mid ribT \mid repA \mid repE \mid repF \mid repY \mid repZ$$
$$\mid facP \mid facN \mid facH \mid facU \mid facV \mid facX \mid facY \mid facZ.$$

This *genome* consists of a single plasmid containing 586 combinators comprising 17 genes. The minimum *phenome* required for bootstrapping the self-replicating system consists of a replisome factory F_Q, a ribosome factory F_R and a ribosome R. When genome P_{17} and phenome $F_Q + F_R + R$ are placed in the virtual world with a supply of empty objects $\{\ \}_k$ and combinators $\sum_{P_{17}} \sum_C h(p,c)\,c$, the system increases the redundancy of all of its component parts:

$$P_{17} + F_Q + F_R + R + 2\{\ \}_0 + \{\ \}_1 + 4\{\ \}_2 + \{\ \}_3 + 3\sum_{P_{17}} \sum_C h(p,c)\,c$$
$$\rightarrow 2\,P_{17} + 2\,F_Q + 2\,F_R + Q' + repA + \{\ \}_2 + R + R' + ribA.$$

Note that there are three instances of $\sum_{P_{17}} \sum_C h(p,c)\,c$ on the left side of the equation. The ribosome R consumes the first two making two full circuits of the plasmid synthesizing the system's enzymes

$$P_{17} + R + 2\sum_{P_{17}} \sum_C h(p,c)\,c \rightarrow P_{17} + R' + ribA + 2\sum_{P_{17}} E_p$$

while the replisome Q (assembled by F_Q) uses the last copying the plasmid.

Comparison of Normalized Complexities

It is useful to compare the normalized complexities of the self-replicating system of ribosome and replisome factories and the composome defined earlier. Recall that the composome X is composed of 3 enzymes of 3 types: cmpA, cmpB and cmpC; these enzymes are in turn composed of 66 combinators of 17 types. Because the enzymes are defined outside the system, their complexity is non-contingent, and the composome's normalized complexity is quite low:

$$\frac{K(\{\text{cmpA, cmpB, cmpC}\} \mid \text{cmpA} + \text{cmpB} + \text{cmpC})}{K(\text{cmpA} + \text{cmpB} + \text{cmpC} \mid OOCC_{17}) + K(OOCC_{17} \mid ACA) + K(ACA)}$$

where $K(\text{cmpA} + \text{cmpB} + \text{cmpC} \mid OOCC_{17})$ is the portion of the composome's non-contingent complexity contained in its three enzymes.

In contrast, the self-replicating system of ribosome and replisome factories is composed of 17 different behaviors reified as both genes and enzymes; these genes and enzymes are in turn composed of $3 \times 586 = 1758$ combinators of 31 types. Furthermore, because the enzymes are defined within the system itself (by the genes), their complexity is (unlike that of the composome's enzymes) contingent. Consequently, the self-replicating system of ribosome and replisome factories possesses significantly higher normalized complexity than the composome:

$$\frac{K(F_Q + F_R + R \mid \sum_{P_{17}} E_p) + K(\sum_{P_{17}} E_p \mid P_{17}, R, OOCC_{31}) + K(P_{17} \mid OOCC_{31})}{K(OOCC_{31} \mid ACA) + K(ACA)}$$

where $K(F_Q + F_R + R \mid \sum_{P_{17}} E_p)$ and $K(P_{17} \mid OOCC_{31})$ are the compositional and genetically encoded portions of the self-replicating system's contingent complexity and $K(\sum_{P_{17}} E_p \mid P_{17}, R, OOCC_{31})$ is zero because the plasmid P_{17} encodes the enzymes $\sum_{P_{17}} E_p$ using a process defined by the ribosome R and the object-oriented combinator chemistry.

Conclusion

This paper introduced the concept of a self-replicating system's *normalized complexity*. Normalized complexity permits comparisons between artificial organisms defined in different virtual worlds by explicitly accounting for the relative complexities of both organism and world. An *object-oriented combinator chemistry* (introduced in prior work) was used to define a parallel, asynchronous, spatially distributed self-replicating system modeled in part on the living cell. The high normalized complexity of this self-replicating system of ribosome and replisome factories was contrasted with that of a simple composome.

References

Ackley, D. (2013). Bespoke physics for living technology. *Artificial Life*, 34:381–392.

Arbib, M. A. (1966). Simple self-reproducing universal automata. *Information and Control*, 9(2):177–189.

Berman, P. and Simon, J. (1988). Investigations of fault-tolerant networks of computers. In *STOC*, pages 66–77.

Codd, E. F. (1968). *Cellular automata*. Academic Press, London.

di Fenizio, P. S. (2000). A less abstract artificial chemistry. In *Proc. of the 7th Intl. Conf. on the Simulation and Synthesis of Living Systems (ALIFE)*, pages 49–53.

Dittrich, P., Ziegler, J. C., and Banzhaf, W. (2001). Artificial chemistries: a review. *Artificial life*, 7(3):225–275.

Fontana, W. and Buss, L. W. (1996). *The barrier of objects: From dynamical systems to bounded organizations*. International Institute for Applied Systems Analysis.

Hickinbotham, S., Clark, E., Stepney, S., Clarke, T., Nellis, A., Pay, M., and Young, P. (2011). Molecular microprograms. In *European Conference on Artificial Life (ECAL)*, pages 297–304.

Hutton, T. J. (2004). A functional self-reproducing cell in a two-dimensional artificial chemistry. In *Proc. of the 9th Intl. Conf. on the Simulation and Synthesis of Living Systems (ALIFE)*, pages 444–449.

Kolmogorov, A. (1965). Three approaches to the quantitative definition of information. *Problems of Information Transmission*, 1(1):1–7.

Laing, R. A. (1977). Automaton models of reproduction by self-inspection. *Journal of Theoretical Biology*, 66(1):437–456.

Langton, C. G. (1984). Self-reproduction in cellular automata. *Physica D: Nonlinear Phenomena*, 10(1):135–144.

Nakamura, K. (1974). Asynchronous cellular automata and their computational ability. *System Comput. Controls*, 15(5):56–66.

Nehaniv, C. L. (2004). Asynchronous automata networks can emulate any synchronous automata network. *IJAC*, 14(5-6):719–739.

Pattee, H. (1995). Evolving self-reference: Matter, symbols, and semantic closure. *Communication and Cognition - Artificial Intelligence*, 12:9–27.

Paun, G. (1998). Computing with membranes. *Journal of Computer and System Sciences*, 61:108–143.

Segré, D., Ben-Eli, D., and Lancet, D. (2000). Compositional genomes: Prebiotic information transfer in mutually catalytic noncovalent assemblies. *Proceedings of the National Academy of Sciences*, 97(8):4112–4117.

Sipper, M. (1998). Fifty years of research on self-replication: An overview. *Artificial Life*, 4(3):237–257.

Smith, A., Turney, P. D., and Ewaschuk, R. (2003). Self-replicating machines in continuous space with virtual physics. *Artificial Life*, 9(1):21–40.

Smith, A. R. (1971). Cellular automata complexity trade-offs. *Information and Control*, 18(5):466–482.

Taylor, T. (2001). Creativity in evolution: Individuals, interactions and environments. In Bentley, P. and Corne, D., editors, *Creative Evolutionary Systems*, pages 79–108. Morgan Kaufman.

Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. *J. of Math*, 58(345-363):5.

von Neumann, J. (1966). Theory of self-replicating automata. *Urbana: University of Illinois Press*.

Watson, J. D. and Crick, F. H. (1953). Molecular structure of nucleic acids. *Nature*, 171(4356):737–738.

Williams, L. R. (2015). Programs as polypeptides. In *European Conference on Artificial Life (ECAL)*, pages 150–157, York, England.

Living Technology

EvoBot: An Open-Source, Modular Liquid Handling Robot for Nurturing Microbial Fuel Cells

A. Faíña[1,3], F. Nejatimoharrami[1], K. Stoy[1], P. Theodosiou[2], B. Taylor[2] and I. Ieropoulos [2]

[1]Robots, Evolution, and Art Lab (REAL), IT University of Copenhagen, Denmark
[2]Bristol BioEnergy Centre, Bristol Robotics Laboratory, University of the West of England, Bristol, UK
[3] Corresponding author: anfv@itu.dk

Abstract

Liquid handling robots are rarely used in the domain of artificial life. In this field, transitory behaviours of non-equilibrium man-made systems are studied and need an automatic monitoring and logging of results. In addition, artificial life experiments are dynamic with frequent changes, which makes it difficult to apply conventional liquid handling robots as they are designed to automate a pre-defined task. In order to address these issues, we have developed an open source liquid handling robot, EvoBot. It uses a modular approach, which gives us the possibility to reconfigure the robot for different experiments and make it possible for users to add functionality by just developing a function specific module. In addition, it provides sensors and extra functionality for monitoring an experiment, which allows researchers to perform interactive experiments with the aim of prolonging non-equilibrium conditions. In this paper, we describe the modular design of EvoBot, document its performance, and provide a novel example of an interactive experiment in artificial life, where the robot nurtures a microbial fuel cell based on its voltage output.

Introduction

Liquid handling robots are often employed in chemical and biochemical laboratories in order to automate repetitive tasks (see (Kong et al., 2012) for a useful overview of liquid handling robots for lab automation). The benefit of this is typically an increase in reliability, throughput and precision when compared to manual liquid handling coupled with a reduction in labour cost. However, liquid handling robots are rarely used in the domain of artificial life.

There are two types of challenges that make existing liquid handling robots inappropriate for experiments in artificial life. The first relates to the functionality needed from the robot to be useful in artificial life research. In artificial life we are mostly interested in the initial condition of an experiment and how it develops but not typically in the end result, because the end result is equilibrium which in the context of artificial life corresponds to death. This means that not only should the robot prepare experiments, but also employ automatic monitoring and logging of results. Furthermore, it would be desirable if the liquid handling robot,

based on this monitoring, can interact with the experiment to extend its life-time. Solving this challenge has been the basis for the results of (Gutierrez, 2012; Gutierrez et al., 2014; Hanczyc et al., 2015; Müller et al., 2015), who all made an ad-hoc liquid handling robot for specific experiments in artificial chemical life.

The second type of challenge, which is the focus of our work, is the practical challenge of employing liquid handling robots in artificial life research. Artificial life research is often performed in small, government funded labs and as such the cost of acquisition is a limiting factor. Artificial life experiments are not static, but develop in response to the insights obtained from the experiments and this makes it difficult to apply conventional liquid handling robots as re-programming and reconfiguration is often tedious and time-consuming if at all possible for the end user. Furthermore, conventional liquid handling robots are often only setup to perform one specific task, thus making it difficult to modify with functionalities required for new types of experiments. If we could provide solutions to these challenges, artificial life research could also benefit from the advantages of automation namely increased reliability, throughput, precision and reduced cost. Turning these challenges around we are looking for liquid handling robot technology that has the following characteristics:

- Reconfigurable

- Versatile

- Low cost

- Extendable

In order to address these challenges, we propose the use of a modular design borrowing a significant number of advantages from modular robots(Yim et al., 2007). A modular design allows a non-expert user to reconfigure the robot for different experiments by swapping in and out modules. Expert knowledge is only required when designing modules with new functionalities. A modular approach also increases the versatility of liquid handling robots because they can easily be reconfigured for many different types of experiment
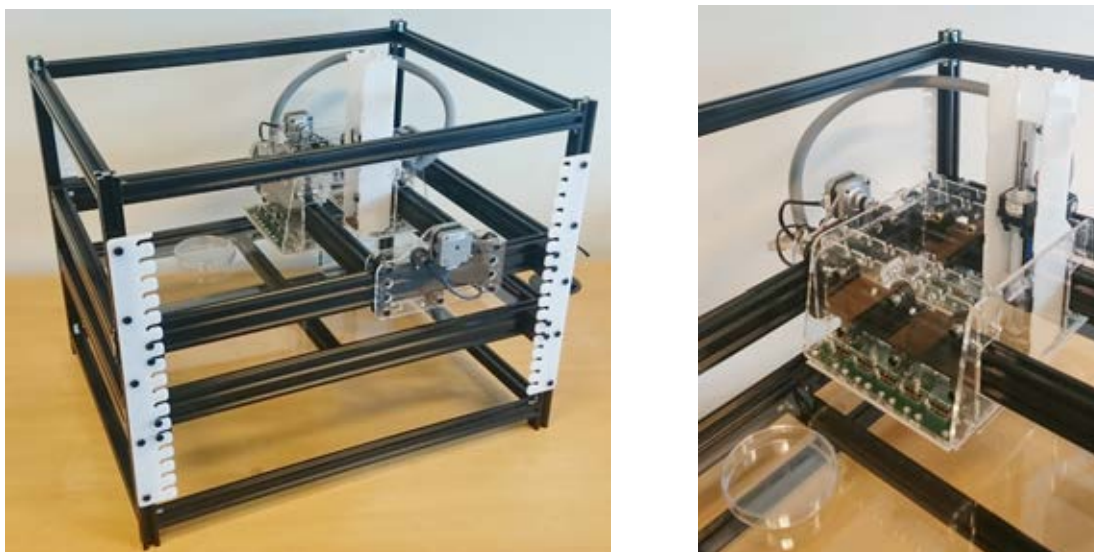
Figure 1: The EvoBot liquid handling robot (left) and a close-up of the movable head equipped with one syringe module.

during their life-time. We have made EvoBot open-source, which reduces the cost of acquisition and also allows researchers to build on our reliable platform and extend it with modules that have the necessary functionality for their specific experiment[1].

In this paper we present the modular design of the liquid handling robot, EvoBot, and carefully document its performance to make it possible for potential users to evaluate how appropriate it is for their experiment. We have already demonstrated its usefulness in artificial chemical life (Nejatimoharrami et al., 2016), but here we further evaluate its use in a new application domain, nurturing microbial fuel cells (MFCs). MFCs are devices where microbes convert organic matter directly into electricity. MFCs have demonstrated their utility as a basis for building robots with an artificial metabolism (Ieropoulos et al., 2010).

The conclusion is that EvoBot due to its versatility, extendability, and low cost was successfully implemented in the new application domain of nurturing microbial fuel cells with very limited modification. It is also demonstrated that using the robot for maintaining microbial fuel cells, compared to manually nurturing them, has numerous advantages, and thus in time may provide insights that can help researchers develop more efficient microbial fuel cells.

EvoBot Design Principles

As outlined above, a key goal of the EvoBot design was to develop a robotic platform, which can be configured for a

wide range of experiments without the involvement of an expert user. In order to achieve this, we modularised the design building on research in the field of modular robotics (Yim et al., 2007). A modular approach allows us to encapsulate complexity while providing a simple mechatronics plug'n'play interface to the system.

EvoBot (see Figure 1) consists of one structural frame and three types of layers, which in the default configuration are organised as follows: the top layer carries actuation, the middle layer is the experimental section, and the bottom layer is an observation platform. However, this default layout can easily be amended, e.g. several experimental layers can be introduced if a cascading experiment is under investigation. These three layers can easily be moved up and down in the frame. Functionality is provided in the form of modules, which allow functionality to be added incrementally in the form of new modules.

In order to create a high quality and cheap platform, we built the robot from off-the-shelf components and, where possible, used components used by the open-source 3D printer community and therefore readily available. For instance, we used Nema17 motors for actuation and Arduino based electronics for control of the robot. Another key principle was to favour laser-cut acrylics over 3D printing. The reasons being lower production time and high-quality of the produced components in comparison with the elements produced through 3D-printing. However, we did use 3D printed components outside the core mechanical structure, primarily inside the syringe module (described later) due to the geometrical flexibility afforded by 3D printers.

---

[1]Source code and design files can be found in Git repositories, https://bitbucket.org/afaina/evobliss-software and https://bitbucket.org/afaina/evobliss-hardware respectively.

EvoBot Implementation

In the following section, we will provide an overview of the hardware (mechanics & electronics), and software of EvoBot.

Mechanics

First, we will describe the layers of EvoBot in more detail followed by the implementation of the modules.

Frame   The frame is made of aluminium profiles and it provides the physical support for the layers. They can be attached to the frame at specific heights every 20mm. Additionally, it allows levelling the robot with four adjustable feet. The frame measures 600x400x600mm but it can easily be extended.

Layers   The EvoBot platform is organised into three types of layer: actuation, experiment, and observation layers.

The actuation layer contains a head, which can move in the horizontal plane using two belt and pinion mechanisms and two stepper motors. Up to 11 modules can be mounted on this head to provide different functionalities. At this point, only an actuated syringe module and a heavy payload module have been implemented, but various actuators and sensor modules are envisaged, e.g. temperature sensor, pH sensor, gripper for manipulation of dishes, extruder for printing reaction vessels, etc.

The experimental layer is essentially just a frame with a glass plate where vessels can be organised as required by the specific experiment. There is a hole in one corner, where vessels can be moved and dropped for automatic disposal, and together with a Petri dish dispenser system under development, a large number of experiments can be done in sequence. However, we have found that for now it is enough to clean reaction vessels by going through three water wash cycles, where vessels repeatedly are filled with water and emptied using the syringe modules. This makes the dispenser functionality less critical for long term operation. However, for more sensitive experiments the vessels and also syringe tips may have to be replaced on a per experiment basis.

The observation layer is essentially the same as the actuation layer except that modules cannot interact physically with the experiments above, because they are shielded by the glass plate. This limits the useful modules for this layer to modules that do not directly manipulate the experiments. Currently, a static webcam is used in the observation layer to monitor the experiment and provide feedback to the robot. However, in the longer term thermal imaging, magnetic stirring, liquid effluent sampling or the like could be integrated in modules for the observation layer.

Syringe and Heavy Payload Module   The syringe module can be seen in Figure 2.a-b has two degrees of freedom. It has a linear stepper motor (a stepper motor with a lead screw
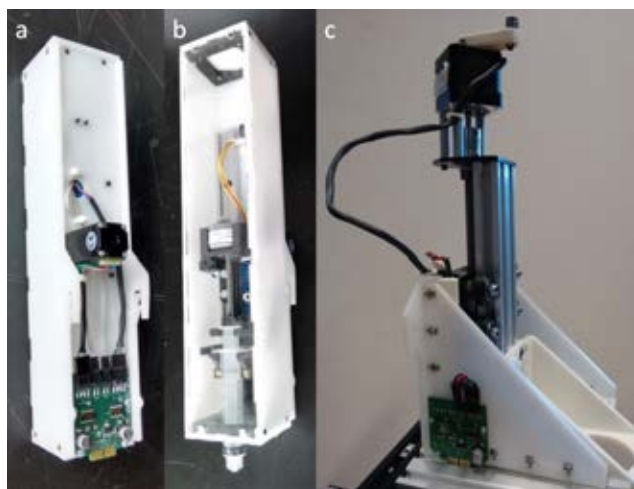

Figure 2: Syringe module (a, b) and heavy payload module (c).

and an internal nut) for moving the plunger up and down and a rack and pinion mechanism with another stepper motor for moving the syringe up and down. Syringes up to 20ml can be used and they can easily be replaced giving the user the opportunity to use the syringe that matches the experimental requirements.

The heavy payload module is designed to hold and move up and down big and heavy parts. A lead screw mechanism is used to hold the payload even if its motor is switched off. A Nema 17 stepper motor moves the payload but it can also be manually operated with a crank. Its stroke is 80mm and the maximum speed is 8mm/s. This module is shown in Figure 2.c and its end effector is designed to hold an OCT scanner, but it can easily be modified to hold other devices.

More modules are under development including a module with a gripper, and a module to measure pH. However, an EvoBot with one syringe is enough to perform useful experiments in artificial chemical life as demonstrated in (Hanczyc et al., 2015; Gutierrez et al., 2014).

Electronics

The core of the electronics of EvoBot is based on electronics used in the open-source 3D printer community. Specifically, we use the Arduino MEGA 2560 R3 with the shield RAMPS v1.4. This provides us with a mature and cheap electronics platform to build on, but perhaps more importantly, allows us to build on existing software for open-source 3D printers.

A key aspect in the electronics design was to keep the number of wires between components as low as possible to avoid interference with experiments, become a source of error if disconnected accidentally, and to give the robot a clean look. In order to do this, a circuit board was designed that routes power and communication to the modules on the head of the robot. This board is fairly large, but con-

sists only of simple routing, two input/output port expanders with serial interface (I2C) and spring connectors. For the modules, a custom board was also made that contains two stepper drivers with SPI communications (L6470) for moving up and down the stepper motors of the module, see Figure 2.a. When a module is placed on the head, the spring connector touches the pads of the board of the syringe and transmits power and the SPI signals. Furthermore, the wires between the Arduino and the head are reduced using the I2C expander ports as they generate the Chip Select signals for the SPI communication. Using this approach, only a power cable and an 8-way ribbon cable with the I2C and SPI buses are necessary to manage the 22 motors of the 11 syringe modules, which can be placed on the head.

Software

The goal of the software was to provide the end-user with a simple programming interface to the robot. The software has a host side and a robot side. The host side communicates with the robot side over a serial USB connection and the robot side software is a modified version of the Marlin firmware used in open-source 3D printers.

On the host side, we have chosen Python as the implementation language as this was the language with which our collaborators have most experience and also due to its simplicity. The software is divided into a graphical user-interface for manual control of the robot and a simple application programming interface.

Robot Manual Control The purpose of the manual control graphical user interface is simply to be able to test various functions of the robot without having to resort to programming. However, the most important use of the manual control program is to learn the position of interesting objects in the robots coordination frame. For example, the user could move the robot until it is approximately in the centre of a Petri dish of interest and lower the syringe so the tip is in the vessel. This is an empirical way of defining the geometry for an experiment, and perhaps the most practical for a range of experiments.

Application Programming Interface The application programming interface is also kept as simple as possible. The interface gives the programmer access to moving the robot head, moving the syringes and plungers, and inquiring about the positions of these elements.

The application programming interface is built on top of printcore, which is an open source Python library for interacting with 3D printers and pySerial, which handles the serial communication. The printcore library, however, has been heavily reduced and is likely to be unnecessary in the near future.

At the bottom of the software stack we have implemented a basic simulation that can be swapped instead of the serial communication library. This makes it possible to debug

Table 1: Max speed and accelerations of the robot.

	X	Y	Z (Syringe)	Plunger
Max Speed (mm/s)	180	180	235	8
Acceleration (mm/s^2)	3000	3000	235	4

the upper layers without actually moving the physical robot. This is a feature that has saved a significant amount of development time.

Marlin-Based Firmware On the robot side we run an extended version of Marlin, a firmware used to control open-source 3D printers. This gives us a mature basis for our robot controller. The head of the actuation layer is controlled directly using the functionality Marlin provides. For the syringe or heavy payload modules, we have extended Marlin with G-code commands for interacting with them. The use of Marlin is also beneficial as it is our plan to make an extruder module, which can 3D print reaction vessels and thus this aspect of the firmware can also be put to good use.

Precision and Performance

For the reader interested in understanding if EvoBot is suitable for their experiments and to facilitate comparison between liquid handling robots, we provide below a careful investigation of the precision and performance of EvoBot.

Speed

A key parameter to perform experiments is the speed of the robot because the experiments could take a long time if the speed of the robot is low or it is not able to manage a set of tests simultaneously. With this objective in mind, the maximum speed of the robot was tested for each axis. The test consisted of movements on each axis, increasing its acceleration and maximum speed until the robot lost some steps. In other words, the maximum speed is when the positioning of the robot after several movements is not accurate anymore. Table 1 contains the results of these tests and displays that they are significantly higher than the requested accuracy by our collaborators.

Positioning Performance

The positioning performance has been evaluated based on the ISO 9283:1998. Thus, one syringe module was modified to hold a test probe, instead of a syringe. This probe was used as an end effector to touch a dial indicator, which has an accuracy of 0.01mm. Given the short stroke of the indicator (25mm), we ran two different tests to calculate the accuracy and the repeatability of the robot on each axis.

Table 2: Positioning accuracy for each axis of the robot.

	0mm	1mm	2mm	10mm	20mm
X (mm)	0.04	0.03	-0.04	-0.11	-0.17
Y (mm)	0.01	-0.05	-0.08	-0.14	-0.18
Z (mm)	0.10	0.02	0.09	0.28	0.35

Table 3: Repeatability for each axis of the robot.

	X	Y	Z
RP (mm)	0.067	0.013	0.106

Accuracy is simply defined as $A = \bar{q} - q_t$, where q_t is the target position and $\bar{q}$ is the average of n measured coordinates. In this test, the robot was homed, and the dial indicator was placed in contact with the probe and set to zero. Then, the robot moved one axis sequentially to five different positions (1, 2, 10, 20 and 0 mm) and the real position of the probe was measured. This sequence was repeated 30 times and the accuracy for each point was calculated. The data is shown in Table 2, where we can observe that the positioning is quite reasonable for a low cost machine and there is no noticeable discrepancy. Furthermore, the slow increment of the inaccuracy when moving the robot to 10 and 20mm is mainly caused by a misalignment between the axis of the indicator and the axis of the robot. This is confirmed by moving the robot to longer distances where we do not detect any appreciable error. In the future, we will employ computer vision technology to calibrate the positioning accuracy measuring the three axes at the same time.

Regarding the repeatability or precision of the robot, we ran another experiment. Thus, five points were defined all along the workspace of the robot and the robot moved the probe sequentially from one point to the next one. At the last point, the end effector touched a dial indicator to measure its coordinates. This sequence was repeated 30 times for each axis and repeatability was calculated using Equation 1, where q are the measurements, $\bar{q}$ is the average of these measurements, n the number of trials, S the standard deviation. Table 3 shows the repeatability for each axis. Results display very good repeatability in the three axis, around or less than 0.1mm.

$$RP = 3S_q = 3\sqrt{\frac{\sum_{i=1}^{n}(q_i - \bar{q})^2}{n-1}} \qquad (1)$$

Liquid Handling Performance

In order to measure the precision and accuracy of handling droplets, we performed various experiments with different
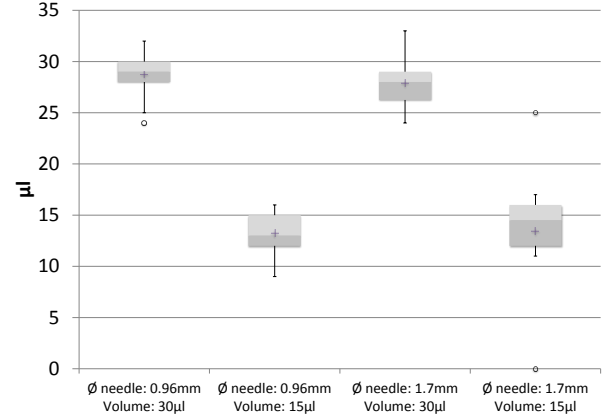

Figure 3: Boxplot comparing experiments with $100\mu l$ syringe. The whiskers represent the lowest and highest datum still within 1.5 IQR (interquartile range) of the lower quartile or of the upper quartile, respectively.

syringe sizes, needle diameters and droplet volumes. The experiment was to absorb distilled water from a Petri dish and dispense it over a scale with ±0.003g precision, and therefore obtaining droplet volume by the mass-volume relation. Four experiments were performed with a professional 100 μl syringe (Hamilton 710 LT) to handle liquid volumes of 15 μl and 30μl, with needle internal diameters of 0.96mm and 1.7mm. To obtain better results in these experiments, a small quantity of air was taken into the syringe prior to absorbing water. Four other experiments were performed with a cheap disposable 5ml syringe (Braun Omnifix) to handle liquid volumes of 1ml and 0.5ml. In these experiments, the needle internal diameters were 0.96mm and 1.7mm. Each experiment was run 30 times.

The results of the experiments have been compared using box-plots with 100 μl and 5ml syringes respectively, Figure 3 and 4. Considering the error introduced by the scale, the results obtained with the 100 μl syringe are quite reasonable. They are all accurate and the repeatability is good, except for the 15μl test with a needle of 1.7mm. This is due to surface tension, which randomly prevents the last droplet from being ejected. That amount of residual water in the tip, is taken back in during the next test, which results in an additively larger volume of water, with the fresh sample coming in. This explains the outliers of the experiment (0 and 25μl respectively). Regarding the experiments with 1 and 0.5ml, results show some inaccuracy, but an acceptable repeatability. Nevertheless, we could calibrate the robot to increase the accuracy while keeping costs low. In addition, (again due to surface tension) the last droplet is lost in almost all the tests, which decreases the repeatability. Again, this effect is more noticeable for the needle with the 1.7mm diameter. We will study how to avoid this negative effect
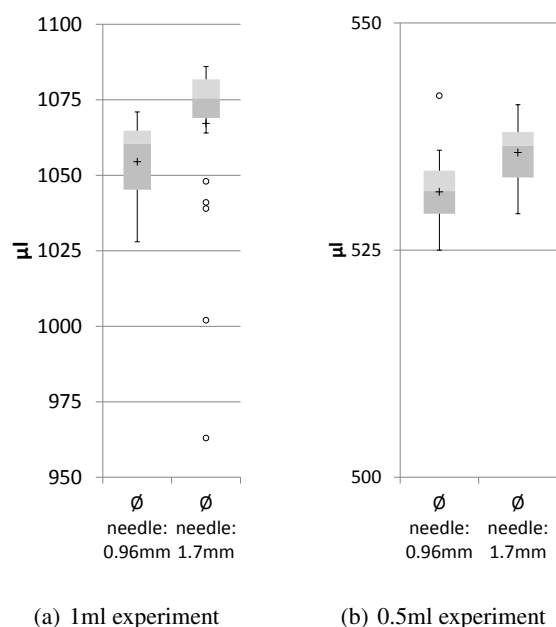
(a) 1ml experiment   (b) 0.5ml experiment

Figure 4: Boxplot comparing experiments with 5ml syringe. The whiskers represent the lowest and highest datum still within 1.5 IQR of the lower quartile or of the upper quartile, respectively.

in future work. Some alternatives could be to increase the speed of the plunger while dispensing the liquid, to generate a small vibration when moving up the syringe quickly or to use pipettes with low surface tension.

Interactive Feeding of Microbial Fuel Cells

In this experiment we document how we applied EvoBot to the task of maintaining microbial fuel cells (MFCs). The microbial metabolism, utilises the carbon energy source within the anode chamber, which eventually depletes the carbon content and results in decreasing the voltage output from the MFC; in other words, as the fuel runs out, the voltage decreases. Thus, the robot feeds the MFC with more organic material when the voltage is below a threshold. And as a result of the interaction, the experiment is prolonged.

In order to analyse the advantages of using a liquid handling robot, two parallel microbial fuel cell experiments were performed using identical materials and methods with the only difference being that one of them was carried out manually, as a replica (control) experiment, and the other one was carried out by the EvoBot platform (Figure 5). In the case of the robot experiment, the voltage is sampled every minute, the MFCs are hydrated every four hours and only fed if the voltage is below the specified threshold. In contrast, the voltage of the replica (on the bench) was sampled every three minutes, and the MFCs were hydrated twice a day - morning and afternoon - and fed once every morning.

Microbial Fuel Cell Structure

Nine small-scale, 3D printed from Nanocure® resin, open-to-air cathode MFCs were used in these experiments. The volume of each of the fuel cell anode chambers was 6.25 mL, and the anode and cathode chambers were separated by a single sheet of activated cation exchange membrane, CMI-7000S (Membranes International Inc., Ringwood, USA). Two rubber gaskets, one for each half-cell, *sandwiched* the membrane, and ensured watertight sealing, after the two chambers were bolted together using stainless steel studding and nuts.

Electrode Material (Anode and Cathode)

Untreated (catalyst free) carbon fibre veil, with $30g/m^2$ carbon loading (PMF Composites, Dorset, U.K.) was used as anode electrode with a total surface area of 168 cm^2. The anode electrode was folded down 5 times, until the projected (exposed) surface area was 5.25 cm^2 and could therefore fit into the anodic chamber (18 mm x 28 mm). The cathode electrode was made of two layers, a gas diffusion layer (GDL) and a Micro-Porous Layer (MPL). The GDL was a single sheet of carbon veil coated with 30% Polytetrafluoroethylene (PTFE) (Sigma Aldrich, UK). Once GDL cured, activated carbon paste was applied on top to form a thick MPL (1 mm). The activated carbon paste was a mixture of activated carbon powder (G.Baldwins & Co., London, U.K.) blended with PTFE in a 4:1 ratio and deionised water (120 mL). The activated carbon paste was then hot pressed, using a household iron (Gajda et al., 2015), and subsequently heated for 15 minutes to 200°C to allow MPL liquefaction.

Inoculation

For the MFC experiment on the EvoBot platform, the inoculation (i.e. introduction of live microorganisms in a sterile MFC) was done using the anolyte from already established MFCs. This was activated sludge fed with carbon sources such as tryptone yeast-extract (TYE) over a period of months, which had been sieved to remove large particles (>1 mm) so as to prevent blockage of the syringe needle on Evobot. For the replica bench experiment, neat activated sludge supplied by the Wessex Water Scientific Laboratory (Saltford, UK) was used as the initial inoculum. All MFCs in both experiments were kept under a fixed load of 3.9kΩ, for the whole duration.

Experimental software setup

Each MFC was individually connected to a separate channel on the Picolog data acquisition unit (ADC-24, Pico Technology, Cambridgeshire, UK) which was then connected to a PC, so that the DC voltage output of each unit could be continuously recorded. A set of MFC feed functions was created in LabView (National Instruments Corporation [UK] Ltd, Berkshire, UK), which sampled the Picolog DLL file every 60 seconds for the MFC voltage reading. A threshold
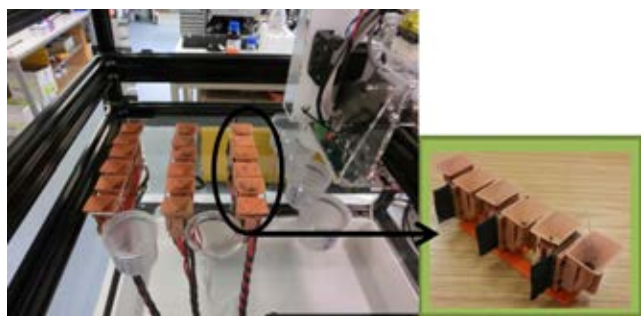
Figure 5: Interactive Feeding of Microbial Fuel Cells. Nine individually connected MFCs were operated with Evobot platform. The experiment took place on the experiment layer of the robot. The replica bench experiment was setup in exactly the same manner.
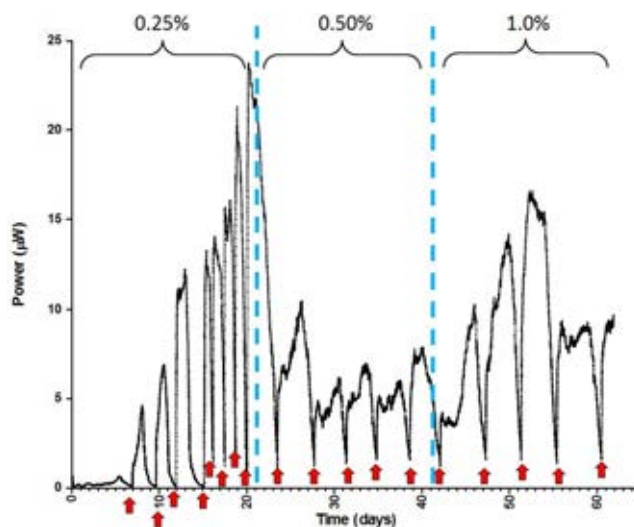

Figure 6: Power output from one of the MFCs (MFC2) fed by the Evobot with sodium acetate. The red arrows indicate the points where the voltage output of the MFC dropped below the 80mV threshold, which was the trigger for feeding the MFC. The dotted lines indicate the periods during which three different concentrations of sodium acetate were tested.

limit for each MFC was set in LabView where if the voltage dropped below this threshold, then a Python script would activate to move the head of the robot over the food source, draw 3 mL of substrate (carbon fuel) and then inject this into the MFC.

After the feed substrate has been deposited into the anode, the syringe module would go through a wash cycle where 3.5 mL of 70% ethanol was drawn into the syringe and disposed of down a waste tube on the Evobot platform, followed by the taking of 3.5 mL of sterile distilled H_2O into the syringe, before disposing this down a waste tube and then returning the robot head to the home position. At the home position, the feed function paused for 60 minutes to allow stabilisation of the MFC and for the voltage to increase above the threshold.

A cathode hydration cycle was also incorporated in a separate function in LabView, which activated a Python script every 4 hours. This script moved the robot head over the position of each MFC and deposited 3.12 mL of deionised water into the cathode chamber before returning to the home position.

Although for the Evobot experiments, the 'wash cycle' with ethanol and deionised water was deemed necessary, in order to avoid the cross-contamination between the different carbon-energy sources (acetate, lactate and cellulose), this was not necessary for the replica bench experiment, which was carried out manually.

Results

The experiment lasted approximately 8 weeks for the Evobot experiment and 4 weeks for the replica and the comparisons between the experiments are made for the same 4 week period. The power output of one MFC fed by EvoBot and the feeding events are shown in Figure 6.

A significant performance difference was found between the replica experiment and the EvBbot experiment; the replica experiment (data not shown) showed higher power

output levels from the MFCs, compared to those from the MFCs on EvoBot. This could have been due to the wash cycle with ethanol, which would have inevitably left residual ethanol in the syringe during the course of the experiment - the replica experiment did not require a cleaning cycle. Also, the sieving of sludge for the inoculation of the MFCs may have well resulted in a less enriched inoculum, and this was done to prevent the syringe needle from blocking - again, this was not an issue for the parallel bench experiment.

Nevertheless, having automated feeding pulses which were dictated by the voltage threshold, the behaviour of a MFC could more closely be monitored "day or night" in a way that would otherwise require an operator to be continuously present. In addition, the automated hydration cycle was advantageous, since it helped us identify empirically the aqueous O_2 saturation levels for the oxygen-reduction-reaction (ORR) that is necessary for the open-to-air-cathodes. In other words, beyond this 'performance saturation' point, the addition of more water did not result in an increased MFC performance. This is shown in Figure 7, which illustrates the behaviour of the acetate fed MFC, from days 31-34 (data taken from Figure 6). The fluctuating electrical output is in the response to the cathode hydration, however as can be seen, the overall performance during that feeding cycle, remains the same (on average).
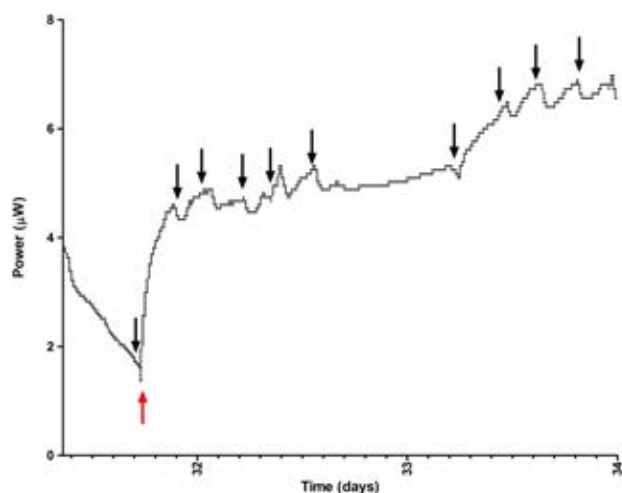
Figure 7: Power output from the same MFC2, following cathode hydrations by Evobot, during one feeding cycle between days 31-34 of the experiment. Black arrows indicate points of EvoBot hydration; the red arrow indicates a point of feeding.

Future Work

We are currently working to address the issues caused by the 'wash cycle'. A simple option would be to increase the number of times that the syringe is washed with distilled water, but we are also looking into other different techniques to avoid cross contamination. First, a dispensing module is under development, which uses external pumps to provide pure reagents or solutions. Thus, different organic materials could be used to feed the MFCs without having to use the same syringe. Additionally, we are also starting to use disposable pipette tips, but they have to be placed in the syringe manually. We hope to automate the change of these pipette tips in the future.

Regarding the interactive experiments, they are now based on the voltage of the MFCs. However, we are planning to extend the robot with sensing modules for measuring more parameters. As an example, we are developing a pH module, which will allow us to control the pH of the MFCs by adding acid or alkaline solutions. Our intention is that new interactive experiments will allow us to pose different scientific questions, possess novel data never recorded before and help us develop better MFCs.

Conclusions

The EvoBot robot is an open-source, modular liquid handling robot. Our design focuses on high quality, relying on open-source components and software, and being easily reconfigurable and extendable. Thus, EvoBot provides a versatile and low cost tool to carry out research in multiple fields. We have in this paper described the overall implementation of the mechanical structure consisting of layers

and modules and documented the performance of the robot, which are easily within the parameters required for a wide range of liquid-based experiments. In particular, a novel experiment with MFCs has been carried out, where the robot nurtures interactively an MFC based on its voltage. We hope that the unique features of this platform can form the basis for new lines of research in artificial life.

References

Gajda, I., Greenman, J., Melhuish, C., and Ieropoulos, I. (2015). Simultaneous electricity generation and microbially-assisted electrosynthesis in ceramic mfcs. *Bioelectrochemistry*, 105:58–64.

Gutierrez, J. (2012). Automatic liquid handling for artificial life research. Master's thesis, University of Southern Denmark.

Gutierrez, J., Hinkley, T., Taylor, J., Yanev, K., and Cronin, L. (2014). Evolution of oil droplets in a chemorobotic platform. *Nature Communications*, 5:1–8.

Hanczyc, M., Parrilla, J., Nicholson, A., Yanev, K., and Stoy, K. (2015). Creating and maintaining chemical artificial life by robotic symbiosis. *Artificial Life*, 1(1):47–54.

Ieropoulos, I., Greenman, J., Melhuish, C., and Horsfield, I. (2010). Ecobot-iii: a robot with guts. In *International Conference on the Synthesis and Simulation of Living Systems (ALIFE)*.

Kong, F., Yuan, L., Zheng, Y., and Chen, W. (2012). Automatic liquid handling for life science: A critical review of the current state of the art. *Journal of Laboratory Automation*, 17(3):169–185.

Müller, A., Amaldass, A., Yanev, K., and Rasmussen, S. (2015). Do it yourself (diy) liquid handling robot for evolutionary search exploration. In *European Conference on Artificial Life*.

Nejatimoharrami, F., Stoy, K., and Faina, A. (2016). An open-source, low-cost robot for performing reactive liquid handling experiments. In *Society for Laboratory Automation and Screening (SLAS 2016)*.

Yim, M., Shen, W.-M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., and Chirikjian, G. (2007). Modular self-reconfigurable robot systems. In *IEEE Robotics & Automation Magazine*, pages 43–52.

Robotic Automation to Augment Quality of Artificial Chemical Life Experiments

F. Nejatimoharrami[1], A. Faíña[1], J. Cejkova[2,3], MM. Hanczyc[2] and K. Stoy[1]

[1]Robots, Evolution, and Art Lab (REAL), IT University of Copenhagen, Denmark
[2]University of Trento, Italy
[3]University of Chemistry and Technology Prague, Czech Republic
fnej@itu.dk

Introduction

A key objective of artificial life research is grasping how non-living matter can recreate the essential properties of life (Hanczyc et al. 2015). An understanding of the essential properties of life not only may result in synthesizing artificial life and self-reproduction (Bedau et al. 2010), but may also lead to understanding the complexity of natural life. In this paper we focus on movement of droplets in concentration gradients, which can be viewed as a simplistic model of life that mimics the behavior of cells that move away from their metabolic products into regions with fresh nutrients (Hanczyc et al. 2007).

Droplet experiments require precise positioning of reagents with respect to stationary or even motile droplets. This makes them difficult to replicate by hand because humans introduce systematic errors and noise. This originates from the fact that humans are unable to precisely perceive spatial distances and dynamics. Even if they could, placing reagents at a known distance by hand is imprecise. Furthermore, the angle of the pipette tip, the time to dispense liquid, the force of dispensing, and the distance between pipette tip and liquid surface are parameters that could potentially affect the experiment. In addition, over the course of long experiments, human exhaustion could further affect the experiment.

The confluence of computer vision and robotic automation makes automation of these experiments possible. Real-time analysis of experimental images provides data about droplet properties and behavior. The preciseness of robot automation enables significant reduction of noise related to positioning of reagents and control over parameters such as fixed dispensing angel, dispense time, dispensing force, and dispense distance to liquid surface. It is also possible to make reactive experiments because the data obtained from the computer vision system may trigger the robot to perform a specific action. Hence, the introduction of automation makes it is possible to control experimental parameters precisely and significantly reduce the noise leading to improved statistical significance of results.

In this paper we automate an experiment whose purpose
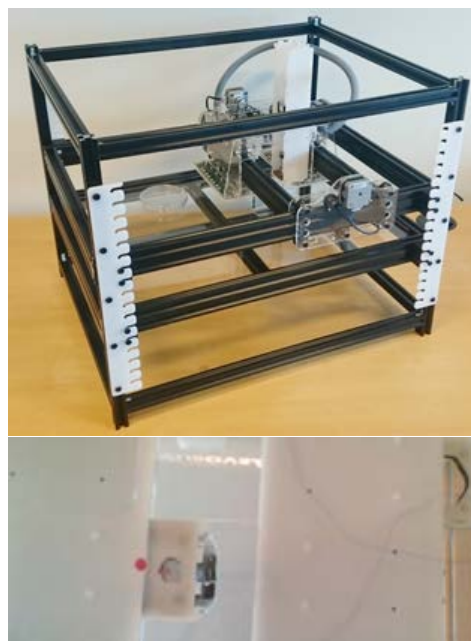

Figure 1: EvoBot (top), bottom camera view of decanol droplet on microscope slide (down).

was to understand the response of a droplet as a function of distance to a reagent (Cejkova et al. 2014). Chemists did this experiment by hand, but given they were not able to place reagents precisely they had to rely on intuition and luck to get a sufficient coverage of distances. Whether they were successful or not could first be verified after the experiments were performed by analyzing the experiment videos. In contrast, by employing computer vision driven automation it is easy to ensure systematic coverage of relevant distances and we were also successful in reducing the noise of the experiments.

Implementation

The robot we have developed, EvoBot shown in Figure 1 (top), is one possible design meeting the image processing and robotic automation requirements to perform auto-

mated artificial chemical life experiments (Faina et al. 2016). EvoBot is designed in 3 layers. An experimental arena, to hold microscope slides, Petri dishes or other reaction vessels. On top of the experimental layer is the robot head, accommodating up to 11 syringe modules to absorb or dispense liquid. The sensor layer is under the experimental layer where the camera is placed.

Various droplet data collected from the image analysis of the camera is used as feedback to make artificial life experiments possible. The camera provides data about droplet properties, such as position, speed, area, and color, or about change in droplet behavior, such as droplets merging or dividing, clustering or declustering. Based on this data, EvoBot will interact with the experiment.

EvoBot can be used to automate a variety of artificial life experiments (Nejatimoharrami et al. 2016). We have used EvoBot to track motile droplets, and interact with the experiment, e.g absorbing a droplet when its speed goes below a threshold. We have also used EvoBot to detect when a collection of droplets behaves in a certain way, e.g droplets clustering, and accordingly interact with the experiment, e.g inject a reagent at a certain distance from the droplet cluster. Another application of EvoBot is 2D or 3D positioning of droplets at precise positions or in specific patterns forming complicated geometric shapes.

Experiment Results

To verify the reduction in variability of reactive artificial life experiments, we used EvoBot to duplicate the experimet reported by Cejkova et al. 2014, as shown in Figure 1 (down). The experiment was performed 15 times with 1 mL of 10 mM sodium decanoate, $5\mu L$ droplet of decanol, and $10\mu L$ droplet of a 1 M NaCl solution (i.e., $10\mu mol$ of NaCl) dispensed at a distance of 50 mm to a decanol droplet.

Figure 2 shows the position of a decanol droplet over time, its induction time, i.e., the delay between NaCl addition and the start of its chemotaxis, and chemotactic droplet speed. The results of our experiments verify the behavior of decanol droplets in presence of salt concentration gradients observed by (Cejkova et al. 2014). Comparison of the results obtained from pipetting by hand and the robot, shows a reduction in the variability of the results from the robot, in particular about 26% decrease in coefficient of variation for the induction time.

Conclusion

In this work, we described how we applied computer vision and automation to improve an artificial chemical life experiment. For this experiment the use of the robot enabled systematic control of experimental parameters resulting in less noisy experimental data. Overall, we conclude that computer vision and robot automation make it possible to perform chemical experiments where precise spatial placement and timing are important.
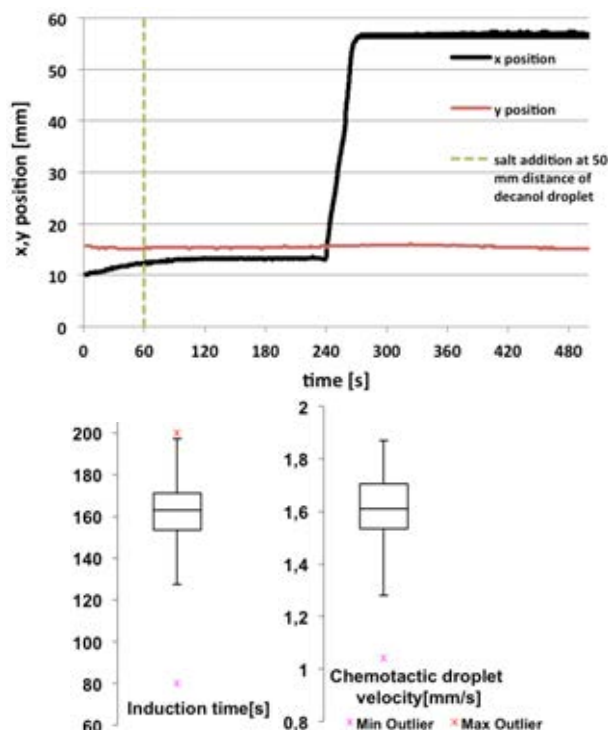

Figure 2: The position of the droplet as a function of time (top), induction time (left), and chemotactic droplet speed (right).

References

Bedau, M., McCaskill, J. S., Packard, N., and Rasmussen, S. (2010). Living technology: Exploiting lifes principles in technology. *Artificial Life*, (1):89–97.

Cejkova, J., Novak, M., Stepanek, F., and Hanczyc, M. (2014). Dynamics of chemotactic droplets in salt concentration gradients. *Langmuir*, 30(40):11937–11944.

Faina, A., Nejatimoharrami, F., Stoy, K., Theodosiou, P., Taylor, B., and Ieropoulos, I. (2016). Evobot: An open-source, modular liquid handling robot for nurturing microbial fuel cells. In *Proceedings of the 15th International Conference on the Synthesis and Simulation of Living Systems*.

Hanczyc, M. M., Gutierrez, J. M. P., Nicholson, A., Yanev, K., and Støy, K. (2015). Creating and maintaining chemical artificial life by robotic symbiosis. *Artificial Life*, 21(1):47–54.

Hanczyc, M. M., Toyota, T., Ikegami, T., Packard, N., and Sugawara, T. (2007). Fatty acid chemistry at the oil-water interface: Self-propelled oil droplets. *Journal of the American Chemical Society*, 129(30):9386–9391.

Nejatimoharrami, F., Faina, A., and Støy, K. (2016). Reactive vision-based liquid handling robot for macro-scale droplet experiments. *to be submitted to Journal of Laboratory Automation*.

Artificial Interaction between Two Isolated Micro-Algae Populations for Autonomous Pattern and Rhythm Formation

Kazunari Ozasa[1], June Won[2] June, Simon Song[2], and Mizuo Maeda[1]

[1]RIKEN, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan
[2]Hanyang University, 17 Haendang-dong, Seongdong-gu, Seoul, 133-791, Korea
ozasa@riken.jp

Abstract

We demonstrated new scheme of artificial life, which conducts the temporal evolution of real-living-cell distribution by giving programmable interactions to the cells. By using optical interlink feedback, two groups of isolated micro-algae cells were artificially interacted each other. The micro-algae cells responded to the illumination pattern produced with artificially designed algorithms, leading to the autonomous evolution of cell distribution in micro-aquariums. Habitat domain separation and autonomous oscillation of cell density were realized with the interlink feedback. In habitat domain separation, the initial fluctuation of cell density distribution grew with the interlink feedback, accompanying clustering of high-density areas. In autonomous oscillation, the photo-responses of two micro-algae determined the period and waveform of the oscillation.

Introduction

Various life-based phenomena are derived from interaction among different living species. For an example, when two microbial species interact repulsively each other, the habitat domains of each species were spontaneously separated, resulting in habitat domain separation (Schloter, 2000). One promising way to investigate the life-based phenomena and their temporal evolutions is to compose a hybrid system with living cells and a computer system to generate external stimuli. The natural behavior of living cells is affected by the external stimuli generated by programmable manner, leading to the spontaneous evolution of cell distribution through the artificially designed interaction between the cells.

Here we describe our artificial interlink feedback system with confining motile and photo-responsive micro-algae cells in two separated micro-aquariums. By utilizing the photophobic responses of the micro-algae cells, the controlled interaction between two groups of micro-algae cells was achieved; the irradiation of light patterns dynamically produced according to a designed algorithm evokes the photophobic responses of the cells. Autonomous domain separation and oscillation between the two micro-algae cells were demonstrated as the result of the artificial interlinking.

Experiment

One single optical feedback system was composed of a microscope, objective lens, video camera, data processing PC,

LC projector, and reduction lens system. The detail of the system can be found in our previous report (Ozasa 2013). Micro-algae cells were confined in micro-aquariums, which had 5×5 squares of 480 µm in width, connected to neighboring squares with paths of 90 µm in width, as shown in Fig. 1. The micro-aquarium was 120 µm deep. Several hundreds cells of *Euglena gracilis* or *Chlamydomonas reinhardtii* were used for one micro-aquarium. The cell activity in 25 individual squares was evaluated as a set of new measure "trace momentum (TM)". Based on the set of TM values, illumination patterns were produced according to a designed algorithm, and projected onto the micro-aquarium. The illumination evoked the photophobic responses of the cells, and the spatial cell distribution was changed through the photo-responses of the cells. The feedback time step was approximately 1.47 s/cycle.

The artificial interlinking between two sets of the above optical feedback systems was realized by exchanging the data sets of TMs each other. Two types of the feedback algorithms were used, one for habitat domain separation and one for autonomous oscillation. For the habitat domain separation, the illumination intensity for each of 25 squares was determined as to proportional to the normalized TM of the square positioned correspondingly in the counter micro-aquarium. For the autonomous oscillation, the 25 squares were divided into two groups of 12 fixed squares, and the illumination to the 12 squares was switched on/off all together, with a condition that the group TM for the corresponding group in the counter micro-aquarium exceeded a prefixed threshold ratio of the grand total TM. For simplicity, we denote one of the two micro-aquariums as A, and the other as B.

Results and Discussion

We used *E. gracilis* for both of micro-aquarium A and B in the habitat domain separation experiment. After the interlink feedback started, the number of squares occupied (having a larger normalized TM) by the micro-aquarium A was increased gradually, overcoming that of the micro-aquarium B until a time step of 1200, as shown in Fig. 1(a). After many transitions, the number became balanced between the micro-aquarium A and B at the end of the experiment. Figure 1(b) shows the final trace image obtained at a time step of 4000. The squares with a high cell density were clustering into a

larger domain. A few squares with a high cell density remained un-clustered, probably because the surrounding squares were taken by the counter micro-aquarium.
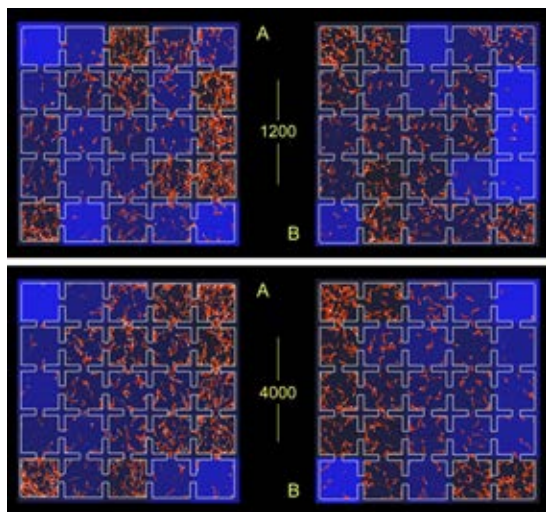

Figure 1: Trace image of micro-aquarium A and B observed at a time step of 1200 (a, top) and 4000 (b, bottom). The traces of swimming cells were displayed by red lines. Illumination intensity for each square was super imposed as blue colors.

The habitat domain separation realized in the experiments was caused by the enhancement of initial fluctuation of cell density among the squares, and by the clustering of the squares of higher cell densities. The result in this experiment can be considered as artificial community formation among the real living cells. Many types of artificial clustering can be produced in the time course of the experiment, by modifying the interlink feedback algorithm.

For the autonomous oscillation experiments, *C. reinhardtii* cells were confined in the micro-aquarium A, whereas *E gracilis* cells in B. The photophobic response of *E. gracilis* cells is to escape from the light, whereas that of *C. reinhardtii* is to be activated by the light.

Figure 2 shows the temporal change of cell distribution and light illumination for each micro-aquarium, observed with a prefixed threshold value R = 0.3. The cell distribution is represented by A1 - A2 or B1 - B2 in Fig. 2, which is the TM difference between illuminated and un-illuminated square group. An autonomous oscillation with a period of approx. 5.9 min was observed, showing that the natural photo-response of each micro-algae generated the cell density oscillation via artificially produced interlinking. The cells of *E. gracilis* migrate out from the illuminated area, whereas those of *C. reinhardtii* are activated in the illuminated area. This difference in survival strategy between two micro-algae species determined the period and waveform of the oscillation. When the prefixed threshold value R was reduced to 0.1, the oscillation period became approx. 1.9 min, indicating that the oscillation can be tuned by the parameter.

The new scheme of artificial life demonstrated here can also be used for soft-computing by employing artificial optical stimulation with light patterns. The natural cell behavior based on their survival strategy will affect the soft-computing process, which will be useful for solving combinatorial optimization problems or network flow problems.
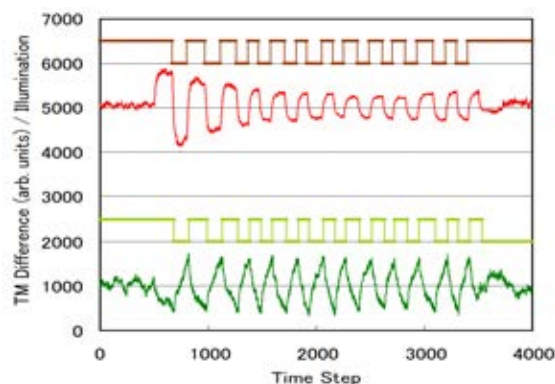

Figure 2: Autonomous oscillation observed with interlink feedback. TM differences between two groups of 12 squares were plotted for the micro-aquarium A with *C. reinhardtii* (upper) and B with *E. gracilis* (lower), together with illumination switching On/Off. TM differences were vertically shifted for easy-to-see. The interlink feedback was active for time steps of 500-3500. The prefixed threshold level was 30% in this experiment.

Conclusion

A new scheme of artificial life has been demonstrated, which conducts temporal evolution of real-living-cell distribution by giving programmable interactions to the cells. The habitat domain separation in nature was reproduced with the scheme with using two isolated groups of *E. gracilis* in the micro-aquariums. Two cell groups were correlated with an artificially designed algorithm to avoid a high cell density area in the counter micro-aquarium through the designed algorithm. The habitat domain separation was gradually developed as the reversal illumination patterns, starting with the initial cell density fluctuation, accompanying the clustering of the un-illuminated squares with higher cell densities. We also demonstrated autonomous oscillation of cell density for two different species of micro-algae, *E. gracilis* and *C. reinhardtii*. The oscillation period and waveform were determined by the difference of photo-response between the two micro-algae. The study shows that various interactions between living cells can be realized with artificially designed interlink feedback algorithms, which will contribute to developing artificially interlinked life systems and soft-computing with living cells.

Reference

Schloter, M., Lebuhn, M., Heulin, T., and Hartmann A. (2000). Ecology and evolution of bacterial microdiversity. *FEMS Microbiology Reviews*, 24:647-660.

Ozasa, K., Lee, J., Song, S., Hara, M., and Maeda, M. (2013). *Appl. Soft Comput.*, 13:527-538.

Support Vector Machine and Spiking Neural Networks for Data Driven prediction of crowd character movement

Israel Tabarez-Paz[1], Isaac Rudomin[1] and Hugo Pérez[1]

[1]Barcelona Super Computing Centre
israeltabarez, rudomin.isaac, vhpvmx @gmail.com

Abstract

Microscopic crowd simulation usually uses ad-hoc models. While these have been proven to be useful, they are difficult to calibrate and do not always reflect real behaviour. For this reason we propose a machine learning approach using neural networks. The main contribution of the project is a first exploration of prediction of agent trajectories using two specific types of neural networks, Support Vector Machine (SVM) and Spiking Neural Networks (SNN).

Introduction

Developing a computing model for simulating the behaviour of large groups allows us to study the movement of people and improve buildings, plan logistics in health, retail and other services, or even void traffic jams. Also, these models allow us to reduce risks during natural disasters and medical emergencies.

There are two ways in which crowd simulation is usually carried out: macroscopic and microscopic Rivas et al. (2014). Macroscopic simulation describes global interactions with the environment and the crowd itself. In contrast, microscopic simulation exposes the interactions between individuals within a group; each agent is processed individually to simulate a crowd. In particular, Agent-Based Models (ABM) simulate the actions and interactions of autonomous agents that generate global-scale behaviours Bonabeau (2002). One of the behaviors of such autonomous agents is to decide where each will move next based on their perception of their environment. For this purpose, the usual methods in microscopic crowd simulation are ad-hoc models such as steering rules, social forces, geometric methods such as ORCA or even approaches based on synthetic vision.

All these have been proven to be useful, but they are difficult to calibrate to reflect real behaviour. For this reason we propose using real data and a machine learning approach. We will be using trained neural networks to make decisions as to what the next step a simulated agent should take, in effect using a prediction of their trajectories. In the following sections we explore some background: crowd behavior models and artificial neural networks (in particular spiking neural networks, SNN, and support vector machine, SVM, which are the ones we will use).

On the other hand, in previous work we have proved the efficiency of third generation neural networks (SNN and SVM) over other Artificial Neural Networks (ANN) in order to solve classification problems for separable and non separable data. In fact the processing speed for training and execution is better when these methods are parallelized, Paz et al. (2014). This is the main reason why in that article we selected both methods for use in classification. Now, the most popular methods used for obstacle avoidance and to predict trajectories have been tried by Huang et al. (2016) and Abbeel et al. (2008); the evaluation of local costs is considered in these papers to get the less computationally expensive trajectory $A*$. Natural movement of crowds cannot always be obtained by simply evaluating costs locally, although for single agents it could work. Real people can see further than what we can cheaply evaluate at during simulation. This represents the main advantage of using real data to train ANNs instead of locally evaluating ad-hoc models. Our main contribution is to show that we obtain trajectories that avoid obstacles and produce what seems to be a more natural movement by using a prediction system that applies third generation neural networks to characters in crowds and to individuals. That is, a method that can use the knowledge embedded in the trajectories followed by real people, that is not explicitly present when taking decisions by evaluating costs locally.

In the rest of this paper we will explore trajectory prediction, perform some experiments, obtain some results and come to some conclusions.

Background

Reynolds (1987) proposed three basic rules for the behavior of members of crowds, which are: separation, alignment, and cohesion. These remarkably simple rules maintain together a group of boids, give them a direction of movement and keep them free of collisions. Another important approach is social forces Helbing and Molnar (1995), where agent behavior is based on a collection of forces,

called social forces. These can be attraction or repulsion. In its simplest form, a pedestrian can use these forces to get to its destination and avoid obstacles or other pedestrians. In predictive/Velocity-based models Paris et al. (2007), agents calculate the velocities necessary to avoid collisions. Within these velocities they can choose how to move to their goal avoiding collisions. This concept is expanded by Van Den Berg et al. (2011); van den Berg et al. (2008), who introduce the notions of Velocity Obstacles (VO) and Reciprocal Velocity Obstacles (RVO) and the notion of Optimal Reciprocal Collision Avoidance (ORCA). Lastly, synthetic vision Ondřej et al. (2010); Moussaïd et al. (2011) is based on using visual information obtained from the perception of the environment to get a safe trajectory to the agents goal without collision.

All these models predict individual trajectories and collective patterns of motion and can have a relatively good quantitative agreement with a large variety of empirical and experimental data. A deep study of this topic is beyond the scope of this section, thus we recommend our previous work Rivalcoba and Ruiz (2013). In previous work Rivas et al. (2014) a group of real people were coupled with virtual humans to get a plausible reaction to real people. The simulation of virtual humans was based on social forces. However, in this paper we want the agents to exhibit behavior learned from real examples, since this should lead to a more natural behavior without the tedious process of hand calibration of rules or forces. There is some work in this direction, for example Rodriguez et al. (2011) presents a crowd analysis algorithm by using behavior priors that are learned from a database of crowd videos gathered from the Internet. The algorithm learned a set of crowd behaviour priors off-line. Behaviour is compared in order to validate their data-driven crowd model. One strategy to reduce the complexity is looking in a local space. They adopt a linear Kalman filter to evaluate tracking. Error of trajectory is presented, however it is important to know the efficiency of used method, which is not discussed.

Wang et al. (2016) proposes a new approach based on finding path patterns in both real and simulated data in order to analyse and compare them, by using unsupervised clustering by non-parametric Bayesian inference. They offer to take both the global and local properties of crowd motion into account for analysis of the data. In this case, author considered three parameters, state of the agents (position and orientation), state of space, a probability over the path and pattern path. They simulate 64 agents with obstacles and one important contribution is that they got several patterns of path crowds in some environments. However, their method does not directly measure individual trajectories thus does not reflect individual visual similarities. In our case, we will measure individual trajectories and measure collisions with speed, direction, goal and occupation. We believe that ANN is a less complex method and can achieve good results.

Sujeong Kim and Manocha (2016) presents an algorithm that combines realistic trajectory behaviors from videos for simulations statistical techniques to compute movement patterns and motion dynamics from noisy 2D trajectories extracted from crowd videos in order to generate realistic crowd movements performing tasks. The main limitation in this approach is that it may not work well if the layout of the obstacles in the virtual environment is different from captured in the original video. Sujeong Kim and Manocha (2015) applies interactive techniques for analyzing crowd videos combining online tracking algorithms from computer vision, non-linear pedestrian motion models from computer graphics, and machine learning techniques. Also, they use Bayesian inferencing technique to compute the trajectory behavior feature for each agent. In this case they do not require learning a dataset. Lee et al. (2007) is focused on learning an agent model that controls the motion of each agent in a crowd, what is based on a locally weighted linear regression. Their model can be learned to imitate the rule-based flocking or insects. Also, they uses attraction to keeps the local formation of agents.

Artificial Neural Networks

An Artificial Neural network is a system composed of simplified abstractions of neurons that are used to solve computational problems by imitating the way neurons are fired or activated in the brain, in which many neurons work in parallel to produce a result. There are three ways a neural network can learn: Supervised learning, Unsupervised learning and Reinforcement learning. These methods all work by either minimizing or maximizing a cost function, but differ on the way this cost function is defined. In supervised learning, example inputs and the correct output are used to train the network. Unsupervised learning only uses inputs, and the network figures out relationships or categories. A reinforcement learning neural network learns from examples of actions and by evaluating their cost and assigning rewards and penalties. Throughout their development, ANN's have been evolving towards more powerful and more biologically realistic models. In the last decade, the third generation Spiking Neural Networks (SNN's) have been developed which comprise of spiking neurons. Information transfer in these neurons models the information transfer in biological neurons, i.e., via the precise timing of spikes or a sequence of spikes. Addition of the temporal dimension for information encoding in SNNs yields new insight into the dynamics of the human brain and has the potential to result in compact representations of large neural networks. As such, SNN's have great potential for solving complicated time-dependent pattern recognition problems defined by time series because of their inherent dynamic representation. The two important methods that we considered to predict the trajectory are Spiking Neural Networks (SNN), mentioned above, and Support Vector Machines (SVM). In both cases

they are highly parallelizable by using GPUs, Tabarez-Paz et al. (2013).

Comparison between SNN and SVM

Respect to advantages of SVM, it finds the Global Minimum Value, while ANN finds the Local Minimum Value for separable and non-separable data Burges (1998), Cortes and Vapnik (1995). The VC dimension of SVM is infinite, however its computational complexity is high. According to Yang et al. (2009) SVM has $O(n3)$ computational complexity and $O(n2)$ memory complexity in the training phase, where n is the training size. Furthermore, the number of support vectors, that is equal to quantity of neurons in the hidden layer, and it grows linearly with n and the computational complexity is $O(n)$, where the kernels are given by the training inputs Horvath (2003). This implies a limitation for real-world applications, whose training size is typically far beyond thousands. However, the complexity of the network architecture is independent of the dimension of the hyperplane. Although back-propagation converges to the Minimum Local Value, the solution is related to the number of hidden layers and number of neurons in the hidden layers. For Suykens et al. (2002), SVM solutions are characterized by convex optimization problems, up to the determination of a few additional tuning parameters. In contrast, SNN for discontinuous time, its computational complexity depends on discrete time $t(n)$, the time complexity is $O(t(n))$, where n is the discrete time.

In the case of **continuous time**, the computational complexity is logarithmic as is highlighted by Maass (1995) in theorem 1.

Theorem 1 *The VC-dimension and pseudo-dimension of any SNN N with piecewise linear response-and threshold-functions, arbitrary real-valued parameters and time-dependent weights can be bounded (even for real valued inputs and outputs) by*

$$O(|E| \cdot |W|.S(log|V| + logS))$$

if N uses in each computation at most S spikes where V is the number of neurons, S is the number of synaptic connections, and $VC - dimension$ is for computations with up to S spikes as large as $\Omega(|E| \cdot S)$

In the training process data needs to be codified in the time dimension, Ghosh-Dastidar and Adeli (2009), so it does not depend on whether the database has one or more types of data (multiclass). Nevertheless, there are many applications that have used SVMs, such as classification database, prediction, pattern recognition and regression.

Performance for SVMs is better than typical ANNs in large databases for separable and non-separable data. There are important differences between these typical ANNs and SVM:

General view of our approach

This work focuses on the decision making process of virtual agents that must pick a position to go to. These agents will consider their situation and use a trained ANN to predict this next position. The learning process uses data obtained from trajectories of real pedestrians, extracted from video. From these trajectories, the 2D positions of all pedestrians of the crowd in any given frame are obtained, and from them we obtained what we call an occupation code, in other words, the trajectory database provides specific information related to the position of each person in a given frame. From this data we know the next position of each person and all people nearby. In this way we obtain the occupation code of each agent in each frame.

With this data, and given an objective and a position we train an ANN that is then used for simulation. The simulator uses OpenGL and CUDA 7.0 and was tested in a workstation containing both a Tesla K40c and a GeForce GTX TITAN.

The main parameters that we must take into account is the position and velocity of the nearby pedestrians, how near these pedestrians are, the final objective, and the density of the crowd. Our main motivation is getting a trajectory that is more natural in a crowd simulation by using the trained neural network to predict the next step of each character instead of evaluating costs by using only local information.

The trajectories were obtained from third person top view videos of the crowd such as those available in Lerner et al. (2009). These consist of control points of Catmull Rom splines of said trajectories.

A block diagram of the system can be seen in figure 1. In other words, Catmull Rom splines representing trajectories are obtained from the data collection and their control points ordered according to frame number. Since not all characters have control points that appear in the same frame, in order to get the occupation code for all agents, we have to interpolate the position of all the agents for all the frames. After that, we obtain a large table that contains position, direction, speed, frame, sense and goal. Finally, in another algorithm we apply the data calculated in the interpolation to compute the occupation code for each agent. However, repetitive occupation codes are obtained and the table must be simplified. Finally this data will be used to train the ANN.

Simulating crowd behaviour using Artificial Neural Networks

We have chosen to use SNN's due to their characteristics for training multiclass data with various attributes per instance as well as for its acceptable efficiency, although we also considered SVM for prediction in order to compare SVM results. We highlighted the term **Multiclass** for a database if data can be classified in more than one type according to a common characteristic. For example in Weka3, there is a free database that can be downloaded in order to test some classification algorithms for prediction or clustering, an Iris
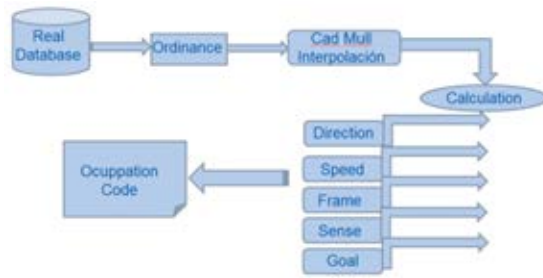
Figure 1: Block Diagram of System.

data set is classified in three categories of flowers because of some common physical characteristics Bifet et al. (2015).

In our case, we have seven classes that are the seven directions of the agents and our dataset is the occupation code, with a size proportional to the variety of the cases collected from the trajectories in the videos. This data depends on some variables such as the speed of the agent, the distribution of other agents in the viewing area, and the density of the crowd. As mentioned above, we use trajectories obtained from videos of pedestrians in a crowd, and we interpolate the data to get positions per frame of all agents, and from these calculate the direction of movement of the agents in the previous and following frames, as well as their speed. The process consists of codifying the occupation patterns around each element in the crowd. The videos and trajectories are those used by Lerner et al. (2009) and can be seen in figure 2.

First experiment: Results using SNN

In our first experiment we used a set consisting of the occupation code and the goal of every agent as a row in the input matrix (M) of the SNN. After that, the output of the ANN is the decision that was taken for the next step of the pedestrian. The occupation code was computed using a radius of agent and an angle of view divided in seven sectors, as can be seen in figure 3, each of which represents one bit of the occupation code. We define the radius of view as the maximal distance that will be explored in order to decide that a sector of circle of the respective bit is occupied.

The set of seven bits form an occupation pattern. If there is a pedestrian is in this sector, then the sector is occupied and the bit is one but if the sector is empty the bit is zero, such as in figure figure 4.

Results of this experiment using SNN with the codification explained above is presented in figure 5, so we can check two simulation tests: in the first one (Test 1), agents start all at random positions the same line at the bottom, and have as their goal a single point in the top: we see their trace from start to finish. In (Test 2) the other tests all agents lie on a circle, and each has as its goal the point exactly on the opposing side of the circle, usually called circle test. Both simulations used 100 agents, and by the time the agents had

reached their goal Test 1 had 396 and Test 2 had 72 collisions, respectively; most of thee collisions were concentrated in a very few agents.

We performed other testing; 100 agents whose position is divided into the four sides of a square that (Test 3). Initially they are randomly distributed on the four sides of a square, and have a randomly assigned goal on the line opposite to their initial position. In (Test 4) agents are initially distributed on a grid. In each case the goal has the same x coordinate but the y coordinate is on the opposite side. Test 3 had 17 collisions and Test 4 had 0 collisions.

In order to take the distance of collision into account and to reduce this average improving precision, we tried using a different coding using three radii, as is described in the next section.

Second experiment: Results using SNN and SVM

After obtaining the first results we observed that the simulated characters detected collisions and avoided them but that their reaction was not very natural: in particular when crowd density increases. We therefore proceeded to use an expanded coding which consists of dividing the sectors by three different arcs, with three different radius , as we can see in figure 6, still dividing each arc in seven sectors.

To code we took into account the distance from the center of the semicircle to the agent in the three radii, i.e. the number of bits, where it had 1, this number is replaced by its distance from the centre with three options, so, the number 1 will replaced by 2 if its distance is between $\frac{3}{4}Radius$ and $Radius$; or will be replaced by 1 if its distance is between $\frac{2}{4}$ and $\frac{3}{4}Radius$; or will be replaced by 0 if its distance is between 0 and $\frac{1}{4}Radius$, such as in figure 6. This code represents the empty spaces, or steps, that agents could move toward its goal according to the predicted step.

Additionally we controlled the rotation per agent. In other words, after getting the predicted step, the agent rotates toward the goal. Also, agent speed is coded using the seven digits, for example if the speed of nearest intruder is bigger than that of the main agent, that speed bit will be 3 and the agent can consider that space as empty for the next step; also if the speed of nearest intruder is the same that of the main agent, that speed bit will be 2 and also this space could be considered empty; but if the speed of nearest intruder is smaller than that of the main agent, then that speed bit will be 1 and the agent could not consider that empty for the next step.

Once the SNN has been trained the simulation is carried out as explained before: given an occupation pattern of the agents and the objective vector the next position is determined by the output of the trained neural net.

In case of some sector being occupied but with velocity component greater or equal in the direction of that of the agent being analysed, the sector is considered empty, figure 7 .

Figure 2: Images from sample videos. Trajectories derived from these videos were used as input data.
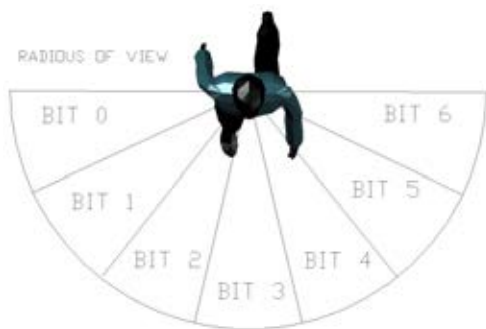

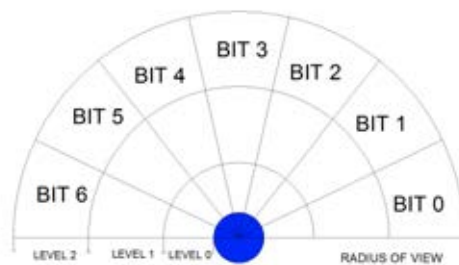Figure 3: Proposal 1: Codifying the occupation pattern.

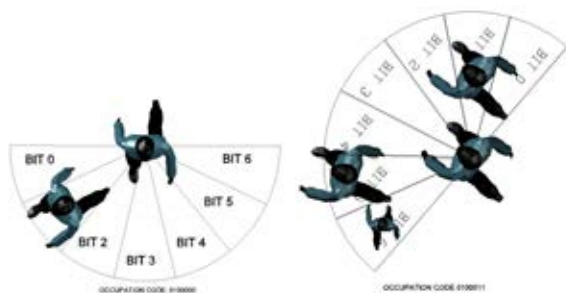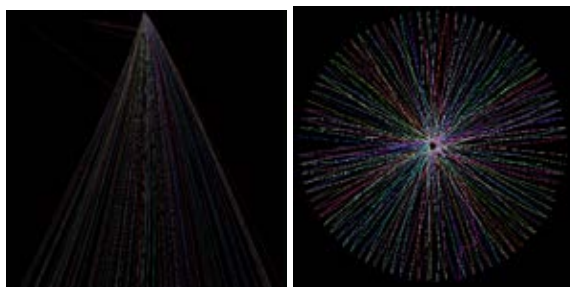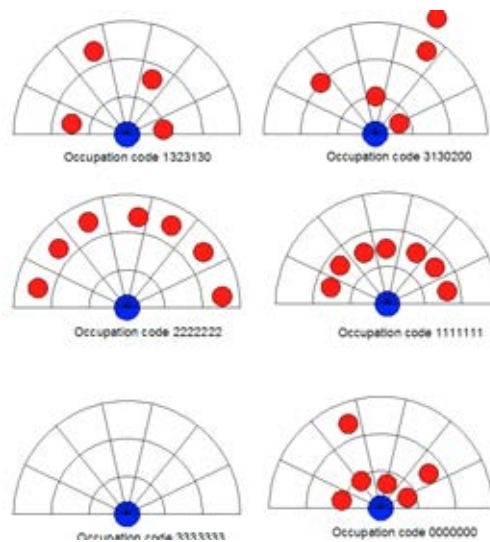Figure 6: Codifying with three radii

Figure 4: Test 1: A, B

Figure 7: Proposal 2: Cases of codifying.
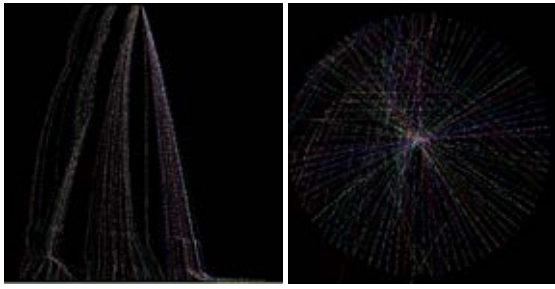

Figure 5: Test 1, 2
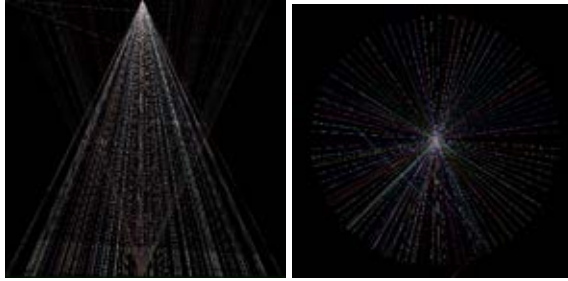
Figure 8: Test 5, 6

Figure 9: Test 9, 10 SVM: Finishing in the opposite goal with 100 agents

Second experiment: Results using SNN In the following paragraphs we present some results of experiments made with SNN using the second approach. In figure 8, test 5, agents start in the opposite place of the goal, with 27 collisions and deviation standard of 0.93; in test 6 the circle test was tried with 49 collisions with deviation standard of 1.6. That means that the most of agents not contributed with collision. We used 100 agents with SNN.

We conducted other tests: (Test 7) is like Test 3, and Test 8 is like Test 4. In this case we had only 4 and 0 collisions respectively. With respect to precision, this was not better than first approach, but most of agents got less than 1 unit of distance to the goal and there were fewer collisions.

We got better results with this approach than the first. So, it is important to compare with SVM with this approach, as shown in the next section.

Second experiment: Results using SVM The same experiments were tested, using this second coding approach, but using SVM to test the advantages or disadvantages of SVM respect to SNN, such as the ability to find the global minimal value and faster execution of prediction. In Figure 9 respect to collisions we got 227, with 227 agents, and the test circle 10 we got 779, with 100 agents. And the standard deviation was 8.79 and 15.16, respectively, significantly higher than with SNN. Probably we could get better results with other methodologies derived from SVM such us SVM light or SMO Joachims (1999). This means that collisions per agent are more with SVMs than with SNNs.

We also conducted (Test 11) analogous to Test3 and Test 12) analogous to Test 4 Collisions measured were 29 with a standard deviation of 0.52 and 157 collisions with a standard deviation of 1.02, respectively.

In figure 10 we see an example of our simulation using our visualization system for diverse animated crowds where by combining geometry from different characters and using shaders, we generate varied characters using few assets. Also, different discrete levels of detail are used to allow the system to render crowds composed up to a quarter million characters. We suggest as a reference, Hierarchical level of detail for varied animated crowds Hernández and Rudomin (2011). Likewise, techniques for generating many varied and animatable characters for crowdswith reduced memory requirements, and where the use of a texture space techniques allows painless animation transfer between different characters and for different levels of detail Ruiz et al. (2013).

Simulation speed of SVM is better than for SNN. Distance to the goal from the final position reached in simulation is also less with SVM than with SNN. But the trajectories obtained with SNN were better in avoiding obstacles and seemed to give a more natural-looking behaviour of the agents. Also, as we can see in previous work Paz et al. (2013), with respect to training time, SVM is faster than SNN. However, training is still much slower than simulation, so the design of an algorithm that learns faster, or even in real time, requires parallelization of training.

Conclusions

According to experiments, SNN has better performance for behavior of crowds than SVM at predicting trajectories, in the statistics, SNN got its best results in the second approach achieving significantly fewer collisions. On the other hand, all agents arrived to their goal with SVM, while the distance to the goal with SNN was less than 1 in most cases, so it is still possible to achieve better speed and precision with SNN in future work.

Since our model takes into account those limitations by allowing only decisions included in the trajectories contained in our database Lerner et al. (2009), which consists basically of moving characters avoiding each other, this implies that our model works rather well in that characters avoid collisions while getting to the goals. However we did not consider large static obstacles, and in preliminary testing, as expected, the system as presented has limitations in this case.

Another limitation is that precision depends on efficiency of the ANN used; in Sujeong Kim and Manocha (2015) experiments do not work with very dense crowds, and in our case we have only tried up to 255 agents.

Finally, some advantages of simulating with ANN are that carrying out mathematical computations, using any physical or mechanical data , or guessing what were the appropriate parameter settings was not necessary. However the method has results that depend very strongly on the trajectory data

Figure 10: Simulation a), b) Circle Test SNN, c), d)Test SVM finishing in the opposite goal objetive with 100 agents.

used fort training, and it should be possible to get more natural movement if the database is larger and contains more non repetitive cases.

With respect to behaviour shown in the test, we can see that the trajectories have more natural shapes and form geometric patterns that look like those of real crowds. As future work, it is important to compare this methodology with the common ad-hoc methods and to train for tests that include large obstacles.

References

Abbeel, P., Dolgov, D., Ng, A. Y., and Thrun, S. (2008). Apprenticeship learning for motion planning with application to parking lot navigation. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1083–1090. IEEE.

Bifet, A., de Francisci Morales, G., Read, J., Holmes, G., and Pfahringer, B. (2015). Efficient online evaluation of big data stream classifiers. In *Proc 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 59–68. ACM.

Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7280–7287.

Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Ghosh-Dastidar, S. and Adeli, H. (2009). A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection. *Neural Networks*, 22(10):1419–1431.

Helbing, D. and Molnar, P. (1995). Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282.

Hernández, B. and Rudomin, I. (2011). A rendering pipeline for real-time crowds. *GPU Pro*, 2:369–383.

Horvath, G. (2003). Cmac neural network as an svm with b-spline kernel functions. In *Instrumentation and Measurement Technology Conference, 2003. IMTC'03. Proceedings of the 20th IEEE*, volume 2, pages 1108–1113. IEEE.

Huang, S., Li, X., Zhang, Z., He, Z., Wu, F., Liu, W., Tang, J., and Zhuang, Y. (2016). Deep learning driven visual path prediction from a single image. *arXiv preprint arXiv:1601.07265*.

Joachims, T. (1999). Svmlight: Support vector machine. *SVM-Light Support Vector Machine http://svmlight. joachims. org/, University of Dortmund*, 19(4).

Lee, K. H., Choi, M. G., Hong, Q., and Lee, J. (2007). Group behavior from video: a data-driven approach to crowd simulation. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 109–118. Eurographics Association.

Lerner, A., Fitusi, E., Chrysanthou, Y., and Cohen-Or, D. (2009). Fitting behaviors to pedestrian simulations. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 199–208. ACM.

Maass, W. (1995). On the computational complexity of networks of spiking neurons. *Advances in neural information processing systems*, pages 183–190.

Moussaïd, M., Helbing, D., and Theraulaz, G. (2011). How simple rules determine pedestrian behavior and crowd disasters. *Proceedings of the National Academy of Sciences*, 108(17):6884–6888.

Ondřej, J., Pettré, J., Olivier, A.-H., and Donikian, S. (2010). A synthetic-vision based steering approach for crowd simulation. In *ACM Transactions on Graphics (TOG)*, volume 29, page 123. ACM.

Paris, S., Pettré, J., and Donikian, S. (2007). Pedestrian reactive navigation for crowd simulation: a predictive approach. In *Computer Graphics Forum*, volume 26, pages 665–674. Wiley Online Library.

Paz, I. T., Gress, N. H., and Mendoza, M. G. (2013). Pattern recognition with spiking neural networks. In *Advances in Soft Computing and Its Applications*, pages 279–288. Springer.

Paz, I. T., Gress, N. H., and Mendoza, M. G. (2014). Classification of database by using parallelization of algorithms third generation in a gpu. In *Engineering Applications of Neural Networks*, pages 25–38. Springer.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. In *ACM SIGGRAPH computer graphics*, volume 21, pages 25–34. ACM.

Rivalcoba, I. and Ruiz, S. (2013). Gpu generation of large varied animated crowds. *Computación y Sistemas*, 17(3):365–380.

Rivas, J. I. R., De Gyves, O., Rudomin, I., and Pelechano, N. (2014). Coupling camera-tracked humans with a simulated virtual crowd. In *Computer Graphics Theory and Applications (GRAPP), 2014 International Conference on*, pages 1–10. IEEE.

Rodriguez, M., Sivic, J., Laptev, I., and Audibert, J.-Y. (2011). Data-driven crowd analysis in videos. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1235–1242. IEEE.

Ruiz, S., Hernández, B., Alvarado, A., and Rudomín, I. (2013). Reducing memory requirements for diverse animated crowds. In *Proceedings of Motion on Games*, pages 77–86. ACM.

Sujeong Kim, A. B. and Manocha, D. (2015). Interactive crowd content generation and analysis using trajectory-level behavior learning. In *Proceedings of IEEE International Symposium on Multimedia*. IEEE.

Sujeong Kim, Aniket Bera, A. B. R. C. and Manocha, D. (2016). Interactive and adaptive data-driven crowd simulation. In *Proceedings of VR*. IEEE.

Suykens, J. A., Van Gestel, T., De Brabanter, J., De Moor, B., Vandewalle, J., Suykens, J., and Van Gestel, T. (2002). *Least squares support vector machines*, volume 4. World Scientific.

Tabarez-Paz, I., Hernández-Gress, N., and Mendoza, M. G. (2013). A survey of spiking neural networks and support vector machine performance byusinggpu's. *International Journal on Soft Computing*, 4(3):1.

Van Den Berg, J., Guy, S. J., Lin, M., and Manocha, D. (2011). Reciprocal n-body collision avoidance. In *Robotics research*, pages 3–19. Springer.

van den Berg, J., Patil, S., Sewall, J., Manocha, D., and Lin, M. (2008). Interactive navigation of multiple agents in crowded environments. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pages 139–147. ACM.

Wang, H., Ondřej, J., and O'Sullivan, C. (2016). Path patterns: Analyzing and comparing real and simulated crowds. In *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '16, pages 49–57, New York, NY, USA. ACM.

Yang, J., Yu, K., Gong, Y., and Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE.

Human-Computer Interaction

The human in the loop: volunteer-based metacomputers as a socio-technical system

Juan-J. Merelo*[1], Paloma de las Cuevas [1], Pablo García-Sánchez [1], Mario García-Valdez[2]

[1]Dept. of Computer Architecture and Technology and CITIC University of Granada
[2]Dept. of Graduate Studies at Instituto Tecnológico de Tijuana
jmerelo@ugr.es, mario@tectijuana.edu.mx

Abstract

Volunteer computing is a form of distributed computing where users decide on their participation and the amount of time and other resources they will "lend". This makes them an essential part of the algorithm and of the performance of the whole system. As a socio-technical system, this participation follows some patterns and in this paper we examine the result of several volunteer distributed evolutionary computation experiments and try to find out what those patterns are and what makes an experiment successful or not, including the feedback loop that is created between the users and the algorithm itself.

Introduction

Some time ago, our group faced the problem of diminishing funds for buying new hardware. This was aggravated by the increasing maintenance costs and extended downtime resulting from the continuous failures of existing clusters. Considering this, we leveraged our experience in the design of web applications with JavaScript and other volunteer and unconventional distributed evolutionary computing systems to design and release a new free framework that would allow anyone to create a volunteer distributed evolutionary computation (EC) experiment using cloud resources as servers and browsers as clients. This framework was called NodIO (Merelo et al., 2016). NodIO provides server infrastructure for volunteer-based distributed evolutionary computing experiments by providing a chromosome *pool*. This pool is used by clients in browsers and any other using the application programming interface (API) to put chromosomes and retrieve them, working then as a loose, asynchronous and *ad hoc* connection among all clients using it.

This loose connection provides a low-overhead way to connect desktop experiments with volunteer-based ones, with all of them contributing to the pool, but every one of them working as separate island carrying out their own evolutionary algorithms. This is why NodIO is proposed mainly as a *complement* to existing resources such as desktop systems or laptops. As long as it provides a non-null computational capability that can help existing resources find the solution faster it will have found its purpose. The main

use case is someone setting up NodIO in the cloud, writing a fitness evaluation function and running a client from his or her own computer, but requesting help in social networks for additional resources. That is why, in this system, we have considered the whole *social* aspect in the design, with issues related to security, trust and privacy among others. The computing system becomes a *socio-technical system* (Vespignani et al., 2009). In this paper, we are going to focus on measuring the response of users to experiments, that is, the time they spend running it, but at the same time we will also focus on the technical aspects of the server and how these might change the behavior of users, improving the capability of the system.

Our research group is committed to open science, and we think this is a very important part of the techno-social system. By being transparent, incentives to cheat are reduced and, in fact, we have detected no issue for the time being. Next we present the state of the art in web-based volunteer computing systems along with attempts to predict and model its behavior.

State of the art

Volunteer computing involves a user running a program voluntarily and, as such, has been deployed in many different ways from the beginning of the Internet, starting with the SETI@home framework for processing extraterrestrial signals (Anderson et al., 2002). However the dual introduction of JavaScript as a universal language for the browser and the browser as an ubiquitous web and Internet client has made this combination the most popular for volunteer computing frameworks such as the one we are using here, and whose first version was described in (Merelo-Guervós and García-Sánchez, 2015).

JavaScript can be used for either unwitting (Klein and Spector, 2007; Boldrin et al., 2007) or volunteer (Langdon, 2005; Merelo et al., 2007) distributed evolutionary computation and it has been used ever since by several authors, including more recent efforts (Desell et al., 2008; Duda and Dłubacz, 2013; Gonzalez et al., 2008). Many other researchers have used Java (Chong and Langdon, 1999); oth-

ers have embraced peer to peer systems (Jin et al., 2006; Wang and Xu, 2008; Merelo-Guervós et al., 2012). These computing platforms avoid single points of failure, the server, and once installed need no effort to gather new users for an experiment, but the cost of acquiring new users is high since they need to be set up.

The number of users is key in the performance of these systems, but it also essential to adapt the algorithm itself to the available resources, as shown by (Milani, 2004), although EAs can be readily distributed via population splitting or by farming out the evaluation to all the nodes available. However, user churn affects experiment performance (González Lombraña et al., 2010; Nogueras and Cotta, 2015) and also the performance of the algorithm itself (Laredo et al., 2014). All these issues imply that a the performance of a volunteer system cannot be measured without first understanding its dynamics. Initial work was done for peer to peer systems by Stutzbach et al. (Stutzbach and Rejaie, 2006) and extended to volunteer computing by Laredo et al. (Laredo et al., 2008a,b). A similar study was performed by Martınez et al. on the Capataz system (Martınez and Val, 2015); however, in this case the number of computers used was known in advance and the main focus was on measuring the speed up and how job bundling helped to reduce overhead and enhance performance. On the contrary, in this paper, we will use *actual* volunteers.

Some of the essential metrics in volunteer computing like the number of users or the time spent by every one in the computation in browser-based volunteer computing experiments, have only been studied in a limited way in (Laredo et al., 2014) on the basis of a single run. Studies using volunteer computing platforms such as SETI@home (Javadi et al., 2009; Merelo et al., 2008) found out that the Weibull, log-normal and Gamma distribution modeled quite well the availability of resources in several clusters of that framework; the shape of those distributions is a skewed bell with more resources in the *low* areas than in the high areas: there are many users that give a small amount of cycles, while there are just a few that give many cycles.

As far as we know, this paper presents one of the few experiments that measure the performance of a socio-technical metacomputer, that is, a spontaneously created parallel computer that uses social networks for operations such as gathering new users. Apolónia et al. (Apolónia et al., 2012) used the Facebook protocol to distribute tasks among the *walls* of friends, explicitly using the social network for computing. However, it stopped short of relating performance to the macro measures of the users' social networks. As in the previous example, a social network was used to get new network nodes; in the previous case a web page was used, while Facebook's wall was used here.

In our case, social networks are an integral part of the system and used to spontaneously obtain users. The algorithms used, as well as the methodology for gathering resources will be described next, together with the results obtained in this initial setup.

Description of the framework

In general, a distributed volunteer-based evolutionary computation system based on the browser is simply a client-server system whose client is, or it can be, embedded in the browser via JavaScript. Since JavaScript is the only language that is present across all browsers, the choice was quite clear. We should emphasize that NodIO is more intended as an auxiliary computing engine, more than the main one, so performance of JavaScript as a language is not so important; even so, we have made a comparison between JavaScript and other languages (Merelo et al., 2015) that shows that the performance of JavaScript is comparable to other interpreted languages; compiled languages would be faster, but, of course, it is impossible to gather volunteers spontaneously and without any installation with them.

In this sense, in this paper we propose the NodIO framework, a cloud or bare metal based volunteer evolutionary computing system derived from the NodEO library, whose architecture has been developed using JavaScript on the client as well as the server. All parts of the framework are free and available with a free license from `https://github.com/JJ/splash-volunteer`.

Thus, NodIO architecture has two tiers:

1. A REST server, that is, a server that includes several *routes* that can be called for storing and retrieving information (the 'CRUD' cycle: create, request, update, and delete) from the server. A JSON data format is used for the communication between clients and the server. There are two kinds of information: *problem* based, that is, related to the evolutionary algorithm such as PUTing a chromosome in or GETing a random chromosome from it, and *information* related to the performance and state of the experiments. It also performs logging duties, but they are basically a very lightweight and high performance data storage (Merelo, 2015). The server has the capability to run a single experiment, storing the chromosomes in a key-value store that is reset when the solution is found. This store can hold every chromosome in a particular experiment, or have a finite size that erases the oldest chromosomes once it has filled to capacity, acting as a cache. In this paper we will test both implementations.

2. A client that includes the evolutionary algorithm as JavaScript code embedded in a web page that displays graphs, some additional links, and information on the experiment. This code runs an evolutionary algorithm *island* that starts with a random population, then after every 100 generations, it sends the best individual back to the server (via a PUT request), and then requests a random individual back from the server (via a GET request). We have kept the number of generations between migrations fixed
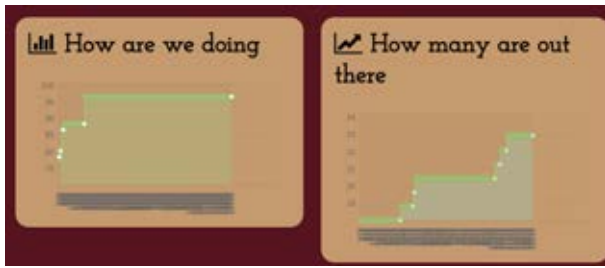
Figure 1: Screenshot showing the fitness and number of users as seen by volunteers. The one on the left shows fitness, on the right the accumulated number of users.
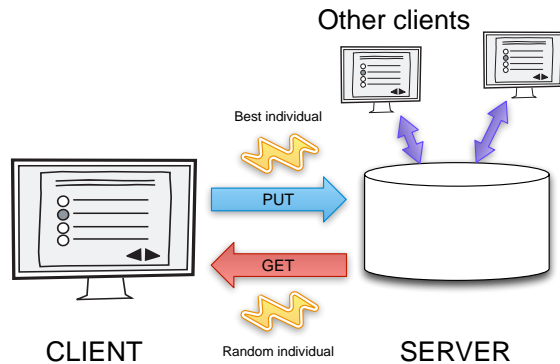

Figure 2: Description of the proposed system. Clients execute a JavaScript EA in the browser, which, every 100 generations, sends the best individual and receives a random one back from the server.

since it is a way of finding out how much real work every client has done.

Figure 2 describes the general system architecture and algorithm behavior. Different web technologies, such as JQuery and `Chart.js` have been used to build the user interface elements of the framework, a part of which is shown in Figure 1 and should be running in `http://nodio-jmerelo.rhcloud.com`.

JavaScript is a functional language, so in order to work with a problem, a fitness function must be supplied at the creation of the algorithm object, called `Classic`. In this case the classical Trap function (Ackley, 1987) has been used. Depending on the problem additional handing functions and configurations can be supplied.

The next Section will describe experiments performed to establish a baseline performance and gather initial performance results. In the first set of experiments, performed last year, we used the 40-trap problem, while the current experiments changed to the more difficult 50-trap in order to compare the performance with a more computationally intensive problem.

Modeling the performance of a volunteer-based distributed computer

Table 1: Experiment table, with summary of results.

Experiment	#Runs	Different IPs	Traps
April 4th 4/4	57	191	40
April 24th 4/24	231	559	40
July 31th 7/31	97	179	40
February, cache=128	61	75	50
February, cache=64	61	220	50
February, cache=32	39	86	50

Initial experiments were set up using the OpenShift PaaS `www.openshift.com`, which provides a free tier, making the whole experiment cost equal to $0.00. Experiments were announced through a series of posts on Twitter and, in the latest case, Telegram, and results were published in(Merelo-Guervós and García-Sánchez, 2015). For the purpose of this paper, we repeated the announcement several times through the month of February of the next year. All in all, we have the set of runs with the characteristics shown in Table 1. In general, every experiment took several days. No particular care was taken about the time of the announcement or the particular wording. Every *experiment* consisted in running until the solution of the 50-trap problem was found. When the correct solution was sent to the server, the counter was updated and the pool of solutions resets to the void set. There was no special intention to wait until all clients had finished, thus it might happen that. In fact, the islands running in the browser *spill* from one experiment to the next. If an individual that is close or even at optimum value is stored in the cache the next experiment will be affected, and it will require less time to finish. This problem has been addressed in later versions of the system, however, time to solution is not the important measure here, where we are concerned with the computational power of the system established by the number of volunteers.

The table 1shows that every experiment included more than 30 runs. The number of different IPs intervening in them varied from more than one hundred to more than five hundred in the second experiment, with a number around 50 in the second, and most recent, batch of experiments.

A summary of the results of each run is also shown in Table 2, which shows the median number of IPs intervening in each experiment, median time needed to finish the experiment, median number of HTTP `PUT`s per IP. The first striking result is that in all cases, 50% of the experiments involved 5 or less IPs. This is consistent with previous results (Merelo-Guervós and García-Sánchez, 2015) which found 6 to be the expected number of volunteer IPs. The maximum number of different IPs for each experiment is also in the same range and of the order of 10, which is also consistent

Table 2: Summary of time per run, number of IPs and number of `PUT`s per IP in the initial runs.

Experiment	IPs Median	Max	Median time (s)	#`PUT`s
4/4	5	16	2040	18
4/24	5	29	732	11
7/31	5	14	260	23
Cache=128	5	17	222.2	124
Cache=64	8	38	51.3	100
Cache=32	6	19	58.9	45

with prior work and does not vary across the two different batches.

We will have to analyze differently the median time, since the two batches are solving different problems. In both cases it possesses a big range of variation, but 50% of the time takes less than several minutes, from around 4 minutes in the best case to roughly 2/3 of an hour in the worst case. Remarkably enough, the time is more consistent in the second batch and always around one minute, in two cases even less, and that happens when the median number of IPs is higher. It should be noted that while the first batch of experiments took several days in each case, the second only lasted for a few hours, with a more continued effort of publicizing it in social networks. This is specially true in the case of cache equal to 64, which is noticed by the high number of volunteers participating in the experiment. The conclusion is that, in general, the key factor in the time needed to find the solution is, as expected, the number of volunteers it is able to gather on a short notice.

The number of `PUT`s, every one corresponding to 100 generations, is the algorithmic result. It is relatively unchanged for the first batch and around 20, that is, 2000 generations or 2000*128 = 256000 evaluations. In this case, an "unlimited" cache was used, with all individuals sent from clients stored until the end of the experiment. However, we were interested in measuring also the performance of the algorithm itself by changing the cache size, after making it limited. As it can be seen in the table, there is a clear change in the number of evaluations needed, with smaller cache sizes producing solutions in less evaluations, until it is for the cache size = 32 roughly twice as much as with the previous problem, with 40 traps. This is a good result and is also algorithmically consistent with other results obtained using the same type of problems. Since in this paper we were interested in leveraging the user's CPU cycles by improving the algorithm, a good conclusion of this paper is that having a small pool size helps clients to obtain "good" individuals from the pool, as opposed to any individual that could be obtained before. Besides, the cache policy deletes the oldest individuals, which makes those in the pool be *current*, helping then newcomers and any participant obtain the best

individuals found in the last part of the experiment. Besides, a limited cache helps also in cases with a bigger search space or longer running times when the server simply crashed due to lack of RAM.
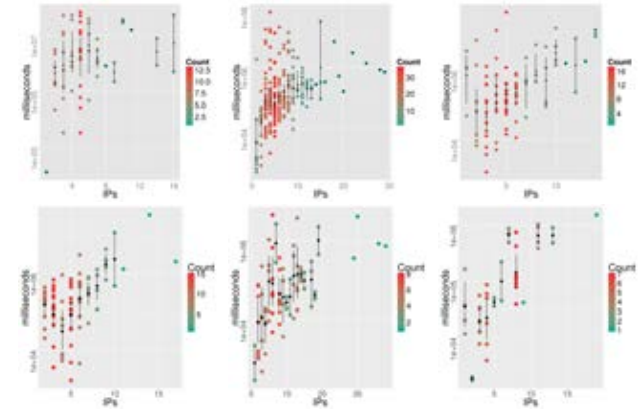

Figure 3: Duration of experiments vs. number of different IPs (nodes) participating in it, with averages per number of IPs and standard deviation shown as red dots; in the case there is a single red dot, there was a single experiment in which that many computers participated (for instance, 16 computers in the experiment in the far left or 29 in the middle one). Shades of blue indicate how many experiments included that many unique IPs, so lighter shade for a column of dots indicates that a particular number of computers happened less frequently, while darker shadow means more. From left to right and top to bottom experiments 4/4, 4/24 and 7/31, followed by experiments with 50 traps, cache=128, 64, 32.

We will have to analyze experimental data a bit further to find out why this happens and also if there are some patterns in the three sets of experiments. An interesting question to ask, for instance, is if by adding more computers it makes the experiment take less time. In fact, as shown in Figure 3, the *addition* of more computers does not seem to contribute to decreasing the time needed to finish the experiment. However, the cause-effect relationship is not clear at all. It might be the opposite: since experiments take longer to finish and might in fact be abandoned with no one contributing for some time, the probability of someone new joining them is higher. In fact, with experiments taking a few seconds and due to the way the experiments are announced, it is quite difficult that several volunteers join in in such a short period of time, even more if we take into account that volunteers are not *carried over* from previous experiments.

That is why we used a more difficult problem in the second batch of experiments, which is shown in the bottom row of Figure 3. The pattern is remarkably similar, showing a positive correlation between the time for solving the problem and the number of computers, at least for cache sizes
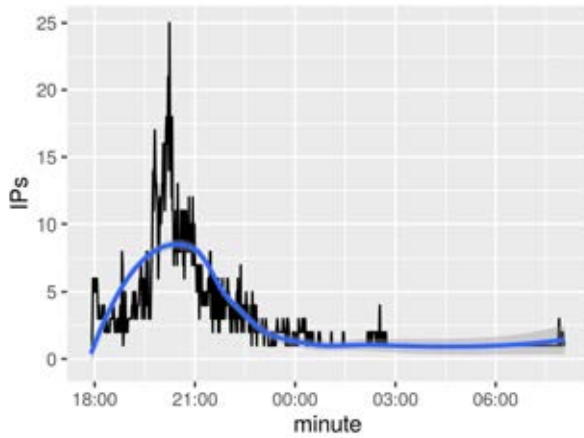
Figure 4: Simultaneous IPs every minute of the experiment with cache = 64.

128 and 64. However, it is interesting to observe that, for cache=32, the time needed to find the solution decreases from one to approximately 4-5 nodes, to then increase for a higher number of participating computers, distinguished by IP. The green dot at the bottom is probably an outlier that we will try to explain later on. This leads us to conclude that a larger amount of computers might contribute to speed up the solution, if the time the experiment ideally takes is sufficient, that is, of the order of a minute, and enough volunteers concur simultaneously. This is also observed, not so clearly, in the case of cache=64, with an interval of around 10 IPs obtaining less time than experiments with less or more IPs, and of the same order, between 10 and 100 seconds. If we look at the graph that shows the number of IPs or volunteers per minute for this experiment, shown in Figure 4, we see that there are peaks of more than 25 volunteers, and a period of several hours with a minimum of 6 computers and peaks of more than 10. The long period after midnight where there is a single volunteer left masks the success achieved during this set of experiments, from which we draw two lessons: first, you need a social network influencer to announce your experiments and second, no matter what, do not do any experiment after midnight. This statement, which might seem tongue in cheek, in fact, it is a conclusion drawn from the experimental data and to what extent the social network is an essential part of the description and performance of the NodIO volunteer computing system.

It is also interesting to check the distribution of the experiment duration, shown in Figure 5 and which roughly follows a Zipf's law, with similar distribution along all three runs. The 4/24 run is the most complete and shows an S-shape, which implies an accumulation of experiments taking similar time and around 100 seconds; this S-shape appears too in the experiments with cache=128 (bottom row, left). The most interesting part is the *tail*, which shows how many ex-
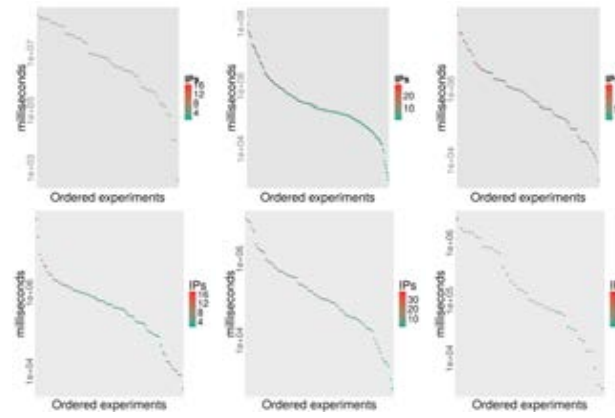

Figure 5: Duration of experiments vs. rank of experiments sorted by descending duration, with y axis in a logarithmic scale. Dot color is related to the number of IPs participating in the experiment. From left to right and top to bottom, experiments 4/4, 4/24 and 7/31 and caches=128, 64, 32.

periments took a desirable amount of time, on the order of 10 seconds, and which appears in all three graphs. As it can be seen, it sharply drops implying there are just a few of them, and with diminishing probability as time decreases. However, since they have a greenish color, implying a low number of IPs, they might be due to clients *carrying over* from the previous one. This is a characteristic of this implementation which will be examined later on, but at any rate, if we discard those experiments that take too much or too little, there is a decreasing exponential distribution that corresponds to the Zipf's law.
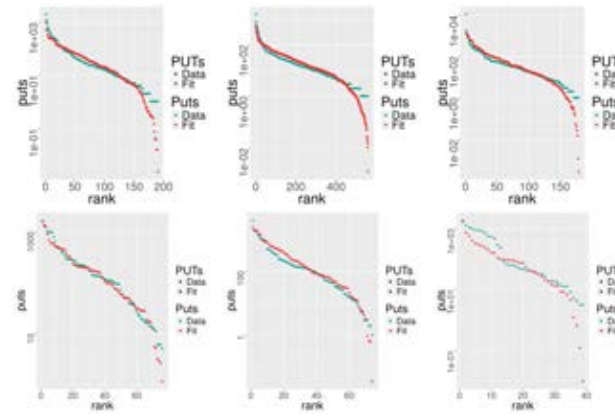

Figure 6: Number of PUTs per unique IP and fit to a Weibull distribution (in red); axis x shows IPs sorted by descending number of PUTs. From left to right and top to bottom, experiments 4/4, 4/24 and 7/31 and new experiments with cache=128, 64, 32.

A similar exponential distribution also appears if we rank HTTP PUTs, equivalents to the number of generations di-

Table 3: Weibull distribution parameters of the fit of the number of `PUTs` per unique IP.

Experiment	Scale σ	Shape ξ
4/4	43.07 ± 5.80	0.57 ± 0.03
4/24	22.97 ± 1.57	0.66 ± 0.02
7/31	53.18 ± 7.77	0.54 ± 0.03
Cache 128	205.28 ± 32.10	0.77 ± 0.07
Cache 64	178.99 ± 36.44	0.60 ± 0.05
Cache 32	168.15 ± 49.28	0.57 ± 0.07

vided by 100, or to evaluations divided by 12800, contributed by every user, which is shown in Figure 6. These results show a Zipf-like behavior, that is, a power law with a small *bump* in the lowest values. After testing the Generalized Extreme Value distribution and failing for the new batch of experiments, we have fitted it to a Weibull distribution Thoman et al. (1969) with the resulting parameters shown in Table 3. The inverse Weibull distribution is a special case of the GEV distribution in those papers, and appears usually in natural sciences and artificial life, usually related to decay. It has been frequently fitted to volunteer computing frameworks such as SETI@home (Javadi et al., 2009). The model that user behavior follows can be explained straighforwardly: when users visit the page, it draws their attention for a limited amount of time. They give it a chance for a few seconds. If something there amuses them or they can engage in a conversation about it, they stay for a while longer, otherwise, they leave. The *scale* parameter, which is around 20-40 in the first batch for the 40 Trap problem and between 160 and 200 in the 60 traps problem, depends mainly on the maximum number of generations people leave it running. Since it finishes or stalls after a number of generations shown in Figure 2, volunteers just leave after that. Curiously enough, the scale parameter is roughly twice the median number of `PUTs` per experiment, showing that, on average, the most loyal users reload the page twice after finishing or after seeing the evolution does not progress. This rule of thumb breaks down with the last experiment, however, which is interesting by itself, too.

The slope or shape parameter, on the other hand, indicates the overall shape of the curve. A value less than 1 indicates a concave (in a non-algorithmic scale) curve, with figures closer to one indicating a smaller slope. In all cases values are between 0.54 and 0.77, independently of the experiment. It might be the case that this number depends more on the total number of experiments carried out, with sets with more experiments, both in the middle, having values between 0.60 and 0.70. The distribution is remarkably similar which gives us a model of user behavior that is, to a certain extent, independent of the experiment.

These experiments show that, as it was proved for other volunteer computer frameworks and also in the case of

games, user engagement follows a Weibull distribution. This makes engagement the key for leveraging the performance of the socio-technical metacomputer and a way to improve results in the future.

Conclusion

Our intention in this paper was to assess the capabilities of a socio-technical system formed by a client-server web-based framework running a distributed evolutionary algorithms and the volunteers that participate in the experiment. These volunteers are *in the cloud*, that is, available as *CPU as a service* for the persons running the experiment. In this paper we have tried to put some figures on the real size of that *cloud* and how it can be used standalone if there is no alternative, or, if other computing resources are available, in conjunction with other local or cloud-based methods to add computing power in a seamless way through the pool that NodIO creates.

After running the experiment on the 40-trap problem whose running time could be as low as a few seconds, we switched to another batch of experiments where we used the 50-trap problem and also to a pool of limited, and dwindling through the three experiments, size. Since our initial results indicated that what happened on the screen, a flat graph with no improvements or the experiment finished, influenced the amount of time that the users devoted to the experiment, a longer one could yield different results and, at the same time, result in a big pool that might either crash the server or return useless individuals to the volunteers. These new experiments have proved that using the limited pool is beneficial to finding the solution, since less evaluations are needed, but also that since the problem is more difficult, the users stay for longer in the web page, making less ephemeral the socio-technical computing system created by the simulation.

The second objective of this paper was to model the user behavior in a first attempt to try and predict performance. As should be expected, the model depends on the implementation, with contributions following a Weibull distribution, which reflects the fact that volunteer computing follows a model quite similar to that found for games or other online activities. The reverse might be true: if we want to have returning users for the experiments, it is probable that we should *gamify* the experience so that once they've done it once, they might do it more times. In the spirit of Open Science, this gamification might involve computing in real time data such as the one presented in this paper and showing it in the same page or presenting user results alongside others.

In general, linking and finding correlations between user choices and performance is an interesting avenue to explore in the future. Even if these experiments were published in a similar way, one obtained up to five times more total cycles than the one with the least number of cycles. It is also essential to obtain volunteers as fast and simultaneously as possible, so it is possible that the features of the social network in

terms of real-time use will also play a big role; synchronous webs such as Snapchat, which mystifies the writers of this paper, and Twitter, thanks to its real time nature, might be better suited than Facebook, LinkedIn or Google plus. Even as it is difficult to create controlled experiments in this area, it is an interesting challenge to explore in the future.

The other area to explore is the algorithmic area itself. Are there ways to change the evolutionary algorithm, or its visualization, so that the user has a bigger impact on the result? One of the users in Twitter even suggested to embed videos so that people spent time looking at them, but other possible way was to make the user engage the algorithm by giving him or her buttons to change the mutation rate when the algorithm is stalled, for instance. If this is combined with a score board where local performance is compared to other users, engagement might be increased and thus the performance of the system. In general there are many issues with the evolutionary algorithm implementation itself, including using different, or adaptive, policies for inserting and sending individuals to the pool, using different policies for population initialization, and also the incorporation of high-speed local resources to the pool to check what would be the real influence of the volunteer pool to the final performance.

Finally, the implementation needs some refinement in terms of programming and also ease of use. Tools such as Yeoman for generating easily new experiments might be used, so that the user would have to create only a fitness function, with the rest of the framework wrapped around automatically.

All these avenues of experimentation will be done openly following the Open Science policy of our group, which, in fact, contributes to establish trust and security between us and volunteers and is an essential feature of the system. That is why this paper, as well as the data and processing scripts, are published with a free license in GitHub at `https://github.com/JJ/modeling-volunteer-computing`.

Acknowledgments

References

Ackley, D. H. (1987). *A connectionist machine for genetic hillclimbing*. Kluwer Academic Publishers, Norwell, MA, USA.

Anderson, D. P., Cobb, J., Korpela, E., Lebofsky, M., and Werthimer, D. (2002). SETI@home: an experiment in public-resource computing. *Commun. ACM*, 45(11):56–61.

Apolónia, N., Ferreira, P., and Veiga, L. (2012). Enhancing online communities with cycle-sharing for social networks. In *Computational Social Networks*, pages 161–195. Springer.

Boldrin, F., Taddia, C., and Mazzini, G. (2007). Distributed computing through web browser. In *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*, pages 2020–2024. IEEE.

Chong, F. S. and Langdon, W. B. (1999). Java based distributed Genetic Programming on the internet. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., and Smith, R. E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, page 1229, Orlando, Florida, USA. Morgan Kaufmann. Full text in technical report CSRP-99-7.

Desell, T., Szymanski, B., and Varela, C. (2008). An asynchronous hybrid genetic-simplex search for modeling the Milky Way galaxy using volunteer computing. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, GECCO '08, pages 921–928, New York, NY, USA. ACM.

Duda, J. and Dłubacz, W. (2013). Distributed evolutionary computing system based on web browsers with JavaScript. In *Applied Parallel and Scientific Computing*, pages 183–191. Springer.

Gonzalez, D. L., de Vega, F. F., Trujillo, L., Olague, G., de la O, F. C., Cardenas, M., Araujo, L., Castillo, P. A., and Sharman, K. (2008). Increasing GP computing power via volunteer computing. *CoRR*, abs/0801.1210.

González Lombraña, D., Laredo, J. L. J., Fernández de Vega, F., and Merelo Guervós, J. J. (2010). Characterizing fault-tolerance of genetic algorithms in desktop grid systems. In *Evolutionary Computation in Combinatorial Optimization*, pages 131–142. Springer.

Javadi, B., Kondo, D., Vincent, J.-M., and Anderson, D. P. (2009). Mining for statistical models of availability in large-scale distributed systems: An empirical study of SETI@home. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems, 2009. MASCOTS'09. IEEE International Symposium on*, pages 1–10. IEEE.

Jin, H., Luo, F., Liao, X., Zhang, Q., and Zhang, H. (2006). Constructing a P2P-based high performance

computing platform. In *2006 International Workshop on P2P for High Performance Computational Sciences (P2P-HPCS06)*, volume 3994 of *LECTURE NOTES IN COMPUTER SCIENCE*, pages 380–387. Springer.

Klein, J. and Spector, L. (2007). Unwitting distributed genetic programming via asynchronous JavaScript and XML. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1628–1635, New York, NY, USA. ACM.

Langdon, W. B. (2005). Pfeiffer – A distributed open-ended evolutionary system. In Edmonds, B., Gilbert, N., Gustafson, S., Hales, D., and Krasnogor, N., editors, *AISB'05: Proceedings of the Joint Symposium on Socially Inspired Computing (METAS 2005)*, pages 7–13, University of Hertfordshire, Hatfield, UK. SSAISB 2005 Convention.

Laredo, J. L. J., Bouvry, P., González, D. L., de Vega, F. F., García-Arenas, M., Merelo-Guervós, J. J., and Fernandes, C. M. (2014). Designing robust volunteer-based evolutionary algorithms. *Genetic Programming and Evolvable Machines*, 15(3):221–244.

Laredo, J. L. J., Castillo, P. A., Mora, A. M., Fernandes, C., and Merelo, J. J. (2008a). Addressing Churn in a Peer-to-Peer Evolutionary Algorithm. In *WPABA'08 - First International Workshop on Parallel Architectures and Bioinspired Algorithms, Toronto, Canada*, pages 5–12. Complutense University Of Madrid.

Laredo, J. L. J., Castillo, P. A., Mora, A. M., Fernandes, C. M., and Merelo, J. J. (2008b). Resilience to churn of a peer-to-peer evolutionary algorithm. *International Journal of High Performance Systems Architecture*, 1(4):260–268.

Martınez, G. J. and Val, L. (2015). Capataz: a framework for distributing algorithms via the World Wide Web. *CLEI Electronic Journal*, 18(2):1.

Merelo, J. J. (2015). Low or no cost distributed evolutionary computation. In Camacho, D., Braubach, L., Venticinque, S., and Badica, C., editors, *Intelligent Distributed Computing VIII*, volume 570 of *Studies in Computational Intelligence*, pages 3–4. Springer International Publishing.

Merelo, J. J., Castillo, P., Laredo, J., Mora, A., and Prieto, A. (2008). Asynchronous distributed genetic algorithms with JavaScript and JSON. In *WCCI 2008 Proceedings*, pages 1372–1379. IEEE Press.

Merelo, J. J., García, A. M., Laredo, J. L. J., Lupión, J., and Tricas, F. (2007). Browser-based distributed evolutionary computation: performance and scaling behavior. In

GECCO '07: Proceedings of the 2007 GECCO conference companion on Genetic and evolutionary computation, pages 2851–2858, New York, NY, USA. ACM Press.

Merelo, J.-J., García-Sánchez, P., García-Valdez, M., and Blancas, I. (2015). There is no fast lunch: an examination of the running speed of evolutionary algorithms in several languages. *ArXiv e-prints*.

Merelo, J.-J., García-Valdez, M., Castillo, P. A., García-Sánchez, P., de las Cuevas, P., and Rico, N. (2016). NodIO, a JavaScript framework for volunteer-based evolutionary algorithms : first results. *ArXiv e-prints*.

Merelo-Guervós, J.-J. and García-Sánchez, P. (2015). Designing and modeling a browser-based distributed evolutionary computation system. In Laredo, J. L. J., Silva, S., and Esparcia-Alcázar, A. I., editors, *Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11-15, 2015, Companion Material Proceedings*, pages 1117–1124. ACM.

Merelo-Guervós, J.-J., Mora, A.-M., Fernandes, C.-M., Esparcia-Alcázar, A.-I., and Jiménez-Laredo, J.-L. (2012). Pool vs. island based evolutionary algorithms: An initial exploration. In Xhafa, F., Barolli, L., and Li, K. F., editors, *3PGCIC*, pages 19–24. IEEE.

Milani, A. (2004). Online genetic algorithms. Technical report, Institute of Information Theories and Applications FOI ITHEA.

Nogueras, R. and Cotta, C. (2015). Studying fault-tolerance in island-based evolutionary and multimemetic algorithms. *Journal of Grid Computing*, pages 1–24.

Stutzbach, D. and Rejaie, R. (2006). Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 189–202. ACM.

Thoman, D. R., Bain, L. J., and Antle, C. E. (1969). Inferences on the parameters of the weibull distribution. *Technometrics*, 11(3):445–460.

Vespignani, A. et al. (2009). Predicting the behavior of techno-social systems. *Science*, 325(5939):425.

Wang, X. and Xu, S. (2008). P2HP: Construction of a cooperative server group based volunteer computing environment. In *International Conference on Internet Computing in Science and Engineering*, volume 0, pages 389–395, Los Alamitos, CA, USA. IEEE Computer Society.

A Bio-Inspired Artificial Agent to Complete a Herding Task with Novices

Patrick Nalepka[1], Maurice Lamb[1], Rachel W. Kallen[1], Kevin Shockley[1], Anthony Chemero[1] and
Michael J. Richardson[1]

[1]Center for Cognition, Action and Perception, University of Cincinnati, OH, USA 45221
nalepkpk@mail.uc.edu

Abstract

Models of robust human-human coordination can guide the design of adaptive and responsive human-robot systems. Here we test an artificial agent that embodies low-dimensional nonlinear dynamic equations derived from human behavior while completing a two-agent herding task, where the goal is to contain reactive spheres to the center of a target region. The model was able to complete the task alongside human novices in a virtual version of the experimental setup used in Nalepka and colleagues (submitted). Not only did the model lead participants to successful performance, but also 12 out of 18 participants reported that they believed their partner was a human participant in another room. The model was therefore able to capture the complex social behavior that defined robust task success in terms of lower dimensional dynamical equations that characterizes the emergent behavioral dynamics of embedded multiagent behavior.

Introduction

Joint human coordination is adaptive and robust – pairs of individuals can work together to complete tasks, such as carrying a couch, through a variety of different environments without much difficulty. Developing robotic systems that have these qualities will be useful for problems that need to be solved in a wide range of environments and contexts, with both human and robot partners. One possibility for creating such adaptive systems is provided by models of the low-dimensional behavioral dynamics (Warren, 2006) that emerge when human pairs complete tasks successfully. Because these models consist of a handful of ordinary differential equations, they can be used to drive the behavior of socially embedded artificial systems and exhibit a wide range of behavioral complexity in a relatively compact form.

Our approach (Richardson et al., 2015; in press) is based on research in cognitive science inspired by dynamical systems theory and complexity science. According to this research, the local actions and behaviors of relatively simple systems can give rise to complex coordinative behavior. Fairly well known examples of this research in non-human systems include the coordination of fish schools and bird flocks, where simple rules

at the local or individual level lead to complex coordinated movement (Couzin, Krause, Franks, & Levin, 2005; Hildenbrandt, Carere, & Hemelrijk, 2010; Reynolds, 1987). Applied to psychology, this has been used to explain path navigation and obstacle avoidance (Fajen & Warren, 2003; 2004; 2007), rhythmic movement coordination (e.g., Richardson et al., 2007; Schmidt & O'Brien, 1997; Schmidt & Turvey, 1994), and anticipation (Washburn et al., 2015). In the case of path navigation and obstacle avoidance, only a small set of nonlinear equations is required to accurately model and predict human movement towards a goal through a cluttered environment (Warren & Fajen, 2003; Warren, 2006). In this model the goal is treated as a point-attractor and obstacles, such as desks, chairs, or other people, are treated as repellors. Fajen and Warren's model has been implemented successfully as the basis for an artificial navigation system (Nemec & Lahajnar, 2009). Similar work has been done to extend this methodology to passing objects between partners in order to reach a goal location (Lucas et al., 2015). In the rhythmic coordination research, artificial agents have been developed to coordinate and produce new stable coordination patterns with partners (Kostrubiec, Dumas, Zanone, & Kelso, 2015; Zhang, Dumas, Kelso, & Tognoli, 2016). However, research is needed to extend this approach to understand and model the coordinative dynamics that emerge in more complex situations that require the control of a dynamically changing environment. In the remainder of the paper, we'll describe and test an artificial agent able to work with novices to contain a set of autonomous agents as an illustration of how these types of situations can be understood.

Behavioral Task

A two-player shepherding game was developed in Nalepka and colleagues (submitted) to understand emergent coordination dynamics in more complex and dynamically changing environments.[1] In the task, dyads stood on either side of a frosted glass tabletop (see Figure 3A) and controlled cube-shaped agents (sheepdogs) with wireless handheld motion sensors tracked using a Polhemus Liberty Latus motion tracking system. Participants interacted in a two-dimensional

---

[1] The task was inspired by shepherding behavior in dogs, but is not set out to explain actual shepherding behavior (see Strömbom et al., 2014), but rather was referred to as such as a way to describe the task to participants.

environment projected onto the glass top (see Figure 1). The environment consisted a fenced grass area and three or seven balls (sheep). The goal of the dyad was to cooperate and discover a solution to contain all the sheep within the red target containment area for a certain length of time and for a certain amount of trials before time ran out (45 minutes). The sheep could move freely within the fenced playing field, but could not pass beyond the fenced region. If the sheep collided with the fence, the trial ended prematurely. If the sheep collided with one another they acted like solid objects and could not pass through one another. When left alone the sheep exhibited Brownian motion, but when a player's sheepdog was within 12 cm of a sheep, the sheep reacted to the players' proximity by moving in the direction opposite of the participant's position. Therefore, participants needed to learn how to push the sheep towards the center of the field without moving too close as to accidently cause the sheep to scatter away from the participant. Participants were not allowed to explicitly communicate with one another regarding their game play or strategy while engaged in the task.
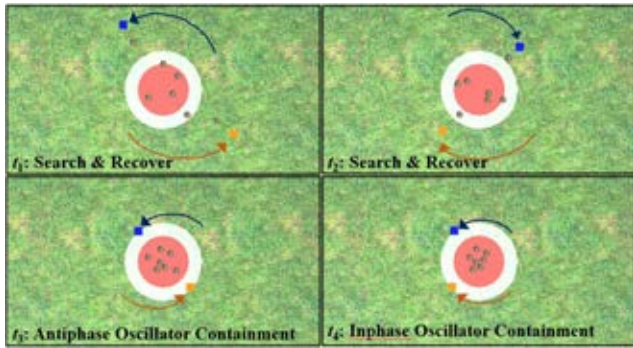

Figure 1: A prototypical time lapsed ($t_1...t_4$) depiction of game play, including an illustration of the search-and-recover (S&R) and coupled oscillatory contained (COC) behavioral modes exhibited by pairs. Arrows indicate movement direction.

Only 31 of 42 tested pairs were able to meet the success criteria, and all but two of them discovered the same coordinative strategy to corral the sheep. The dominant coordinative strategy consisted of two different modes of behavior. We will call these modes the *Search & Recover* (S&R) strategy, depicted in Figure 1t_{1-2} and the *Coupled Oscillator Containment* (COC) strategy, depicted in Figure 1t_{3-4} respectively. When employing the S&R mode of behavior, participants implicitly divided the task space in half, and each individual in the dyad pursued the sheep furthest from the center of the target region, placing their respective marker such that it propelled the farthest sheep on their respective sided towards the center. As the sheep clustered, an oscillatory movement of the sheepdogs began to emerge as the result of participants moving to similar regions in of the playing field to contain the sheep. After some time, a bifurcation occurred and participants begin to make regular antiphase (Figure 1t_3) or inphase (Figure 1t_4) oscillatory movements. This is the emergence of the COC strategy. Once this COC behavioral mode emerged, dyads would begin subsequent trials by moving directly into this behavioral mode. It was found that this

oscillatory behavior is consistent with the dynamics produced by coupled nonlinear oscillators, which have also been used to understand the dynamics of intra- and interpersonal rhythmic movement (e.g., Richardson et al., 2007; Schmidt et al., 1990; Schmidt & O'Brien, 1997; Schmidt & Turvey, 1994). These dynamics, as described by the Haken, Kelso and Bunz model (Haken et al., 1985; Schöner et al., 1986), indicates that two stable oscillatory modes are possible: inphase and antiphase behavior, with inphase behavior being more stable than antiphase. This relative stability is consistent with the experimental results and suggests that the dyads may be embodying similar dynamics when performing the COC behavior.

The Artificial Shepherd

Given the results in Nalepka and colleagues (submitted) we developed an expert artificial agent (EAA) based on a model, visually depicted in Figure 2, which characterizes the two behavioral modes present in past human data (chosen values will be presented in parentheses after each parameter). S&R involved participants selecting a sheep that is furthest from the red target region, and moving their sheepdog so that the targeted sheep is in a line between the player and the central region. This behavior can be modeled with two sets of equations below:

$$\ddot{r}_i + b_{ri}\dot{r}_i + \varepsilon_i\left(r_i - (\varphi_{sd,i(t)} + C_{md,i})\right) = 0 \qquad (1)$$

$$\ddot{\theta}_i + b_{\theta i}\dot{\theta}_i + \mu_i\left(\theta_i - \varphi_{s\theta,i(t)}D_{sd,i}\right) = 0 \qquad (2)$$

The model operates on a polar task axis where r_i is the radial distance of the player's sheepdog from the center of the field, with $\dot{r}$ and $\ddot{r}_i$ being its velocity and acceleration along that axis, respectively. Parameter θ_i is the angle, in radians, from the reference angle which is on the sagittal plane, with $\dot{\theta}_i$ and $\ddot{\theta}_i$ being the angle's velocity and acceleration terms. The subscript i indicates player identification. Parameters b_{ri} (5) and $b_{\theta i}$ (5) are the velocity damping terms, ε_i (20) and μ_i (15) scales the rate at which player i minimizes the difference between their current position to the target radial position, $\varphi_{sd,i(t)}$ and angle, $\varphi_{s\theta,i(t)}$ of the target sheep, $\varphi_{s(t)}$. The parameter $C_{md,i}$ (0.05) is a fixed value which indicates the minimal distance that the agent should stay away from the sheep. This parameter prevents the EAA from moving on top of the target sheep, as well as too close into the central containment region, preventing unreliable sheep repulsion behavior. The parameter $D_{sd,i}$ is a Heaviside parameter defined as,

$$D_{sdi} = \begin{cases} 0, & \varphi_{sd,i(t)} \leq C_{sd,i} \\ 1, & \varphi_{sd,i(t)} > C_{sd,i} \end{cases} \qquad (3)$$

where $D_{sd,i}$ is zero when the farthest sheep is less than a fixed parameter $C_{sd,i}$, (0.08) indicating there is no furthest sheep to corral and so the EAA moves towards the reference angle axis. It should be noted that the model only pursues sheep that are on
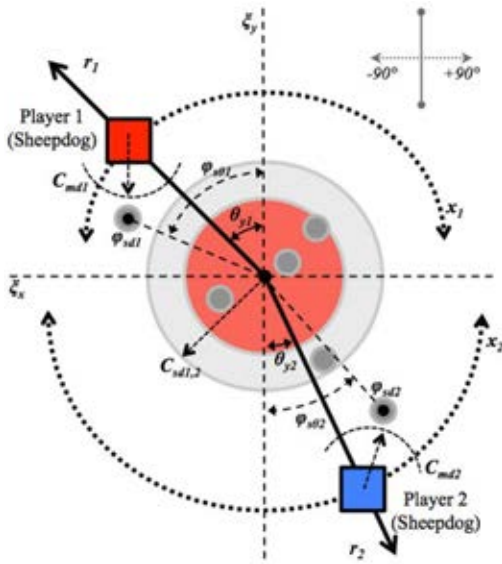
Figure 2: Illustration of the task space axes employed for the bi-agent herding model. Illustration of the task space employed for the sheep-herding model which captures player i's (where $i = 1$ or 2) sheepdog location at any time within the ξ_x and ξ_y planar game space in polar coordinates (r_i, θ_i), with r_i corresponding to the radial distance of player i from the center of the ξ_x and ξ_y planar game space and θ_i corresponding to player i's radial angle ($\pm 90°$) from the ξ_y game space axis on their side of the game field. x_i is the oscillatory perimeter path of each participant i's sheepdog with respect to half (π-radians) of the target containment region of success closest to the participant's side of the game space, such that each participant's perimeter path, x_i, is centered on the participants radial (r_i, θ_i) position within the (ξ_x, ξ_y) game space. Adapted from Nalepka and colleagues (submitted) with permission from the authors.

their half of the playing field, which is consistent with data found in Nalepka and colleagues (submitted).

When a certain level of proficiency is achieved, players transition from the S&R behavior to the COC mode of behavior whose global behavioral dynamics is consistent with the HKB model (Haken et al., 1985). To be consistent with the previous research modeling the dynamics of rhythmic human limb movements (Kay et al., 1987), and rhythmic interlimb and interpersonal coordination (see Haken et al., 1985; Kelso, 1995; Schmidt & Richardson, 2008), the COC mode of behavior was modeled using a set of coupled Rayleigh/van der Pol hybrid nonlinear oscillators of the form,

$$\ddot{x}_1 + \alpha_1 \dot{x}_1 + \beta_1 x_1^3 + \gamma_1 x_1^2 \dot{x}_1 + \omega_1^2 x_1$$
$$= (\dot{x}_1 - \dot{x}_2)(A + B(x_1 - x_2)^2)$$

$$(4)$$

$$\ddot{x}_2 + \alpha_2 \dot{x}_2 + \beta_2 x_2^3 + \gamma_2 x_2^2 \dot{x}_2 + \omega_2^2 x_2$$
$$= (\dot{x}_2 - \dot{x}_1)(A \mp B(x_2 - x_1)^2)$$

where $i = 1$ for player 1 and $i = 2$ for player 2, and the positive/negative (excitatory/inhibitory) damping parameter α_i scaled as a function of sheep distance using the linear equation,

$$\dot{\alpha}_i = \delta_i(C(\varphi_{sd,i(t)}^2) - D(C_{sd,i}) - \alpha_i) \qquad (5)$$

Parameter x_i indicates the player's position on the circular x task axis, centered on the player's (r_i, θ_i). The possible range of motion is roughly $(\theta_i \pm \frac{\pi}{2})$, with $\dot{x}_i$ and $\ddot{x}_i$ being its velocity and acceleration along x task axis. Note that x_i in Eq. 4 outputs a range of values roughly $(-1, 1)$ which is then multiplied by $\frac{\pi}{2}$ to convert the value into game space units after iteration. The parameter ω_i ($\frac{7}{4}\pi$) defines the stiffness or frequency of oscillator of player i's perimeter path movement, and the functions $(\beta_i x_i^3)$ (1) and $(\gamma_i x_i^2 \dot{x}_i)$ (5) corresponding to the Rayleigh and van der Pol escapements terms for each player's

x_i perimeter path movement, respectively. The coupling function to the right of the equals sign in each equation is the same as that previously derived by Kelso and colleagues (e.g., see Haken et al., 1985; Kelso, 1995), and defines both inphase (0°) and antiphase (180°) relative phase relationships as the stable coordination modes between the two oscillators (when $\alpha_i < 0$), with the relative strength of these two coordination modes defined by the parameters A and B (both -0.2). The system is bi-stable when $|4B| > |A|$, but mono-stable (inphase only) when $|4B| < |A|$.

The change between the two modes of behavior, S&R and COC, is the Hopf bifurcation in the variable x_i that occurs for each oscillator system. The bifurcation is driven by Eq. 5: when $\alpha_i > 0$, behavior along the x_i task axis corresponds to that of a nonlinear mass damped spring with a stable fixed point solution; when $\alpha_i < 0$, behavior along the x_i task axis corresponds to that of a nonlinear limit cycle oscillator, with an amplitude of movement approximately equal to

$$x_{i,AMP} = 2\sqrt{|\alpha_i|/\gamma_i} \qquad (6)$$

when $\alpha_i < 0$, $\beta_i > 0$ $\gamma_i > 0$ and $|\alpha_i| \ll \omega_i$ (see Kay et al., 1987 for more details). The value α_i in Eq.5 at any instance in time, (t), is a differential function of the distance, $\varphi_{sd,i(t)}$, of the furthest sheep on player i's side of the game space with respect to a maximum safe containment distance, $C_{sd,i}$, with parameter δ_i (25) which controls the rate at which $\dot{\alpha}_i$ reaches 0. Parameters C (9) and D (8) adjusts the weight given to $\varphi_{sd,i(t)}^2$ and $C_{sd,i}$ in determining $\dot{\alpha}_i$. If the distance, $\varphi_{sd,i(t)}$, of the sheep furthest from the center of the game space on player i's side of the game space is outside player i's maximum safe containment distance, $C_{sd,i}$, and $\alpha_i > 0$, then behavior along the x_i task axis corresponds to that of a nonlinear damped mass spring. Conversely, if the distance, $\varphi_{sd,i(t)}$, of the sheep furthest from the center of the game space on player i's side is inside player i's maximum safe containment distance, $C_{sd,i}$, and $\alpha_i < 0$, then behavior along the x_i task axis corresponds to that of a nonlinear limit cycle oscillator. Values near zero produce

minute scrubbing behavior by the agent which, anecdotally, is consistent with past human data.

After piloting, changes to how the target sheep, $\varphi_{s(t)}$, is determined were made from the model's original formulation in Nalepka and colleagues (submitted). The following Heaviside function,

$$\varphi_{s(t)} = \begin{cases} \min_{j \in \Phi} \Phi_{dj}, & \alpha_i > \tau \\ \max_{j \in \Phi} \Gamma_{dj}, & \alpha_i \leq \tau \end{cases} \quad (7)$$

determines which sheep the agent will pursue. Value Φ_{dj} represents the distance for a given sheep j, to the closest fence segment. Γ_{dj} represents the radial distance of sheep j from the center of the target region, and τ (0.5) represents a fixed parameter determining when the sheep selection strategy changes. Parameter τ is indirectly derived from a specific distance from the center of the field, but is more intuitively operationalized as being associated with a change in the mode of behavior, with $\min_{j \in \Phi} \Phi_{dj}$ used when engaging in S&R behavior, and $\max_{j \in \Phi} \Gamma_{dj}$ when utilizing COC behavior. Equation 7 was included because, due to the rectangular playing field, the EAA was placed in situations where it would pursue a sheep that, although further from the center, caused trials to fail as sheep closer to the center would hit the fence on the short end of the rectangle. This post-hoc method was included as to prevent the model from continuing letting their partner down. Additionally, the function that determines the target sheep was altered to select the position of the target sheep + 1% of its normalized velocity vector. This was done to prevent situations where the model would cause the target sheep to spiral around the target region due to the sheep maintaining its tangential velocity.

Current Study

The present research set out to test and validate the above model against a set of novice participants. For the current study, the coupling term in Eq. 4 was not included. This was done in order to determine if the COC behavior would emerge without the explicit coupling to the participant. The study had three aims: 1) to determine whether the EAA can perform the herding task with a human novice, 2) to determine whether novices learn to coordinate with the artificial agent to produce COC behavior consistent with data presented in Nalepka and colleagues (submitted) and 3) determine whether participants will remain in belief that they are performing the task alongside a human agent as a way to test the *humanness* of the model.

Method

Participants

Eighty participants took part in the study. Thirty pairs were formed in the novice control condition, and the remaining 20 completed the EAA condition. All participants received research credit as part of a class requirement for an undergraduate Psychology course.

Apparatus & Task

The task was designed using the Unity 3D game engine (version 5.2.0; Unity Technologies, San Francisco, California) and was presented to participants via Oculus Rift DK2 (VR) headsets (Oculus VR, Irvine, California). The presented virtual environment (Figure 3B) was modeled at 1:1 scale after the experimental room (Figure 3A). The task was presented in the VR headset to appear on a virtual tabletop modeled at 1:1 scale after the glass tabletop in the real environment which then acted as the solid physical surface for participants to move their motion sensors on. Participants used wireless Latus motion tracking sensors operating at 96 Hz (Polhemus Ltd, Vermont, USA) Participants moved the sensor along the glass tabletop and hand movements translated 1:1 to movements of the player's cube (sheepdog) in the virtual environment. Participants were given a body in the virtual world, modeled after a crash test dummy of height 1.8m whose motion was controlled using an inverse kinematic calculator (model and calculator supplied by Root Motion, Tartu, Estonia) based off the real movements of the participant's right hand (via a Latus motion sensor) and head (via the Oculus Rift). Regardless of height, each participant was calibrated to fit the body of the virtual model, giving each participant an equivalent image size of the playing area. The reason for virtual bodies was threefold: to best emulate the conditions in the original research by Nalepka and colleagues (submitted) where participants' arms were able to occlude the playing surface, to provide approximate information regarding the arm location of one's partner (to avoid hitting), and to further increase the believability that participants were interacting with a human in the EAA condition. Separate computers were used to power the headsets and data was transferred to the host computer via a LAN connection. The maximum display latency between the participants' real-time movements and the virtual (box) sheepdog was 33 ms. Game states were continuously recorded at 50 Hz, including the movements of the virtual sheep and the participant controlled sheepdogs.

Participants were able to move their sheepdogs anywhere in two dimensional space within the 1.17m by 0.62m fenced area of the grass task field. The goal of the game was to contain 3 or 7 wool covered stimulus balls (sheep) within the red circular target containment region measuring 19.2 cm in diameter for 70% of the last 45 seconds of a 60 second game trial (the first 15 seconds of each trial served as time for participants to initiate a behavioral coordination strategy and/or corral the sheep). All sheep needed to be inside this region for it to count towards the dyad's score. Participants received visual feedback regarding their performance at the end of completed trial (i.e., what percentage of time they managed to keep the sheep within the target area). A game trial ended prematurely (i.e., before 60 seconds), however, if one of the sheep managed to hit the perimeter fence or if all sheep escaped the 29 cm white circle that surrounded the red target region circle. At the start of a trial, the sheep were distributed within the red target containment region (Figure 3c-d), with the subsequent motion of the sheep governed by random Brownian motion dynamics. The sheep also dynamically reacted to the participant controlled sheepdogs as if threatened, being repelled away from

a participant's sheepdog when the sheepdog was within 12 cm of the sheep's game location. When threatened, the sheep would move directly away from the player at a speed proportional to the inversed of the squared distance between the sheep and the player. It is also important to note that sheep were programmed to be able to collide (as opposed to pass through) each other. Finally, pairs played the game for a maximum of 45 minutes, with the experiment ending either at the end of this experimental period or earlier if the pairs made it past the 70% success threshold eight times.
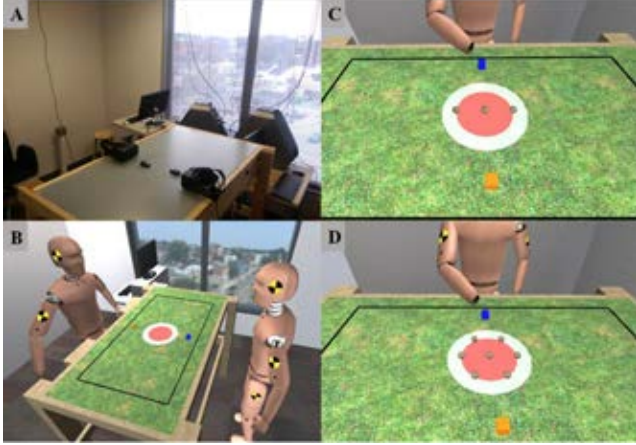

Figure 3: (A) Experimental Room, (B) Virtual replica of (A) presented to participants. (C-D) Depiction of the initial arrangement of the 3 and 7-sheep conditions, respectively from the perspective of the participant.

In the novice control condition, participants stood on either side of the table as depicted in Figure 3B. In the EAA condition, participants were told that their partner had come early and was set up to complete the task remotely in a room next door. In the EAA condition, the model controlled its respective sheepdog via the model presented above. To give additional realism, the EAA model body's head was programmed to linearly interpolate its gaze direction towards the target sheep position, $\varphi_{s(t)}$, with $\pm$ 2.5 cm movement noise. Additional noise was added to the model's radial ($\pm 1\frac{m}{s^2}$ to Eq. 1) and oscillatory movement ($\pm \frac{\pi}{2}\frac{rad}{s^2}$ to Eq. 4). In both conditions, participants were not able to see their partner in between trials so as to avoid the display of task-irrelevant behavior that was not incorporated into the EAA.

Procedure

Prior to arrival participants were randomly assigned to either the three or seven sheep condition. Following informed consent, during which time participants were told that they would be required to play a virtual shepherding game, participants were either lead into the testing room and were randomly assigned to opposite sides of the tabletop display (the novice control condition), or told that their partner had come early and will perform the task in a different room (the EAA condition). Each participant was then handed a wireless Latus motion tracking sensor and informed that they would be using

these motions tracking sensors to control their respective cube (sheepdog) in order to corral a set of balls (sheep) into the red target region of the grass game field. Participants were instructed to hold the motion sensors with their right hand and control their sheepdogs by sliding the sensors on top of the tabletop display. This ensured that the location and movement of their corresponding sheepdog was aligned with the motion tracking sensors. Participants were then shown the game field and the rules of the game were described (i.e., rules for trial success and failure detailed above). Importantly, no instructions about how to best play the game or how to coordinate or corral the sheep within the game region were provided. Participants were simply told to complete the task to the best of their ability. Participants in the novice control condition were told that they were not allowed to talk or verbally strategize at any time during the experimental session (neither within nor between trails). An experimenter was present during the experimental session to enforce this no-talking rule. After the experiment, participants were debriefed on the purpose of the study and participants in the EAA condition were asked questions regarding their interaction with their partner. The first question asked participants if they noticed anything odd in the experiment, the second question asked if they had a feeling that their partner has completed this task before, and the third question asked if they thought at any point in the experiment they doubted that they were completing the task with a human.

Results

The first aim seeks to determine if the EAA can complete the herding task alongside novices. For all analyses, one pair in the novice control condition and two participants in the EAA condition were excluded from analyses due to program malfunction. The remaining participants who met the winning criteria, which was to keep the sheep contained in the red central region for 70% of the remaining 45 seconds of the trial eight times, were kept for analyses. Twenty-three of the remaining 29 pairs (79.3%) in the novice control condition met the winning condition, and all 18 participants in the EAA condition met the winning condition. This confirms that the developed EAA is able to perform the task alongside human novices.

To investigate differences in performance between groups, several summary statistics and performance variables were considered. A 2 (Condition: control, EAA) x 2 (Sheep: 3, 7) between-subjects ANOVA was conducted on the amount of time dyads took to complete the experiment. A significant condition x sheep interaction was found, $F(1,37) = 7.75$, $p = 0.008$, $\eta^2 = 0.17$. For the novice control condition, a significant main effect was found for the number of sheep, $F(1,20) = 21.18$, $p < 0.001$, $\eta^2 = 0.50$, such that less time was taken to complete the 3-sheep condition (15.66 minutes) than the 7-sheep condition (29.07 minutes) ($p < 0.001$; all post-hoc tests in this paper use Bonferroni corrections). No significant difference was found in the EAA condition (3-sheep: 16.53 minutes, 7-sheep: 18.86 minutes; $p = 0.37$). Next, a 2 (Condition: control, EAA) x 2 (Sheep: 3, 7) between-subjects

		Inphase	Antiphase	Both-Phase	No/Other-Phase
Novice Control	3-Sheep	82.14% (4.29%)	5.36% (2.53%)	7.14% (3.13%)	5.36% (2.85%)
	7-Sheep	66.67% (9.77%)	16.67% (8.07%)	8.33% (2.95%)	8.33% (2.95%)
Artificial Agent	3-Sheep	48.44% (12.15%)	26.56% (9.28%)	1.56% (1.56%)	23.44% (8.33%)
	7-Sheep	42.50% (11.21%)	21.25% (5.29%)	12.50% (5.59%)	23.75% (6.57%)

Table 1: Proportion of relative phase modes observed during COC behavior. Note: Both-Phase corresponds to trials in which pairs produced significant periods of both inphase and antiphase coordination. No/Other-Phase represents data remaining that were not categorized under the first three labels. Standard error in parentheses.

ANOVA was conducted to see if there were differences in the amount of time dyads were able to keep the sheep in the red target region on successful trials, measured as the percentage of time within the last 45 seconds of each trial. A significant main effect on condition was found, $F(1,36) = 7.78$, $p = 0.008$, $\eta^2 = 0.18$, such that pairs in the novice control condition had a lower score on average (83.14%) than those who completed the task alongside the artificial model (87.80%) ($p < 0.008$). Not only were participants in the EAA condition able to keep the sheep in the red target region for a longer percentage of time, but the amount of time taken to complete the experiment was consistent across sheep herd conditions.

Another measure to compare game performance across groups involved analyzing the movement of the sheep. In this measure, differentiating what makes a dyad *better* at containing the sheep is operationalized as a dyad's ability to minimize the spread of the sheep and to minimize their movement from the center of the red target region. These qualities were measured by taking the normalized average area of the convex hull that spans over the sheep (measure of spread/sheep), as well as taking the average root mean square (RMS) of the sheep's position from the center (measure of variability from center). Separate 2 (Condition: control, EAA) x 2 (Sheep: 3, 7) between-subjects ANOVAs were conducted on each of these dependent measures. First, significant differences between testing conditions, $F(1,37) = 32.00$, $p < 0.001$, $\eta^2 = 0.46$, and sheep herd size, $F(1,37) = 5.71$, $p = 0.02$ $\eta^2 = 0.13$, on the average sheep spread were found, such that sheep in the EAA condition were less dispersed (3.80 cm^2/sheep) than in the control condition (5.52 cm^2/sheep) ($p < 0.001$), and that sheep in the 7-sheep condition took up more area (5.02 cm^2/sheep) than those in the 3-sheep condition (4.29 cm^2/sheep) ($p < 0.01$).

Second, significant main effects were found for the root mean square (RMS) of the sheep's distance from the center of the red containment region across condition, $F(1,37) = 18.79$, $p < 0.001$, $\eta^2 = 0.34$, and sheep herd size, $F(1,37) = 41.76$, $p < 0.001$, $\eta^2 = 0.53$, such that the average sheep RMS was lower in the EAA condition (3.59 cm) than the control condition (4.18 cm)($p < 0.001$) and that sheep in the 7-sheep condition had a lower RMS (3.44 cm) than in the 3-sheep condition (4.33 cm)($p < 0.001$). These results indicate that participants in the EAA condition were better able to both minimize the spread of the sheep, as well as to keep the sheep closer to the center of the red containment region. Along with the results above, this is most likely due to the EAA producing relatively consistent expert behavior across all participants and trials. Consistent with findings from Nalepka and colleagues (submitted), it is more difficult to keep the sheep in the center of the containment

region, as indicated by the RMS results, in the 3, as opposed to the 7-sheep conditions. The reason for this is due to the sheep-sheep collisions that are possible. A higher probability of collisions are possible in the 7-sheep condition, causing the formation of sheep clusters, which behave as small moving masses which are easier to control than individual sheep.

The second aim was to determine whether participants in the EAA condition are able to discover the COC mode of behavior as a successful strategy to contain the sheep, as well as whether these participants would reproduce the dynamics of nonlinear coupled oscillators, whose behavior produces globally stable in/antiphase behavior as modeled by the HKB model (Haken et al., 1985). To test this, relative phase analyses were conducted on the last 45 seconds of each successful trial. Time-series of each pairs' instantaneous relative phase using the Hilbert transform on the radial angle of each player's sheepdog to a reference axis was calculated (see Pikovsky et al., 2001 for details about this transformation). A 4th order Butterworth band-pass filter excluding frequencies under 0.375 Hz and above 5 Hz was done to exclude oscillations that happen on a timescale irrelevant to the task. Distributions of the absolute relative phase angles that occurred across six 30° regions (i.e., 0°-30°, 30°-60°… 150°-180°) of relative phase between 0° and 180° was calculated for each trial. For these distributions, inphase and antiphase coordination is indicated by a concentration of relative phase angles near 0◦ and 180◦, respectively (Schmidt & O'Brien, 1997; Richardson et al., 2005). In order to determine if a pair's distribution of relative phases were significantly inphase or antiphase, 1000 random relative phase time-series of sample length (45 sec) and sample rate (50 Hz) were created to generate 1000 surrogate random relative phase distributions. The 950th largest value for each 30° relative phase region—17.933%— was then employed as the statistical threshold value and corresponded to a 0.05 significance level (Varlet & Richardson, 2015). In/antiphase coordination was deemed to have occurred for a given trial if the percentage of occurrence of relative phase angles for the 0-30°/150-180° relative phase region was greater than the 17.933% threshold. In addition, intermittent (both) inphase and antiphase were noted to have occurred for a given trial if both the 0-30° and 150-180° regions were greater than the 17.933% threshold level. Table 1 provides a summary of the proportion of trials averaged across pairs that were statistically classified as inphase, antiphase, intermittent in/antiphase (both-phase), or no stable/other stable-phase relationship.

Inspection of Table 1 reveals that over 75% of trials can be classified by dyads in both the novice control and EAA condition as performing in/antiphase rhythmic coordination in

order to complete the task. Additionally, the ratio between inphase and antiphase trials are consistent with previous research on visual rhythmic coordination (e.g., Schmidt at al., 1990; Schmidt & O'Brien, 1997; Richardson et al., 2007), such that inphase coordination occurs more often than antiphase coordination.

A reason for a larger proportion of trials classified as No/Other-Phase for dyads in the EAA condition can be due to the lack of the coupling term for the EAA used in the present experiment. Although the EAA was unable to adjust its frequency to establish a stable inphase/antiphase relationship with its partner, participants nevertheless fell into stable in/antiphase coordination for a majority of trials, giving evidence that even in conditions with unidirectional coupling, participants are able to adjust their movement to the EAA's movements, allowing stable inphase/antiphase coordination to still be possible in most instances. However, if the EAA's inability to adjust its frequency to that of its partner is a reason for the higher percentage of No/Other-Phase trials in the EAA condition, then it is expected that a larger average oscillatory frequency difference would be observed in the EAA condition overall, as opposed to the novice control condition. A participant's oscillatory frequency was determined by first computing a spectral analysis on the z-scores of each participant's radial angle data. To remind the reader, this data was filtered using a 4th order Butterworth band-pass filter excluding frequencies under 0.375 Hz and above 5 Hz in order to only include oscillatory frequencies that are relevant to the COC behavior. The frequency with the most power for each participant and the absolute difference between each participant's peak frequency was used for further analyses. The absolute peak frequency difference for each dyad was then submitted to a 2(Condition: control, EAA) x 2 (Sheep: 3, 7) between-subjects ANOVA. A significant main effect of condition was found, $F(1,37) = 6.02$, $p = 0.02$, $\eta^2 = 0.14$, such that participants in the EAA condition had a greater frequency difference at their respective fundamental frequency (0.14 Hz) than those in the control condition (0.07 Hz), confirming the possibility that the greater frequency difference is responsible for the higher percentage of No/Other-Phase trials in the EAA condition.

The final aim was to determine whether participants would remain in belief that their partner was a human for the entire duration of the experiment. Debriefing in the EAA condition indicated that 12 out of 18 participants (66.67%) believed that their partner was a human for the entire duration of the experiment. Participants were mixed in their views if their partner was as naïve to the study as they were, or if their partner was given additional information about the task (or were a confederate). Some noted that their partner exhibited some "quirky" behavior or found it odd that the partner was able to discover the oscillatory strategy so quickly. Three of the remaining six had passing thoughts that their partner may in fact be a computer, while the remaining three had strong convictions that their partner was a computer for the entire duration of the experiment. In order to determine whether the difference in peak oscillation frequency is associated with believability that one's partner was a computer, a Pearson's correlation was conducted on participants in the EAA condition. The participants who had no thoughts that their partner could be non-human were placed in the *Believed Human* bin (12 individuals), while those participants who either had doubts or were confident that their partner was a computer were placed in the *Thought Computer* bin (6 individuals). The correlation was significant, $r(16) = 0.60$, $p = 0.009$, such that those who thought their partner was a human were associated with having lower peak frequency differences with their EAA partner than those who weren't fully convinced their partner was human, suggesting stronger unidirectional coupling for those in the *Believed Human* bin. It is remains an open question as to what aspects of the EAA made participants question or reject that they were performing the task with a human partner.

Discussion

The research presented in this paper represents developing first pass at developing a minimal bio-inspired artificial model that can serve as a partner in a two-player herding task. The current model demonstrates its ability to complete the task alongside participants while also convincing 12 of the 18 participants to report believing that their partner was a human for the entire duration of the experiment. Future work involves adding a coupling term to the model in order to give it the ability to modulate its oscillatory frequency to match the frequency of its partner. This may further enhance the realism of the model. However, it is interesting to note that even without the model keeping track of the partner's state, the unidirectional coupling of the participants still led to inphase, antiphase or both-phase coordination with the partner for over 75% of trials, showing the human tendency to form these stable phase relationships in the context of relatively complex task constraints and a dynamically changing environment. A reviewer expressed concern that the behavior the EAA exhibited is too prescribed to the specific environment presented in this paper, and may not be appropriate for environments where sheep need to be contained in a triangular or other shape containment regions. Future work will have to test this possibility. Another avenue of interest is to further elucidate the perceptual variables human agents attune to as they complete this task. The current model was built to mimic the behavioral dynamics observed in human hand movement data, without considering the role perception plays in the task.

The EAA algorithm implemented in the herding task provides an example of how modeling human behavioral dynamics can guide the design of socially embedded artificial agents. Note, the proposed model does not aim for the most efficient or optimal model, but the one that best matches the relevant human behavioral dynamic. This model identifies the low dimensional dynamics that drive the coordination behavior required to accomplish the task with another agent and is designed to strike aa balance between providing an exact detailed description of every aspect of a specific human agent (i.e., white-box modeling) and a black box model that might be implemented in any number of ways.

Acknowledgements

References

Couzin, I. D., Krause, J., Franks, N. R., & Levin, S. A. (2005). Effective leadership and decision-making in animal groups on the move. *Nature, 433*(7025), 513-516.

Fajen, B. R. & Warren, W. H. (2003). Behavioral dynamics of steering, obstacle avoidance, and route selection. *Journal of Experimental Psychology: Human Perception and Performance*, *29*(2), 343.

Fajen, B. R. & Warren, W. H. (2004). Visual guidance of intercepting a moving target on foot. *Perception*, *33*(6), 689-715.

Fajen, B. R. & Warren, W. H. (2007). Behavioral dynamics of intercepting a moving target. *Experimental Brain Research*, *180*(2), 303-319.

Haken H., Kelso J. A. S., & Bunz, H. (1985). A theoretical model of phase transitions in human hand movements. *Biological Cybernetics*. *51*, 347–356.

Hildenbrandt, H., Carere, C., & Hemelrijk, C. K. (2010). Self organized aerial displays of thousands of starlings: A model. *Behavioral Ecology*, *21*(6), 1349-1359.

Kay, B. A., Kelso, J. A. S., Saltzman, E. L. & Schöner, G. (1987). Space-time behavior of single and bimanual rhythmical movements: Data and limit cycle model. *Journal of Experimental Psychology: Human Perception and Performance*, *13*, 178-192.

Kelso, J. A. S. (1995). *Dynamic patterns.* Cambridge, MA: MIT Press.

Kostrubiec, V., Dumas, G., Zanone, P., & Kelso, J. A. S. (2015). The virtual teacher (VT) paradigm: Learning new patterns of interpersonal coordination using the human dynamic clamp. *PLoS One*, *10*(11): e014229. Doi:10.1371/journal.pone.0142029.

Lucas, J., Walton, A., Kallen, R. W., Coey, C. A., & Richardson, M. J. (2015). Joint Navigation on the Virtual Table. In J. Weast-Knapp, M. Malone & D. Abney (Eds.) *Studies in Perception & Action XIII*. Taylor & Francis.

Nalepka, P., Kallen, R.W., Chemero, A., Saltzman, E., & Richardson, M. J. (submitted). Herd those sheep: Emergent multiagent coordination and behavioral mode switching.

Nemec, B. & Lahajnar, L. (2009). Control and navigation of the skiing robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009* (pp. 2321-2326).

Pikovsky, A., Rosenblum, M., & Kurths, J. (2001). *Synchronization: A universal concept in nonlinear sciences*. New York: Cambridge University Press.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (pp. 25-34). New York, NY, USA: ACM.

Richardson, M. J., Harrison, S. J., Kallen, R. W., Walton, A., Eiler, B., & Schmidt, R. C. (2015). Self-Organized Complementary Coordination: Dynamics of an Interpersonal Collision-Avoidance Task. *Journal of Experimental Psychology: Human Perception and Performance, 41*, 665-79.

Richardson, M. J., Kallen, R. W., Nalepka, P., Harrison, S. J., Lamb, M., Chemero, A., Saltzman, E., & Schmidt, R. C. (in press). Modeling Embedded Interpersonal and Multiagent Coordination. *Complexis 2016 Conference*.

Richardson, M. J., Marsh, K. L., Isenhower, R., Goodman, J., & Schmidt, R. C. (2007). Rocking together: Dynamics of intentional and unintentional interpersonal coordination. *Human Movement Science*, *26*, 867-891.

Schmidt, R. C., Carello, C., & Turvey, M. T. (1990). Phase transitions and critical fluctuations in the visual coordination of rhythmic movements between people. *Journal of Experimental Psychology: Human Perception and Performance*, *16*, 227-247.

Schmidt, R. C., & O'Brien, B. (1997). Evaluating the dynamics of unintended interpersonal coordination. *Ecological Psychology*, *9*, 189-206.

Schmidt, R. C., & Richardson, M. J. (2008). Dynamics of Interpersonal Coordination. In A. Fuchs & V. Jirsa (Eds.). *Coordination: Neural, Behavioral and Social Dynamics*. (pp. 281-308). Heidelberg: Springer-Verlag.

Schmidt, R. C., & Turvey, M. T. (1994). Phase-entrainment dynamics of visually coupled rhythmic movements. *Biological Cybernetics*, *70*, 369-376.

Schöner, G., Haken, H., & Kelso, J. A. S. (1986). A stochastic model of phase transitions in human hand movement. *Biological Cybernetics, 53*, 247–257.

Strömbom, D., Mann, R. P., Wilson, A. M., Hailes, S., Morton, A. J., Sumpter, D. J., & King, A. J. (2014). Solving the shepherding problem: Heuristics for herding autonomous, interacting agents. Journal of the Royal Society Interface*, 11*(100)*, 20140719.

Varlet, M., & Richardson, M. J. (2015). What would be Usain Bolt's 100-meter sprint world without Tyson Gay? Unintentional interpersonal synchronization between the two sprinters. *Journal of Experimental Psychology: Human Perception and* Performance, *41*, 36-41.

Warren, W. H. (2006). The dynamics of perception and action. *Psychological Review, 113*, 358-389.

Washburn, A., Kallen, R. W., Coey, C. A., Shockley, K., & Richardson, M. J. (2015). Harmony from chaos? Perceptual-motor delays enhance behavioral anticipation in social interaction. *Journal of Experimental Psychology Human Perception & Performance*, *41*(4), 1167-1177.

Zhang, M., Dumas, G., Kelso, J. A. S., & Tognoli, E. (2016). Enhanced emotional responses during social coordination with a virtual partner, *International Journal of Psychophysiology*, doi: 10.1016/j.iipsycho.2016.04.001.

Small Bugs, Big Ideas: Teaching Complex Systems Principles Through Agent-Based Models of Social Insects

Yu Guo and Uri Wilensky

Northwestern University
yuguo2012@u.northwestern.edu

Abstract

Complex systems are challenging for students, especially younger students, to learn. In this paper, we argue that agent-based models (ABMs) of social insects provide an engaging and effective space for students to learn powerful ideas about complex systems. We designed a curricular unit called BeeSmart centering on ABMs of honeybees' collective behavior. Preliminary results from an implementation at a high school showed that ABMs of social insects could be a promising approach to introduce complex systems to a younger audience.

Introduction

The study of complex systems in the past few decades has provided scientists with powerful frameworks to investigate phenomena that were difficult to understand through classic scientific methods (Bar-Yam, 1997; Jacobson & Wilensky 2003). Through a complex systems lens, scientists see the behavior of a system at the macro level as emerging from the interactions of its individual elements (or agents) at the micro level (Epstein, 1999). The complex systems approaches are not only scientifically powerful, but also pedagogically important, because concepts and methods, such as emergence, self-organization, positive feedback loop, and agent-based modeling, have the potential for students to develop new intellectual horizons and new explanatory frameworks that cut across multiple disciplines (Jacobson & Wilensky 2003).

Teaching students complex systems principles is both an opportunity and a challenge. Educational research on students' learning about complex systems shows that they have significant difficulties in understanding complex systems: The aggregated properties of complex systems usually appear to be disconnected from their constituting agents' details (Miller & Page, 2007), which makes emergence counterintuitive to students. Wilensky and Resnick (1999) describe students' difficulties with complex systems as a "deterministic-centralized mindset" (aka DC mindset). Novices tend to see complex systems as a deterministic "clockwork" system, where elements of the system are interconnected like gears in clockwork (Jacobson, 2001). Novices also tend to think that for patterns to emerge, centralized leadership is necessary.

Empirical studies suggest that Agent-Based Models (ABMs) can lower the threshold of learning complex systems and are effective in helping students overcome their DC mindset (e.g., Sengupta & Wilensky 2009). ABMs are computational models that simulate the actions and interactions of agents—individual parts of a complex system—and provide a view to assess the effects of these interactions on the system as a whole. NetLogo (Wilensky, 1999) is a widely used agent-based modeling environment in both scientific research and in education. Much work has been done using NetLogo ABMs to teach existing curricular contents with a complex systems view (Levy & Wilensky 2008; Sengupta & Wilensky, 2009), but few projects focus on explicitly teaching complex systems principles.

ABMs of social insects—honeybees, ants, wasps, and termites—could be a productive way to teach complex systems principles. Social insects' behavior demonstrates several core complex systems principles that are prevalent in natural and artificial complex systems, including positive feedback loops, randomness, interaction, and homogeneity. These principles can explain the apparent disconnection between the system and its parts: how the intelligence of swarms emerges from a collection of non-intelligent insects that follow a set of very simple rules. Learning about social insects' behavior can help students make sense of complex systems principles in rich contexts and provide students a mental model to think with when explaining similar complex systems.

What makes social insects more appropriate as an entry point to study complex systems is that they are both familiar and foreign to students. Students have learned many aspects about bees and ants from an early age, because these social insects have been a popular topic in children's fairy tales, in school curricula, and even embedded in many different cultures and languages. Yet, very few people understand the details of insect colonies' behavior, especially how the overall behavior of a colony can emerge from simple interactions between individual insects. ABMs of social insects allow students to draw prior knowledge about familiar topics to build new knowledge about complex systems.

The BeeSmart Curricular Unit

We designed a curricular unit called BeeSmart for high school students to learn complex systems principles. The unit is based on Seeley's (2010) research findings about how honeybees pick their new hive site. The swarm's decision-making process is best explained through a complex systems lens. The system consists of *multiple agents* that obey simple behavioral rules: if a scout bee discovers a potential hive site, she inspects it. Then she goes back to the swarm and reports

the location, distance, and quality—the suitability of the site is shown by doing a waggle dance. The waggle dance is bees' *medium of communication*. Bees have the instinctual ability to assess the quality of potential hive sites. If the quality is high, bees dance enthusiastically for a long period of time to advertise it; if the quality is low, they make a few brief lackluster dances or do not dance at all. In this way, bees encode differential preference into the dances: the longer the dance, the better the site, and the stronger the signal. Because dances for higher quality sites are presented in the swarm for a longer time, neutral scouts—the observers in the swarm—are more likely to see such dances. When they see a dance, they become recruited and proceed to inspect the advertised hive site. Such simple behavior and interactions between dancers and observers at the *micro level* result in a *positive feedback loop*: signals advocating high quality hives are amplified. Usually, the hive site with the highest quality eventually receives all the support and wins out.

In the BeeSmart Hive Finding model (See Figure 1), each bee obeys the rules described above. In addition, students can use sliders to manipulate environmental variables and bees' behavior. The plots show the count of bees with different states and how their support converges over time. In addition to the model, we designed instructional materials including short readings on complex systems and bees, short videos of bees' waggle dance, illustrations of bee's hive-seeking environment in the real world, instructions of how to use the model, and questions about the model.
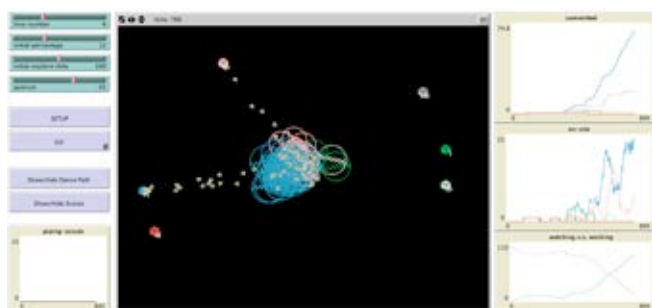

Figure 1. BeeSmart Hive-Finding Model.

School Implementation and Results

We conducted a small-scale pilot study of the BeeSmart unit in a high school mathematical modeling class at an ethnically diverse suburb of Chicago. All 14 students in the class participated in the study. The intent of this study is not to generalize the findings from such a small sample. Instead, we would like to look closely at how students interact with BeeSmart and what is possible to achieve.

Through the 5-day unit, students were highly engaged in honeybees' hive-finding phenomenon and were motivated to answer the driving question "How does a swarm of 10,000 bees pick the best potential hive site from many choices?" The 14 students in the class worked in pairs, interacted with the model, went through the materials, asked questions, and participated in discussions. To test what complex systems principles students learned from the unit, we conducted pre- and post-tests asking the students to explain the hive-finding phenomenon. In addition, we asked students to interact with

another NetLogo model about ants' foraging phenomenon (Wilensky 1997) and explain how it works using principles they learned from the honeybees. Nine students completed both the pre- and post-tests. Four students were selected as focal students to represent different gender and prior knowledge with computational modeling. We interviewed the focal students both before and after their using the unit.

In the pre-test, the students showed relatively high knowledge about bees. They knew that bees' behavior was shaped by evolution, and one student even knew about "swarm mentality". However, most students' answers showed a deterministic mindset, such as the bees splitting into certain number of groups to find new hives. In the post-test, all students provided more sophisticated explanations with complex systems principles such as randomness, interaction, feedback loop, and homogeneity to explain the phenomena of both honeybees and ants. The four students interviewed elaborated on the mechanism of bees' hive finding, incorporating randomness involved in insects' movement and interactions to explain how the group level behavior emerged from simple rules and randomness.

For future work, we will develop more curricula centering on ABMs of social insects for students to explore complex systems. We will also conduct studies at a larger scale to test the effectiveness of this approach.

Acknowledgements

References

Bar-Yam, Y. (1997). *Dynamics of complex systems*. Reading, MA: Addison-Wesley.

Epstein, J. M. (1999). Agent-based computational models and generative social science. *Generative Social Science: Studies in Agent-Based Computational Modeling*, 4–46.

Jacobson, M. J., & Wilensky, U. (2006). Complex systems in education: Scientific and educational importance and implications for the learning sciences. *The Journal of the Learning Sciences*, *15*(1), 11–34.

Levy, S. T., & Wilensky, U. (2008). Inventing a "mid level" to make ends meet: Reasoning between the levels of complexity. *Cognition and Instruction*, *26*(1), 1–47.

Miller, J. H., & Page, S. E. (2007). *Complex adaptive systems an introduction to computational models of social life*. Princeton, N.J.: Princeton University Press.

Seeley, T. D. (2010). *Honeybee democracy*. Princeton, NJ: Princeton University Press.

Sengupta, P., & Wilensky, U. (2009). Learning Electricity with NIELS: Thinking with Electrons and Thinking in Levels. *International Journal of Computers for Mathematical Learning*, *14*(1), 21–50. http://doi.org/10.1007/s10758-009-9144-z

Wilensky, U. 1999. NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL.

Wilensky, U., & Resnick, M. (1999). Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and Technology*, *8*(1), 3–19.

A Telepresence-Robot Approach for Efficient Coordination of Swarms

Karl Tuyls[2]⋆, Sjriek Alers[1], Elisa Cucco[2]⋆, Daniel Claes[2] and Daan Bloembergen[2]

[1]Maastricht University / Fontys University of Applied Sciences, Eindhoven the Netherlands
[2]Department of Computer Science, University of Liverpool, UK
⋆k.tuyls@liverpool.ac.uk, Elisa.Cucco@liverpool.ac.uk

Abstract

In this paper we explore a novel perspective on surveillance robotics, which is based on a coordination principle of honeybees, and on the integration of an autonomous telepresence robot in such system. Coordination principles, based on biological systems such as ant, bee and termite colonies, show several properties which are essential to multi-robot surveillance, including low computation load, robustness, scalability and adaptability. In this paper we aim to improve on the efficiency of such a robotic swarm by taking a human in the loop by means of a telepresence robot. The human operator controlling the telepresence robot will aim to speed up the convergence of the swarm. The experiments, which evaluate the proposed multi-robot coordination system both in simulation and on real robots, show how the telepresence robot substantially increases the efficiency of the process.

Introduction

In recent years there has been a rapidly growing interest in using teams of mobile robots for automatically surveilling environments of different types and complexity. This interest is mainly motivated by the broad spectrum of potential civilian, industrial, and military applications of multi-robot surveillance systems (Kuorilehto et al., 2005; Folgado et al., 2007). Examples of such applications are the protection of safety-critical technical infrastructures, the safeguarding of country-borders, and the monitoring of high risk regions and danger zones which cannot be entered by humans in the case of a nuclear incident, a bio-hazard, or a military conflict.

Triggered by this interest, today automated surveillance is a well-established topic in multi-robot research, which is considered to be of particular practical relevance. Despite the remarkable progress made on this research topic so far, there is still a huge gap between theory and practice of multi-robot surveillance systems, and as a consequence there are still only very few on-field deployments. The reason for this is that many basic questions about coordination among mobile robots are not yet answered in a satisfactory way.

In this paper a new approach on multi-robot surveillance systems is proposed, which is based on a bio-inspired coordination principle from swarm intelligence and on the integration of an autonomous telepresence robot in such system.

Natural entities, such as ant and termite colonies improve their collective performance by influencing one another through local messages they deposit in their shared environment. In computer science, robotics and economics a number of computational variants have been developed, and it has been shown that they allow for very efficient distributed control and optimization in a variety of problem domains. For instance, recent work shows a strong potential in creating artificial systems that mimic insect behaviour that can solve complex coordination tasks such as e.g., routing on the internet, mobile ad hoc network routing, robotic tasks, etc. (Lemmens and Tuyls, 2012; Dressler and Akan, 2010; Floreano and Mattiussi, 2008).

Swarm optimisation algorithms, like ant colony optimisation (Dorigo et al., 2006), rely on pheromone trails to mediate (indirect) communication between agents. These pheromones need to be deposited and sensed by agents while they decay over time. Though easy to simulate, artificial pheromones are hard to bring into real-life robotic applications. However, recently non-pheromone-based algorithms were developed as well (Lemmens, 2011). Such algorithms are inspired by the foraging and nest-site selection behaviour of (mainly) bees. In general, bees explore the environment in search for high quality food sources and once returned to the hive they start to dance in order to communicate the location of the source. Using this dance, bees recruit other colony members for a specific food source. The algorithm we used draw inspiration from these insect behaviours with the goal to create intelligent systems for distributed coordination that can be deployed in real world settings.

The key idea put forward in this paper is that a telepresence robot can improve upon the efficiency of such a swarm. Telepresence robotics is a form of teleoperation, namely the extension of a person's sensing and manipulation capability to a remote location, in which a human operator act as a supervisor intermittently communicating information about goals and

actions relative to a specific task. The human operator will receive information about accomplishments, difficulties and, as requested, raw sensory data, while the subordinate telepresence robot executes task based on information received from the human operator plus its own sensing and artificial intelligence (Sheridan, 1989). In the approach we propose in this paper the human operator controlling the telepresence robot can observe the environment and will aim to steer the behaviour of the swarm by means of direct communication.

In the following sections we introduce telepresence robotics and the biological background of our foraging approach. Then we show our experiments and discuss the efficiency of the algorithm and the improvement obtained by integrating a telepresence robot in the system.

Telepresence robotics

Already more than 30 years ago, artificial intelligence pioneer Marvin Minsky (Minsky, 1980) laid out an ambitious plan calling for the development of advanced teleoperated robotics systems that would result in a remote-controlled economy. He coined the term "telepresence" to describe these systems, which in his futuristic vision would transform work, manufacturing, energy production, medicine and many other facets of modern life. Although the idea of a teleoperated robot for remote presence is not new, only recently telepresence robots become available to the broader public (Lazewatsky and Smart, 2011; Takayama et al., 2011; Tsui et al., 2011). Basically, telepresence robotics systems can be described as embodied video conferencing on wheels, providing a physical presence and independent mobility in addition to communication, unlike other video conferencing technologies, allowing the user to interact more naturally in the remote office environment.

However, telepresence robots can be deployed in a wide range of application domains: the informal meeting scenario in offices, in hospitals to allow doctors to provide consultations from a distance (Tsui et al., 2011) or to pay a virtual visit when it is not possible to be present in person, or to give people with restricted mobility a new way to interact beyond their possibility. Furthermore, many work-sites are hazardous to human health or even survival. With telepresence robotics it will be potentially possible to operate in dangerous environments without such risks.

Adding a level of autonomy to a telepresence robot can greatly improve the experience of the user, as it reduces their cognitive load. This allows to focus more attention on the interaction and to the task and less on controlling the robot (Tsui et al., 2011). However, it remains important for the operator to have control over the behaviour of the system. Indeed, as a telepresence robot is controlled from a remote location, precise control and feedback of the robot is required. One possible solution, assisted navigation, is investigated by Takayama et al. (2011). Adding more autonomy and integrating the findings of recent AI research into the platform can greatly increase the usability of these robots.

Biological coordination

A great deal of research in swarm intelligence is situated in the area of bio-inspired computation; more precisely in the area that investigates algorithms that find inspiration from nature in order to develop novel computational models, often to solve coordination problems. Foraging is one of the coordination problem in this domain. Essentially it consists of two sub-problems: path construction/planning and path exploitation/repair. The task of foraging consists of gathering objects out of the environment and returning them to a central point, most often the starting location. A commonly used method for solving foraging problems focuses mainly on the behaviour of social insects such as ants and bees.

Ants deposit pheromone on the path they take during travel. Using this trail, they are able to navigate toward the food location and communicate with other members of the colony, not directly but by accumulating pheromone trails in the environment. Pheromone strength indicates the "fitness" of a trail but is not able to indicate direction, therefore an ant is not able to know a priori to which destination it is travelling. When a trail is strong enough, other ants are attracted and will follow it towards a destination which results in a reinforcement of the trail. This is known as an autocatalytic process: the more ants follow a trail, the more that trail becomes attractive for being followed. Short paths are reinforced more often over time and will eventually be preferred. This principle is used to address several problems, such as Routing Problem (Di Caro et al., 2005) and area coverage with robots (Wagner et al., 1999; Ranjbar-Sahraei et al., 2012).

On the other hand, bees and desert ants do not use pheromones to navigate in unfamiliar environments. Their navigation mainly consists of Path Integration (PI). The PI vector represents the continuously updated knowledge of direction and distance and, as a consequence, bees are able to return to their starting point by choosing the direct route rather than their outbound trajectory. More precisely, when the path is unobstructed, the insect exploit previous search experience. However, when the path is obstructed, the insect has to fall back on other navigation strategies such as exploration (Collett and Collett, 2009). For recruitment bees communicate with other colony members by means of a waggling dance performed in the hive. The direction of the food source is read from the angle between the sun and the axis of a bee's waggle segment on the vertical hive comb, while the duration of the waggle phase is a measure of the distance to the food and the "fitness" of a solution
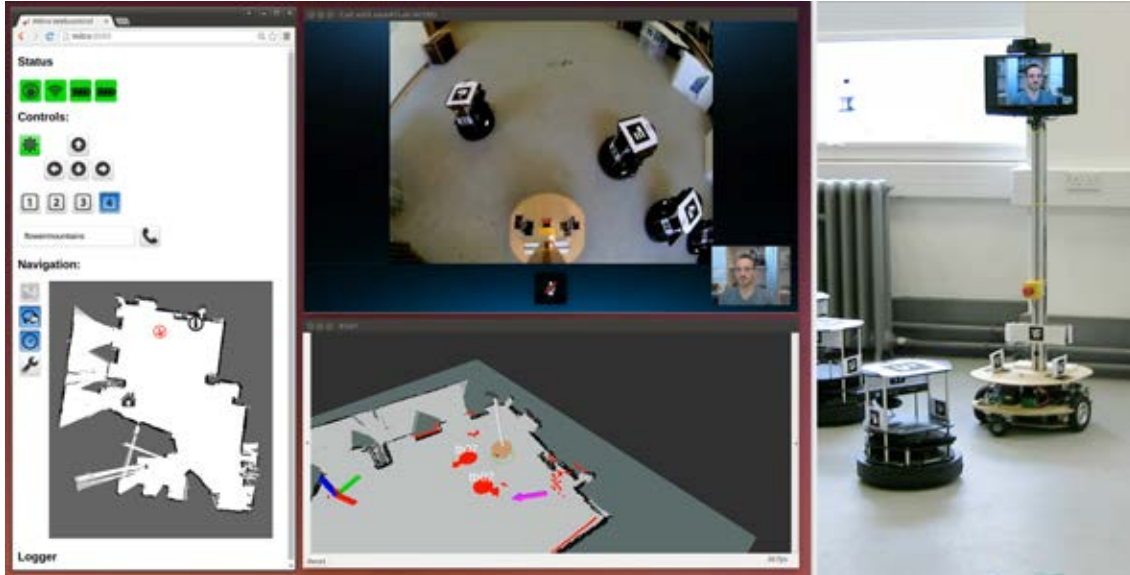
Figure 1: MITRO interface and Turtlebots foraging supervised by a telerobot.

(von Frisch, 1967). More precisely, depending upon the strength of the dance, more bees are attracted and follow the PI vector toward a destination. Furthermore, the more bees follow a PI vector, the more that destination will be communicated and the more it will attracts other bees. Eventually, the best solution prevails.

Transferring these principles to algorithms is the domain of computational swarm intelligence. Comparisons of these algorithms (Lemmens et al., 2008) show that the bee-inspired mechanism is able to collect all the items in the environment faster than the ant-inspired mechanism in a relatively unobstructed environment. However, in an environment with more obstacles and/or dynamic environment, the bee-inspired mechanism is less adaptive.

System & Approach

The main idea of the proposed approach is to integrate swarm algorithms with telepresence robotics. We build on previously developed algorithms in swarm robotics, aiming to achieve a food foraging application in the real world guided by a telepresence robot that will be shepherding the swarm.

Similar to the Path Integration principle, the robots in our swarm estimate their positions by integrating information coming from the gyroscope and the wheel odometry. Using this, the robots can always compute the home vector (HV), and if the food location is seen, the path integration (PI) vector can be used to communicate the location to other robots.

Therefore, no map of the environment needs to be built by the robots and the only common reference point that is needed for the correct communication of the food locations is the hive location, i.e. the HV.

As a consequence, if the odometry is faulty, the robot might not find the hive or food location, and if this problem occurs the robots fall back in search mode. As soon as the hive or the food are seen again, the robots update their internal reference system.

In contrast to the honeybees's behaviour, we also allow communication outside of the hive, since it is very likely that the robots see each other under way. Additionally, there's also a probability that the robots return to the hive after being in the search state for a long time, in order to increase the chance to meet another robot that might already be in foraging mode.

This approach has been demonstrated to work reasonably well for small environments (Alers et al., 2014a,b). However, there was no human supervision involved and also no simulation runs were performed to gain empirical insights about the performance of the swarm, i.e. how long it takes the swarm to converge on the food locations, what is the throughput of the system, etc.

In this paper, we propose a novel approach to add a human shepherd to the system, which can supervise the swarming robots and help to enable faster convergence. The idea is that a human can interact with the swarm using a telepresence robot as a shepherd. The human operator can have more knowledge of the environment, i.e. a map and a camera. After a food location has been found, the shepherd can steer the swarm towards that location or catch "lost" swarm robots.

We implemented the approach using the Turtlebot[1] platform as swarm robots and a custom-built telepresence robot MITRO (Alers et al., 2013) as shepherd. These platforms will be explained in more detail in the next subsections.

---

[1] http://www.turtlebot.com

Figure 1 shows an overview of the system. On the left, the interface for the human controlling MITRO is shown. It gives an overview of the system's status, allows the user to control the robot and shows the live video feed of the environment. Additionally, the internal view of the robot is shown, below the video feed. The reference frame is depicted as axis, and the two circles with arrows are the detected robots. On the right, a picture of the real-life experiment is shown, where MITRO is shepherding in the middle of several Turtlebots.

Swarm robots

As explained before, for the real world experiments, we use the Turtlebot platform. It has a laptop on board with a core-i3 CPU for computation, running the Robot Operating System (ROS) (Quigley et al., 2009) framework. The robots are also equipped with a Kinect sensor and the RGBD information is used to detect and locate AR markers, see black and white markers in Figure 1. This sensor is also used for the obstacle detection, together with three bumpers located in front half of the robot.

To enable visual robot-robot detection every Turtlebot has six unique markers, oriented in a way that at least one marker is always visible. To track and decode these markers we use the ROS wrapper of the ALVAR toolkit [2]. We use a customised bundle detection method to determine the center of the detected robot. Each marker in the bundle encodes the robot number and its location with respect to the center of the robot. This information is used to predict the position of the detected robot. Kalman filtering is also applied to get more stable and accurate estimation of the detected robots position, heading and speed. These parameters are also used for the collision avoidance.

Communication between Turtlebots is realised over wi-fi using a UDP connection to each Turtlebot. Even though global communication would be possible, we limit the communication of each robot to its own channel and allow only communication after visual detection of its peer. Therefore, the robots can communicate only with another robot if it is in close proximity.

In order to avoid collisions between robots we rely on the marker detection to predict positions and speeds of the other robots. The obtained information could be used to efficiently compute a non-colliding speed vector (Claes et al., 2012). In contrast to the previous approach, in which the robots avoided each other by using a global reference frame and broadcasting the positions to all robots via Wi-Fi, we adapted this method to only rely on the marker detection and the predictions using a Kalman filter. However, a few collisions still might occur due to

the failed detection of other robots and additionally in such configurations in which the robots cannot see each other because of the field of view of the Kinect sensor.

Telepresence robot

In addiction to the Turtlebot platform we also use a custom-built telepresence robot, shown in the right panel of Figure 1 (Alers et al., 2013). The advantage of using a custom-built system over a commercial platform is the flexibility, extendibility and knowledge of the complete system, that for our purpose is crucial.

The robot has a height of approximately 160 cm and is based on the Parallax Mobile Robot Base kit, which includes the base plate, powerful motors and 6 inch wheels with pneumatic tires. The sensors include a low-cost LIDAR, an Asus XTION PRO 3D sensor, sonar sensors, and two cameras (one pointing forward for conversations, one fish-eye camera pointing downwards for driving). The robot is also running ROS.

Since the robot is controlled from a remote location, we implemented low level autonomy on the robot in the form of assisted teleoperation. With assisted teleoperation the robot follows the steering commands of the operator except for a situation when there is a high chance of collision. This can easily occur when the user is not experienced in navigating the robot, the network connection is delayed or an obstacle suddenly appears in front of the robot. Additionally, the video feed can be switched from front- to down-facing, and is augmented with a projection of the expected navigational path. Furthermore, the robot is able to perform SLAM (simultaneous localization and mapping) to build a map of its environment (Thrun et al., 2005); this map is used subsequently for localization and autonomous navigation to a chosen destination, or back to his charging location, all to ease the remote operation.

Experiments

In our experiments the Turtlebots are performing a foraging task, starting at the hive (H) location and randomly exploring the unknown environment for a specific food (F) location. The robots can also locate the food location by asking bypassing robots for a known food location, see Figure 2. When the source is found the Turtlebots start to exploit this source, driving from the food to the hive, where they drop the food, until the food is depleted or another source is found. The telepresence robot works as a "shepherd" sending relative location information to the Turtlebots.

We implemented our approach on the real robots as in simulation for getting additional statistics. In this experiment section we will describe the simulation results, in the demonstration section the real-world setting is shown. Simulations are run in real time using
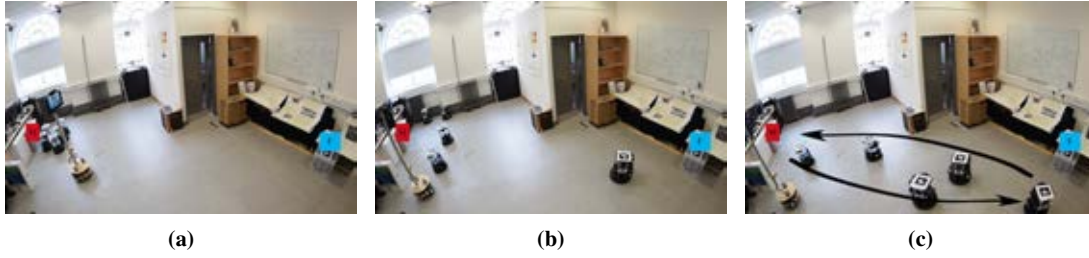
[2]http://www.wiki.ros.org/ar_track_alvar

(a)   **(b)**   **(c)**

Figure 2: Multi-Robot foraging. **(a)** All robots start at the hive (H) location. **(b)** Robots are exploring the unknown environment randomly. The left two robots have found the food (F) location and are foraging between the hive and the food location. **(c)** All robots have converged to foraging behaviour.
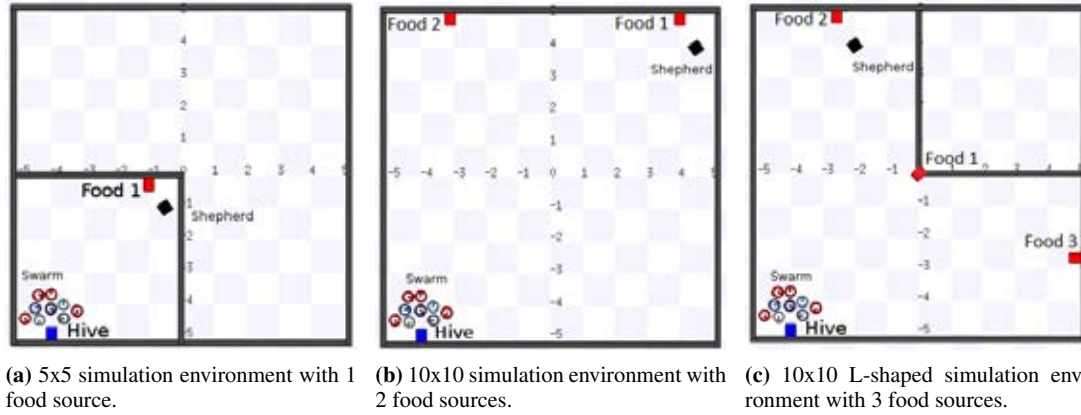

(a) 5x5 simulation environment with 1 food source.   **(b)** 10x10 simulation environment with 2 food sources.   **(c)** 10x10 L-shaped simulation environment with 3 food sources.

Figure 3: The different simulation environments with the shepherding robot (black square), food sources (red square), and 9 robots located at the hive-location (blue square)

Stage (Gerkey et al., 2003; Vaughan, 2008). We use simulated Turtlebots and a simple differential drive robot as telepresence robot. For the detections, mock-ups are written, so that the same state-machine is run on the real robots and in simulation. Having the simulation setup allows us to investigate the system performance for different scenarios and using more repetitions than would be feasible in the real world.

The main goal of our experiments is to compare the performance of the original bee-inspired algorithm with the newly proposed approach that has the telepresence robot in the system. We evaluate the proposed approach in simulation for 3 different environments: 5x5 meters square shaped, 10x10 meters square shaped, and 10x10 meters L-shaped, shown in Figure 3.

In the first case, we compare the performance of the swarm for different numbers of Turtlebots involved in the foraging task. We evaluate the throughput, the speed of convergence and the efficiency of the foraging process with and without the shepherding telepresence robot. We also collect statistics showing the user effort, expressed as the number of times the user interfered (i.e. corrected a Turtlebot's navigation), and the total distance driven by the telepresence robot. We repeat the same experiments in the 10x10 world

and in the 10x10 L-shaped environment with 9 Turtlebots, and for these cases we evaluate the convergence of the algorithm after moving the food to a different location, e.g. due to depletion of the first food source. Each experiment lasts until 50 food units have been transported from the source to the hive. Similarly, in the 10x10 environments, a food source becomes "depleted" after 50 food units, upon which a new source becomes active. Every experiment is repeated 10 times, and the results are averaged.

Discussion

Figure 4 shows the results of simulations in the 5x5 world. In this relatively small environment the swarm will often converge without the interference of the telepresence robot, except for a few cases when the number of robots get too large for the environment, leading to collisions, and robots getting stuck. However, through minimal user effort the shepherd still improves the efficiency of the process.

Figure 4(a) shows the total time, in seconds, needed to complete the task (i.e., transport 50 food units), with the error bars representing the standard deviation intervals. We observe that the optimal swarm size is reached at 6 robots, both with and without shepherd. When the swarm size increases beyond this point, the small environment becomes too clut-
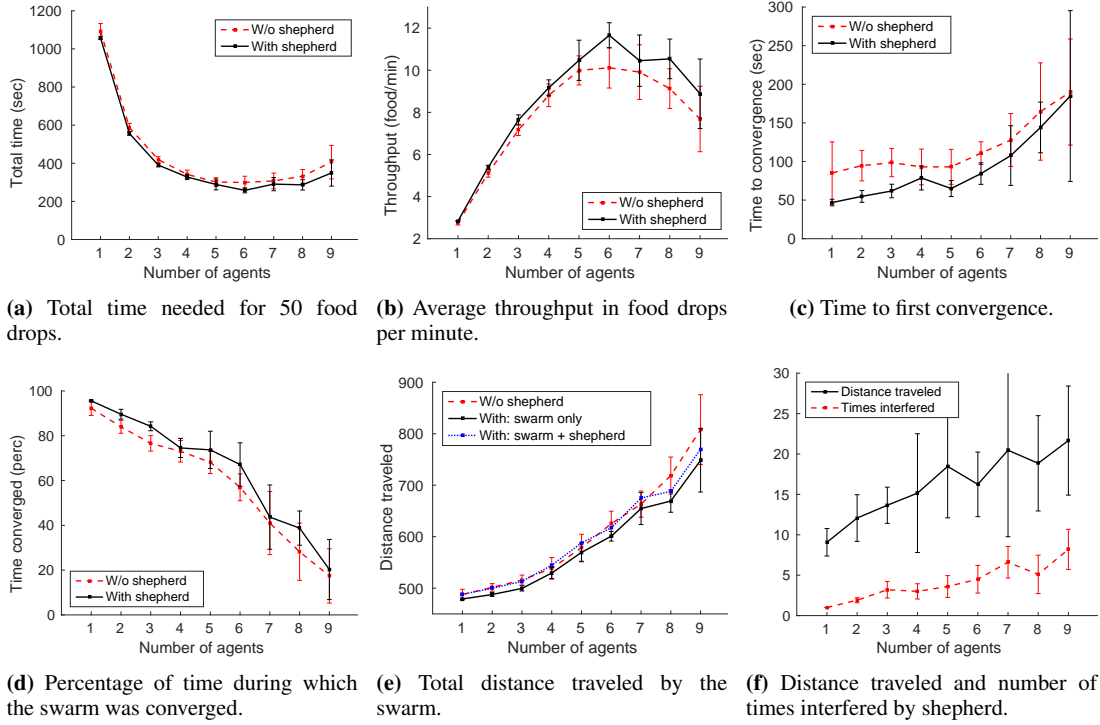
(a) Total time needed for 50 food drops.

(b) Average throughput in food drops per minute.

(c) Time to first convergence.

(d) Percentage of time during which the swarm was converged.

(e) Total distance traveled by the swarm.

(f) Distance traveled and number of times interfered by shepherd.

Figure 4: Results for the 5x5 world with different swarm size.

tered as robots start colliding, hindering each other's performance. The same trend can be observed when looking at the total throughput in Figure 4(b), measured in units of food delivered per minute. Here again we see that the optimum is reached for a swarm size of 6 robots. Shepherding significantly improves the performance of the swarm in both cases.

We also investigate the convergence performance of the system. In Figure 4(c) we show the time needed in seconds until the whole swarm is converged, meaning that all robots are aware of the food location and are continuously going back and forth between the hive and food location to transport food units. The figure shows that the time needed to converge stays more or less stable up to 5 robots, after which the environment becomes more cluttered, preventing the robots from converging quickly. Additionally, the converged state can be lost again, e.g. due to collisions, or robots driving in each others line of sight preventing them from relocating the food. In Figure 4(d) we plot the percentage of experiment time during which the whole swarm is converged, and note that this value decreases approximately linearly with an increasing swarm size. The fact that robots may get in each other's way can also be observed by looking at the total distance travelled (in meters) by the swarm during the course of one experiment, which increases exponentially with the swarm size (Figure 4(e)). This shows that even though a swarm of size 6 is optimal in both time and throughput, it is

not necessarily the most efficient in terms of per robot performance.

Finally, in Figure 4(f) we look at the effort required by the user to guide the swarm. The figure shows the distance travelled by the telepresence robot, as well as the number of interferences, i.e. the number of times that the user has corrected a swarm robot's navigation target. We can see that the required effort doesn't necessarily grow with the number of swarm robots, indicating that the robots are able to relay the new information among the swarm.

We now move on to the larger environment. Table 1 shows the results for the 10x10 world (with terms between parenthesis representing the standard deviation) with and without moving the food source. In both experiments the shepherd can significantly improve the performance of the system. In particular, after moving the food source the swarm without shepherd takes more than twice as long to re-converge (third column in the table) as the swarm with shepherd. Also note that when moving food, without shepherd the swarm only fully re-converged in 3 out of 10 runs, while with shepherd this happened in 9 out of 10 runs.

Results for the 10x10 L-shaped environment are shown in Table 2. Again, shepherding significantly improves the performance of the swarm with relatively limited effort. However, this task is clearly harder, as the food source is moved twice. A breakdown of time to re-convergence, as well as the num-

Table 1: Results for the 10x10 world with and without moving the food source.

	Total time	Time to conv. (1)	Time to conv. (2)	% of time converged	Throughput	Shepherd distance	Times interfered
Static							
W/o shepherd	510.5 (56.4)	304.7 (68.5)	n/a	38.5 (4.9)	5.9 (0.6)	n/a	n/a
With shepherd	389.7 (12.9)	191.9 (42.4)	n/a	47.7 (10.1)	7.7 (0.3)	44.8 (12.6)	8.0 (2.8)
Moving food							
W/o shepherd	922.6 (68.2)	328.1 (75.1)	316.9 (81.7)	20.0 (5.6)	6.5 (0.5)	n/a	n/a
With shepherd	677.7 (42.9)	181.2 (51.3)	146.7 (37.7)	46.0 (11.3)	8.8 (0.5)	88.9 (19.4)	20.4 (3.2)

Table 2: Results for the L-shaped world with and without shepherding.

	Total time	Throughput	Distance traveled	Shepherd distance	Times interfered
W/o shepherd	1068.8 (167.4)	8.3 (1.0)	3347.9 (466.6)	n/a	n/a
With shepherd	895.2 (73.4)	10.1 (0.8)	2984.5 (96.9)	121.2 (14.0)	26.8 (6.6)

Table 3: Break-down of convergence times in the L-shaped world with and without shepherding. Food is moved twice. Convergence times are listed for the three food locations, as well as the number of runs that did converge.

	Time to conv. (1)	Time to conv. (2)	Time to conv. (3)	Nr. of conv. (1)	Nr. of conv. (2)	Nr. of conv. (3)
W/o shepherd	223.3 (57.0)	235.6 (91.3)	278.2 (60.7)	8/10	6/10	4/10
With shepherd	188.9 (73.2)	137.6 (39.0)	201.4 (67.6)	10/10	10/10	9/10

ber of runs that fully re-converged, is given in Table 3. Both metrics are significantly improved by the shepherd. We note that the first time the food is moved, the shepherd is able to make a big difference, as the distance between both food locations is easy to overcome. In contrast, the last food location lies in the opposite side of the L-shape, around the corner. This makes it harder for the swarm to re-locate the food, even with the help of the shepherd.

Demonstration

We have also undertaken a real-world experiment in which 5 Turtlebots are foraging in an unknown environment. All the robots are initially located around the hive and they start to explore the environment randomly for the food location. An operator supervises the group using the MITRO telepresence robot. The user is able to send the food location information to individual Turtlebots, e.g. when they get stuck. A video showing this demonstration can be found online.[3] In this physical implementation the sheperd robot increases the efficiency of the foraging process and speed up the convergence of turtlebots, especially when the food is moved.

Conclusion and further work

We have proposed a new approach for swarm robotics systems, which is based on both the coordination principle of honeybees and on human-robot interaction through telepresence robotics. In order to vali-

date the approach we performed swarm experiments, i.e., a foraging task in a unknown environment, both in simulation and in a situated environment. Our results show that the telepresence robot, acting as a shepherd, can substantially increase the efficiency of the foraging process, especially in dynamic and complex scenarios, in which food sources change over time. Only a limited effort by the telepresence robot can already make a great difference in performance. In future work we aim to integrate an augmented telepresence robot in a swarm, allowing interaction between a human operator and the multi-robot system in a complex, potentially dangerous surveillance task. The human operator would be able to steer the behaviour of the swarm from a remote location by means of direct communication.

References

Alers, S., Bloembergen, D., Claes, D., Fossel, J., Hennes, D., and Tuyls, K. (2013). Telepresence robots as a research platform for AI. In *Proc. of the AAAI Spring Symp. on Designing Intelligent Robots: Reintegrating AI II*, pages 2–3.

Alers, S., Claes, D., Tuyls, K., and Weiss, G. (2014a). Biologically inspired multi-robot foraging. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems (AAMAS)*, pages 1682–1684.

Alers, S., Tuyls, K., Ranjbar-Sahraei, B., Claes, D., and Weiss, G. (2014b). Insect-inspired robot co-

---

[3]http://smartlab.csc.liv.ac.uk/shepherding/

ordination: Foraging and coverage. In *Artificial Life 14*, pages 761–768.

Claes, D., Hennes, D., Tuyls, K., and Meeussen, W. (2012). Collision avoidance under bounded localization uncertainty. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1192–1198.

Collett, M. and Collett, T. S. (2009). Local and global navigational coordinate systems in desert ants. *Journal of Experimental Biology*, 212(7):901–905.

Di Caro, G., Ducatelle, F., and Gambardella, L. (2005). Swarm intelligence for routing in mobile ad hoc networks. In *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, pages 76–83.

Dorigo, M., Birattari, M., and Sttzle, T. (2006). Ant colony optimization – artificial ants as a computational intelligence technique. *IEEE COMPUT. INTELL. MAG*, 1:28–39.

Dressler, F. and Akan, O. B. (2010). A survey on bio-inspired networking. *Computer Networks*, 54(6):881 – 900. New Network Paradigms.

Floreano, D. and Mattiussi, C. (2008). *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. The MIT Press.

Folgado, E., Rincón, M., Álvarez, J. R., and Mira, J. (2007). *Nature Inspired Problem-Solving Methods in Knowledge Engineering: Second International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2007, La Manga del Mar Menor, Spain, June 18-21, 2007, Proceedings, Part II*, chapter A Multi-robot Surveillance System Simulated in Gazebo, pages 202–211. Springer Berlin Heidelberg, Berlin, Heidelberg.

Gerkey, B. P., Vaughan, R. T., and Howard, A. (2003). The player/stage project: Tools for multi-robot and distributed sensor systems. In *In Proceedings of the 11th International Conference on Advanced Robotics*, pages 317–323.

Kuorilehto, M., Hännikäinen, M., and Hämäläinen, T. D. (2005). A survey of application distribution in wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.*, 2005(5):774–788.

Lazewatsky, D. A. and Smart, W. D. (2011). An inexpensive robot platform for teleoperation and experimentation. In *Proceedings of ICRA 2011*.

Lemmens, N. (2011). *Bee-inspired Distributed Optimization*. SIKS dissertation series. Maastricht University.

Lemmens, N., De Jong, S., Tuyls, K., and Nowé, A. (2008). *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning: 5th, 6th, and 7th European Symposium, ALAMAS 2005-2007 on Adaptive and Learning Agents and Multi-Agent Systems, Revised Selected Papers*, chapter Bee Behaviour in Multiagent Systems, pages 145–156. Springer Berlin Heidelberg, Berlin, Heidelberg.

Lemmens, N. and Tuyls, K. (2012). Stigmergic landmark optimization. *Advances in Complex Systems*, 15(8).

Minsky, M. (1980). Telepresence. *Omni*, pages 45–51.

Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*.

Ranjbar-Sahraei, B., Weiss, G., and Nakisaee, A. (2012). Stigmergic coverage algorithm for multi-robot systems (demonstration). In van der Hoek, W., Padgham, L., Conitzer, V., and Winikoff, M., editors, *AAMAS*, pages 1497–1498. IFAAMAS.

Sheridan, T. B. (1989). Telerobotics. *Automatica*, 25(4):487–507.

Takayama, L., Marder-Eppstein, E., Harris, H., and Beer, J. M. (2011). Assisted driving of a mobile remote presence system: System design and controlled user evaluation. In *ICRA*, pages 1883–1889. IEEE.

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic robotics*. MIT press Cambridge.

Tsui, K. M., Desai, M., Yanco, H. A., and Uhlik, C. (2011). Exploring use cases for telepresence robots. In *Proceedings of the 6th International Conference on Human-robot Interaction*, HRI '11, pages 11–18, New York, NY, USA. ACM.

Vaughan, R. (2008). Massively multi-robot simulation in stage. *Swarm Intelligence*, 2(2):189–208.

von Frisch, K. (1967). *The dance language and orientation of bees*. Belknap Press of Harvard University Press.

Wagner, I. A., Lindenbaum, M., and Bruckstein, A. M. (1999). Distributed covering by ant-robots using evaporating traces. *IEEE T. Robotics and Automation*, 15(5):918–933.

673

Social Contribution in the Design of Adaptive Machines on the Web

Mark D. Wagy and Josh C. Bongard

Dept. of Computer Science, University of Vermont
mwagy@uvm.edu

Abstract

The Web has created new opportunities for interactive problem solving and design by large groups. In the context of robotics, we have shown recently that a crowd of non-experts are capable of designing adaptive machines over the Web. However, determining the degree to which collective contribution plays a part in these tasks requires further investigation. We hypothesize that there exist subtle yet measurable social dynamics that occur during the collaborative design of robots on the Web. To test this, we enabled a crowd to rapidly design and train simulated, web-embedded robots[1]. We compared the robots designed by a socially-interacting group of individuals to another group whose members were isolated from one another. We found that there exists a latent quality in the robots designed by the social group that was significantly less prevalent in the robots designed by individuals working alone. Thus, there must exist synergies in the former group that facilitate this design task. We also show that this latent quantity correlates with the desired design outcome, which was fast forward locomotion. However, the quantity – when distilled into its component parts – is not more prevalent in one group than another. This finding demonstrates that there are indeed traces left behind in the machines designed by the crowd that betray the social dynamics that gave rise to them. Demonstrating the existence of such quantities and the methodology for extracting them presents opportunities for crafting interfaces to magnify these synergies and thus improve collective design of robots over the web in particular, and crowd design activities in general.

Introduction

The Web has led to novel modes of social participation (DiMaggio et al. (2001)) and means for contribution to tasks that were previously limited to small groups of experts (Khatib et al. (2011); Lintott et al. (2008); Gowers and Nielsen (2009)). New Web technologies, such as WebGL 3D graphics and Web-embedded physics engines, have made the interactive design of intelligent machines possible over the Web (Moore et al. (2014)). Additionally, collective intelligence methods (Quinn and Bederson (2011)) such as crowdsourcing (Howe (2006)), human computation

(Von Ahn (2009)) and social computing (Wang et al. (2007); Parameswaran and Whinston (2007)) can be used to incorporate contributions of large groups of non-experts – the 'crowd' – into the design of robots on the Web (Wagy and Bongard (2014)). However, the ways that people involved in design and problem-solving tasks synergize remains an open question.

Under certain circumstances, collectives have computational abilities not readily available to the group members in isolation (Couzin (2007)). However, the ability of a group of people to socially coordinate problem solving efforts has limits in physical social interactions (Dunbar (1992)). These limitations have also been shown to persist in Web interactions (Gonçalves et al. (2011)).

We seek to better understand whether a crowd of humans interacting socially through the Web can contribute non-destructively, and potentially in a superadditive way (Page (2008)) to collective problem solving. Whether, and in what ways, constructive social computing phenomena arise in human interactions on the Web remains to be seen. Previous studies (Khatib et al. (2011); Lintott et al. (2008); Gowers and Nielsen (2009)) have demonstrated that a crowd of individuals can complete problem-solving and design tasks while working in parallel. However, understanding the ways the crowd can collectively exhibit abilities that differ from that of an aggregate of individual contributions is underexplored. The present study addresses how the contributions of the crowd as a social entity can be measured as distinct from the contributions of isolated individuals working in parallel.

We have shown that the crowd is capable of collectively designing adaptive machines on the Web (Wagy and Bongard (2014, 2015a)). Thus there is evidence that under some circumstances social synergy can arise in this domain. However, past studies have not uncovered the imprint left behind on the crowd-generated designs that result from this synergy.

We hypothesize that this social contribution is a measurable quantity. If this quantity is indeed measurable, then it could be actively managed. If it constructively contributes to the task at hand, it can be enhanced. If it is destructive,

---

[1]For a video overview of the experiment, see https://youtu.be/ODr-lacPKPQ

it could be actively suppressed. In this study, we seek to demonstrate whether or not the quantity is measurable; and if so, whether it is constructive or destructive with respect to the design of robots. Developing automated means for the discovery of crowd contributions is the first step to actively manage crowd participation towards productive, collective outcomes.

Evidence indicates that humans may be biased, during design, by exposure to the physical environment in which we are embodied. For example, people may favor symmetric robot designs as symmetry is ubiquitous in nature. If incorporated into the physical characteristics of artificial organisms, these biases facilitate design (Wagy and Bongard (2015a)). However, we still do not know whether these biases arise individually or if the bias is reinforced by crowd behavior. If these biases are the result of social pressures, they could be actively enhanced or suppressed depending on whether they were beneficial to the desired design outcome.

Previous work suggests there may exist as-yet undiscovered latent contributions from the crowd (Wagy and Bongard (2015b)). However, the methods suffer from a deficiency: they obscure underlying features of the crowd-generated designs that may contribute to a positive outcome. Additionally, we do not know whether design features were indeed influenced by social processes. This stands in contrast to the present work, in which we demonstrate a method for deriving contributions as linear combinations of simple geometric values.

In the present work, we distill a latent geometric factor from robot designs using Singular Value Decomposition (SVD). We demonstrate that this value is more prevalent in a social design process than it is in designs resulting from the parallel effort of isolated individuals. Furthermore, we show that this latent factor has a beneficial effect on the objective of the design process.

Methods

In this study, we used a dataset of user-contributed robot designs (Wagy and Bongard (2015a))[2]. Participants designed robots using an interactive tool available through their web-browser. When a user visited the study website, they were shown a 5×5 grid of dots. By clicking on a dot and dragging to another dot, they could draw line segments between dots (e.g., Fig. 1). Line segments were only allowed between horizontally- or vertically-adjacent dots to constrain users from crossing segments in their designs. In order to enforce this constraint, the closest set of adjacent lines segments that approximate the diagonal line drawn by the user was shown as the user dragged lines between dots. For example, a line drawn from the top left dot to the bottom right dot was approximated by a zig-zag formation of smaller, adjacent segments between the top left and bottom right dots.
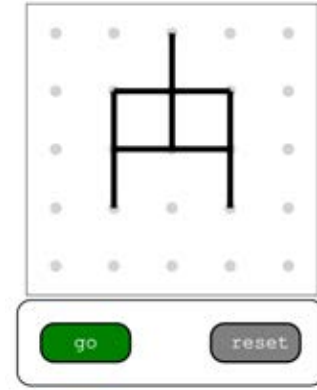
Figure 1: Grid of dots for designing robot bodies by the crowd. An example design is shown drawn on the grid. Users could click on a dot and drag to another dot. Lines were only allowed between adjacent dots. If a user dragged to a dot that was not adjacent, the path of adjacent lines that best approximated the dragged path was generated as the designer drew the line.

When a participant was finished designing a robot, she could click a "GO" button, which launched an instance of her design as a 3D robot in a physics simulation engine within her browser. The simulation contained only the robot and an infinite, flat ground upon which the robot could walk.

Line segments drawn in the grid of dots were translated into the physics simulation as 3D rectangular parallelepiped robot segments. Each dot that was adjacent to at least one line segment was invoked as a 3D cube in the physics simulation. Segments adjacent to a cube were connected to the respective cube with a hinge joint along the axis at the midpoint between adjoining cube and segment faces and in parallel with the ground and these body faces. In this way, the robot was able to push in the direction of the ground. However, as segments flexed and the robot body moved, the robot configuration did allow for sweeping motions across the ground by its members.

Every joint was actuated with a sinusoidal, displacement-controlled signal. All sinusoidal signals driving the hinge joints swept the same angle ($\pm 45°$) at the same frequency (1.5 Hz). However, the phase of the signal could take on one of two possible values: $0°$ or $180°$. A hill-climber search algorithm was used to define which of these phase values was assigned to each of the hinges in the robot created by the user. We will use the term *phase configuration* to refer to a single assignment of phase values to hinge joints for a particular robot body. A separate hill-climber algorithm was maintained for each robot design and for each group. Thus the first instance of a particular robot morphology within a group was assigned a random phase configuration. When that same user or another participant in the same group drew

and simulated an identical robot morphology, the robot was assigned a slightly altered version of the original phase configuration. Thus, each time a user clicked *GO*, they contributed one iteration of the hill-climber algorithm for a particular robot body.

Each robot design was simulated for 15 seconds of physics simulation time (for examples of robots in the web-embedded physics simulation, see Fig. 2). The distance that the robot traveled in these 15 seconds was recorded in a database along with the adjacency matrix that defined the connections.
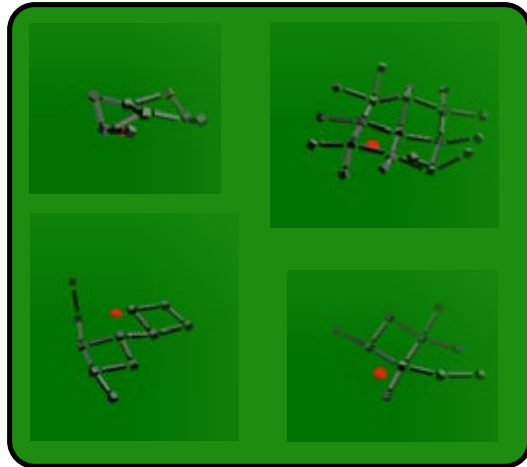
Figure 2: Robots were simulated in a web-embedded physics simulator for 15 seconds. Each line that a user drew was translated into a segment in the robot and each dot that was adjacent to a line was translated into the physics simulation as a cube. The cubes and segments were attached with a hinge joint, which was actuated by a motor with displacement-controlled sinusoidal signal. Red dots indicate the starting point from which the robot started in the simulation.

Participants were asked to design a robot that could move as far as possible across the flat plane and within the allotted 15 seconds of simulation time. However, participants were unpaid volunteers so they were free to use the tool however they desired.

Participants were placed into either a control group or experimental group with an equal probability of being placed into either group. Participant IP addresses were recorded so that if they were to return to the site to design more robots, they would be placed in the same control or experimental group. In a panel at the top of the study website, the experimental group was shown 13 randomly chosen (2D) designs created by other users in their group. We refer to this group of robot designers as the *social group* because they

were given the chance to utilize other users' designs if they so desired. Every time a user returned to the site, they would be shown a potentially new random selection of 13 designs created by other users that were placed in the experimental group. The control group, which we will refer to as the *independent group*, was shown only their own past designs. Thus the user interfaces for the independent and social groups were identical apart from the content of the panels showing historical designs at the top of the site.

When the crowdsourced portion of the experiment was complete, we used the collected robot designs to compare the design preferences for users in the social and independent groups.

Since robot designs consisted of a series of points and edges joining these points, we were able to compute network metrics on the resulting dataset and derive a set of simple geometric measures from each robot designed by the crowd. These geometric measures included minimum, average and maximum degree measures; maximal matching; length of the shortest path; node connectivity; number of legs; number of segments; radius; transitivity; number of cliques; indicators of bipartiteness, regularity, whether the network is a tree and biconnectedness; and symmetry (computed as the maximum proportion of segments that are matched with another segment across either the horizontal, vertical or one of two diagonal axes of symmetry in the 2D design plane).

We then distilled this set of computed geometric measures for each robot morphology into just one representative value of its geometric features. We did this by using the Singular Value Decomposition (SVD) Rajaraman et al. (2012) dimensionality reduction technique to factorize the matrix of all descriptive geometric features into component singular values and matrices. We then used only the first singular value of the decomposition to obtain a one-dimensional representation of the geometric robot feature-space, which allowed us to represent each design by a single number that reflected all computed geometric features into one quantity. This reduced-dimensional representation of robot morphologies will henceforth be referred to as the latent factor representation of a particular robot's morphology, or the *latent factor* in short.

Results

A sample of designs created by participants can be seen in Fig. 3. The T-shaped robot design in the center was the highest performing design overall (the best distance it was able to achieve was approximately 32 meters).

Summaries of team contribution are indicated in Table 1.

The distribution of distances achieved by the social group and that of the independent group can be seen in Fig. 4. The social group achieved higher distances than the independent group at a level that was statistically different ($p < 0.0001$; Kolmogorov-Smirnov test, $D = 0.14151$).

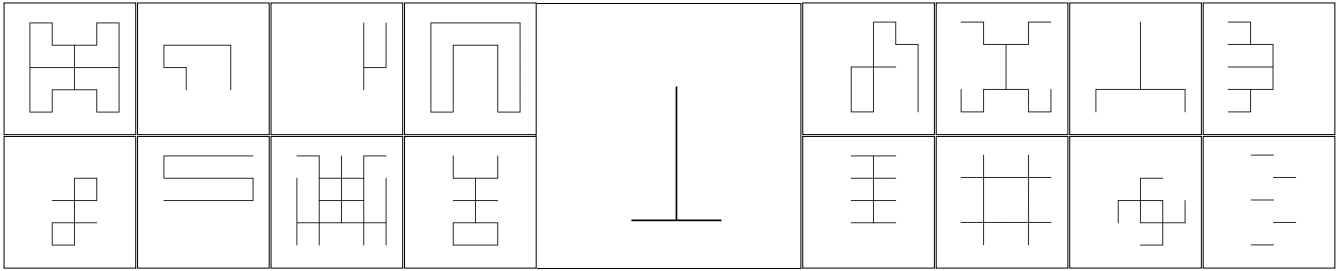The minimum, mean and maximum values of the latent

Figure 3: A sample of robot designs by participants. The highest performing robot (with regard to distance-traveled) was the T-shaped robot in the center of the designs shown.

	Independent	Social
Total Contributions	2825	2984
Total Unique Designs	1245	1136
Number of Participants	364	398

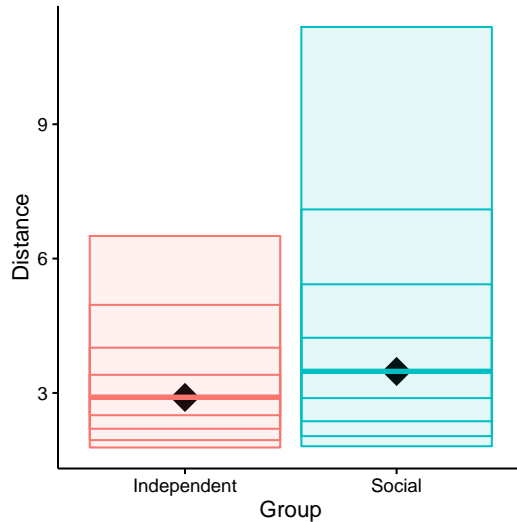Table 1: Social and independent group contributions.

	Independent (min/mean/max)	Social (min/mean/max)
Number of Limbs	(0/0.734/12)	(0/0.656/12)
Symmetry	(0/0.882/1.00)	(0/0.945/1.00)
Latent Factor	(0/0.880/1.10)	(0/0.942/1.11)

Table 2: Latent factor range and ranges of values used to compute latent factor in designs by both groups.

Figure 4: Deciles of distances achieved by robot designs in the group working together (Social) and those achieved by individuals working in isolation (Independent). The median distance value is indicated by a black diamond.

factor and the quantities that are used to compute it are shown in Table 2.

The dimensionality reduction technique resulted in two non-negligible components (coefficients > 0.00001) that contributed to the latent factor. These contributions were the robot's *number of legs* (coefficient of 0.01) and the *symmetry* of the robot body (coefficient of 0.99). Values for symmetry could range from a minimum of no symmetry (0.0) to a maximum value of 1.0, indicating perfect symmetry about at least one of the horizontal, vertical or diagonal reflective

axes. The minimum number of legs found in a design was 0, representing designs whose segments were all connected at both ends to up a maximum value of 12 legs.

There was not a significant difference in the symmetry of the designs created by the independent group compared to those created by the social group ($p = 0.132$; Kolmogorov-Smirnov test, $D = 0.047829$), nor was there a significant difference between the distribution of the number of legs in designs created by the social group and the independent group ($p = 1.0$; Kolmogorov-Smirnov test, $D = 0.0082534$). However, there was a significant difference in the values for latent factor when comparing the distribution of designs created by the independent group and the social group ($p < 0.01$; Kolmogorov-Smirnov test, $D = 0.068714$).

The 5 designs with the highest latent factor value can be seen in Fig. 5.

Discussion

The only difference between the social group and the individual group was that the social group was able to see designs created by others in their group. Thus, if a quantity derived from the designs was more prevalent in the social group versus the independent group, then this quantity was the result of social dynamics.

We derived a measurable latent factor from designs created by each group. We found that this latent value was significantly more prevalent in the social group than that in the individual group at a statistically significant level. Thus exposure to others' designs resulted in increased prevalence of this factor.

	Latent Factor: 1.11 Symmetry: 1.0 # Legs: 12
	Latent Factor: 1.11 Symmetry: 1.0 # Legs: 12
	Latent Factor: 1.10 Symmetry: 1.0 # Legs: 11
	Latent Factor: 1.10 Symmetry: 1.0 # Legs: 11
	Latent Factor: 1.09 Symmetry: 1.0 # Legs: 10

Figure 5: Crowd designs with the highest latent factor values and corresponding symmetry and number of legs calculations.

A synergy is, by definition, a collective outcome that is greater than the sum of individual contributions. Demonstration of a synergy is not predicated on the collective outcome being constructive with respect to the objective of the efforts. However, there is evidence that the latent factor that we have distilled contributes constructively to robot design.

This latent, socially-transmitted value incorporated morphological symmetry, which has been shown to be beneficial for robot locomotion (Bongard and Paul (2000)). While the incorporation of symmetry in user designs alone was not significantly different in the social versus independent groups, the associated p-value ($p = 0.132$) indicates a trend that those in the social group may have favored symmetry in designs over those working independently. It is only by the incorporation of the number of legs in the design that differentiates the social tendency of design with the isolated design tendency.

It appears that there is a latent quantity that – through exposure to designs by other users – is increasing, consciously or otherwise in the social participants. And the social group does indeed design robots that, on average, outperform those designs by participants in the individual group. However, we cannot say with confidence that it is this latent value that leads to the improved performance. There may be other, undiscovered factors that lead to the superior performance by the social group. However, we did find that there is a positive correlation between the discovered latent factor and

the distance that a robot is able to travel (Pearson correlation $= 0.2546499$). Thus, while we cannot say for certain that we found *the* latent factor that contributed to the success of the social group, we can say that – through the distillation of design decisions by a large group of non-expert contributors – we found a quantity that correlates with the desired problem outcome. And that this quantity is, in part, corroborated by findings scientific literature (Bongard and Paul (2000)) on locomotion unbeknownst to the non-expert participants in the study.

The social group designed robots that were capable of moving significantly farther than those designed by individuals in isolation. As can be seen in Fig. 4, the distribution of distances achieved the social group's robots have a higher median value than the independent group. The distribution of distances achieved by the social group has a much higher spread in the upper deciles than the independent group. Here we are reporting the overall ability of the participants in the social group to the ability of the independent group to design robot bodies in tandem with their control strategy. This is in contrast to the performance of the robot morphologies that are independent of their control strategies as described in (Wagy and Bongard (2015a)). Thus users were able to work together to build robot body/brain combinations that outperformed the body/brain combinations of those that worked alone. This need not have been the case: well-known group pathologies such as groupthink (Janis (1972)) could have resulted in an echo-chamber effect. Users working together could have missed promising design possibilities due to a focus on limited regions of the space of possible designs. Also, the number of participants working collectively in this study (398) far exceeds the number of participants considered optimal for effective social interactions Dunbar (1992).

However, there could be a trivial reason for the social group's superior design performance. As described in the Methods section, a hill-climber was assigned to each robot morphology; and each hill-climber instance was shared between members in a group. It is possible that the designs that received the most attention by the social group simply were given more attempts at finding a good control strategy by devoting more hill-climber iterations to the search for a good controller. Thus groupthink may have led the social group to focus on a single design's control strategy rather than concentrating efforts on finding an optimal morphology. If this had been the case, we would have seen that the designs with the most hill-climber iterations are also the highest performing designs. However, this is not what we see in the results. Referring to Fig. 6, we do see that several designs in the social group benefited from many more hill-climber iterations than the independent group. But these designs were not among the high-performing instances. In contrast, many of the highest-performing designs are those that received 10 or less hill-climber iterations. This is much less than some designs, which received in excess of 100 iterations devoted

to finding the best controller. Thus we can see that the robot morphology – not just the number of hill-climber iterations – was an important component of the robot's performance. Therefore we believe that the social group did not rely solely on the focused search efforts for the best control strategy on a limited groups of designs.
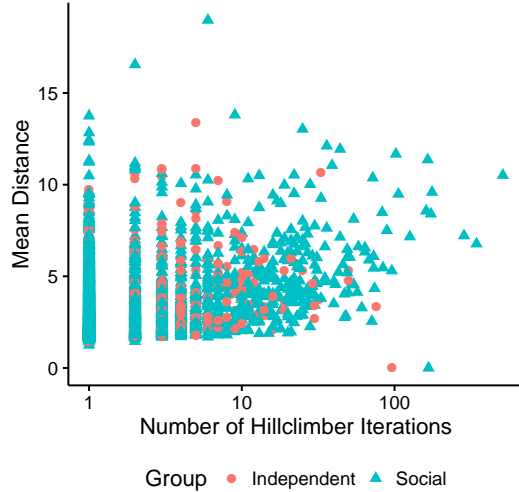

Figure 6: Mean distance achieved by each robot design compared to the number of hill-climber iterations that that design received. A number of designs in the social group did receive substantially more iterations than those in the independent group, but they were not high-performing designs. In contrast, some of the highest-performing designs received very few iterations.

We claim here that the latent factor discovered is an indicator of synergy in the social group. This value is significantly more prevalent in the social group versus independent group. And the only difference between the two groups was the opportunity to synergize. Therefore the only ways that the latent factor could arise is through synergy or by chance. It is unlikely that the value arose by chance, as indicated by the statistical tests performed. However, it has been shown that crowdsourced work follows a heavy-tailed distribution (Swain et al. (2015)). Most participants contribute very little to a crowdsourced activity and a single user or small group of users contributes vastly more than others. This study followed that same trend (see Figure 7). Therefore, it is possible that by chance the top contributing user in the social group favored the latent factor and biased the overall group towards prevalence of this value.

We investigated whether the designs by the top-contributing participant in the social group biased the comparison of the latent factor between the groups. We removed the social group member that contributed the most designs from the social group and compared the mean latent factor

values for users in each group. Even without this top contributor, we found significantly more of the latent factor in the social group versus the independent group ($p = 0.011$ Kolmogorov-Smirnoff test).
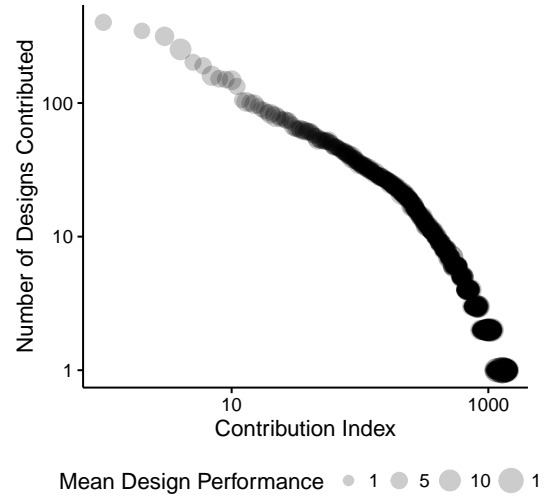

Figure 7: Contributions to this study follow a heavy-tail distribution. The number of designs contributed by most users was small whereas the number of designs contributed by one very enthusiastic user was very high.

Note that the designs with the highest scores of the latent factor are those that maximize both the symmetry measure – a maximum of 1.0 – and the number of legs. The maximum number of legs found in designs created by the crowd was 12. 12 legs is also the maximum number of legs possible when designing single-component, connected robots in this design space. In fact, 8 unique members of the social group drew designs that maximized the number of legs quantity; whereas no users in the individual group drew designs that maximized this quantity. The maximum number of legs found in the independent group was 11. Thus, the designs that maximize the latent factor (Fig. 5) are those with the most legs possible. However, despite having the highest possible symmetry score, the top performing design of all designs only has 3 legs (the largest, T-shaped design in Fig. 3). We cannot say that the best design was the result of this increased latent factor in the social group. Whether the prominence of the latent factor influenced the creation of the best designs requires further exploration. Future work will address the progression of features that lead to specific, optimal outcomes such as the T-shaped robot in Fig. 3.

We can get a sense of the participation of users that contributed to the top designs by looking at Table 4. Repeating numbers across rows in the table indicate that the same user contributed multiple hill-climber runs to a particular design. Two patterns can be seen in these top 10 designs. The top

design (Design Rank = 1) ranking design was created by a user that only contributed 9 runs total to the overall experiment. Similarly, the first five contributions of the sixth and eighth ranking designs were by the same user. However, in almost all other designs in these top 10 designs (with just one exception: the seventh ranked design), we see that the designs were created by a user with lower numbers of total contributions and then picked up by users who contributed more overall to the experiment. For example, the second best design was created by a user who contributed just 4 runs to the experiment. Then a user that contributed 6 runs picked up the design and then a user that devoted 25 runs drew that same robot to contribute a run to the hill-climber. This pattern of contributions could be the result of various social dynamics. It may be that a design that is initially promising may have caught the attention of those users that are more participatory. Or this could be a general pattern of behavior for any design created socially. However, if we examine the same table for the worst-performing designs (Table 3 the pattern is not as prevalent. But we do see this pattern in the eighth and ninth worst designs. Future work will investigate these social dynamics that contribute to specific robot designs.

Rank	User 1	User 2	User 3	User 4	User 5
1	9	9	9	9	9
2	4	6	25	25	25
3	3	17	17	88	88
4	6	29	52	52	52
5	52	52	52	52	316
6	28	28	28	28	28
7	17	14	14	14	14
8	11	11	11	11	11
9	8	8	8	83	79
10	10	13	75	75	75

Table 3: Total number of runs contributed by each of the first 5 users to work on the top ranking designs. Repeating numbers indicate the same user contributing runs to the design. The best design is at the top and the tenth best design is at the bottom.

Additionally, the mean value of this latent factor over the course of time (as measured in number of designs contributed by each group) can be seen in Fig. 8. This figure strongly suggests a slow increase in the social group's usage of the latent factor, whereas it appears that this value fluctuates over time for the independent group. More data is required to evaluate whether this trend continues in order to verify that this is indeed a trend; but the steady increase of this quantity in the social group is suggestive.

Conclusion and Future Work

In this study, anonymous Web participants designed robots in their browsers. Our hypothesis was that there exists a

Rank	User 1	User 2	User 3	User 4	User 5
10	133	133	133	133	133
9	38	148	148	148	148
8	23	79	79	79	79
7	7	4	6	6	15
6	35	35	35	35	35
5	50	50	50	50	50
4	23	23	23	10	252
3	52	52	52	52	52
2	46	46	46	46	46
1	41	41	41	41	41

Table 4: Total number of runs contributed by each of the first 5 users to work on the bottom ranking designs. The worst design is at the bottom and the tenth worst-performing design is at the top.
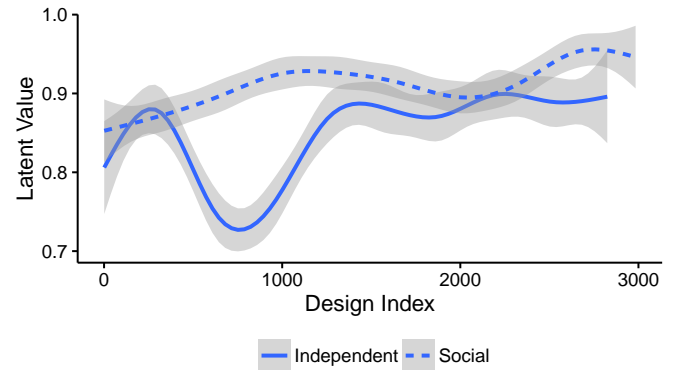

Figure 8: Mean value of latent factor over time (index over designs created by participants). Error bars indicate 95% confidence levels. Participants creating designs socially maintained a near constant increase in the latent value, whereas participants working independently varied their incorporation of this value in their designs.

measurable quantity derived from the social design of adaptive machines. Using the designs created by study participants, we derived a latent geometric quantity through a dimensionality reduction technique. We compared the prevalence of this derived latent value in the social and individual groups. We found that this value was more prominent in a group of participants working socially than those working in isolation. We observed that this value followed an increasing trend in the social group designs whereas the quantity fluctuated in the designs created by the independent group.

The latent value that we uncovered is derived from the symmetry and number of limbs in robot designs. That this value was derived in part from symmetry corroborates previous work on social design of adaptive machines over the Web. The latent value was shown to have a positive correlation with the desired outcome in robot designs, which was

that of fast forward locomotion. Thus, the social quantity may have played a role in the superior outcome in social design of robots. Further work is required to demonstrate how such aggregate social quantities influence specific designs, such as those that are among top performers.

Deriving such measurable social quantities can be useful for their incorporation into automated methods. In future work, we will use crowdseeding (Wagy and Bongard (2015b)) to enable machine design of robots by incorporating the SVD-derived objective into a design objective.

Additionally, we would like to utilize the methodology introduced here to analyze crowd designs in order to provide immediate feedback to the crowd. By providing feedback to the crowd during the social design process, we may be able to capitalize on crowd preferences and biases in real-time to thus accelerate the search process.

This technique may be useful more broadly in social and human computing. But understanding whether social design preferences are destructive or constructive in other domains is critical to determining the general utility of such methods.

Acknowledgments

References

Bongard, J. C. and Paul, C. (2000). Investigating morphological symmetry and locomotive efficiency using virtual embodied evolution. In *From Animals to Animats: The Sixth International Conference on the Simulation of Adaptive Behaviour*.

Couzin, I. (2007). Collective minds. *Nature*, 445(7129):715–715.

DiMaggio, P., Hargittai, E., Neuman, W. R., and Robinson, J. P. (2001). Social implications of the internet. *Annual review of sociology*, pages 307–336.

Dunbar, R. I. (1992). Neocortex size as a constraint on group size in primates. *Journal of Human Evolution*, 22(6):469–493.

Gonçalves, B., Perra, N., and Vespignani, A. (2011). Modeling users' activity on twitter networks: Validation of dunbar's number. *PloS one*, 6(8):e22656.

Gowers, T. and Nielsen, M. (2009). Massively collaborative mathematics. *Nature*, 461(7266):879–881.

Howe, J. (2006). The rise of crowdsourcing. *Wired magazine*, 14(6):1–4.

Janis, I. L. (1972). *Victims of groupthink: A psychological study of foreign-policy decisions and fiascoes.* Houghton Mifflin.

Khatib, F., Cooper, S., Tyka, M. D., Xu, K., Makedon, I., Popović, Z., Baker, D., and Players, F. (2011). Algorithm discovery by protein folding game players. *Proceedings of the National Academy of Sciences*, 108(47):18949–18953.

Lintott, C. J., Schawinski, K., Slosar, A., Land, K., Bamford, S., Thomas, D., Raddick, M. J., Nichol, R. C., Szalay, A., Andreescu, D., et al. (2008). Galaxy zoo: morphologies derived from visual inspection of galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189.

Moore, J., Clark, A., and McKinley, P. (2014). Evolutionary robotics on the web with webgl and javascript. *arXiv preprint arXiv:1406.3337*.

Page, S. E. (2008). *The difference: How the power of diversity creates better groups, firms, schools, and societies.* Princeton University Press.

Parameswaran, M. and Whinston, A. B. (2007). Social computing: An overview. *Communications of the Association for Information Systems*, 19(1):37.

Quinn, A. J. and Bederson, B. B. (2011). Human computation: a survey and taxonomy of a growing field. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1403–1412. ACM.

Rajaraman, A., Ullman, J. D., Ullman, J. D., and Ullman, J. D. (2012). *Mining of massive datasets*, volume 1. Cambridge University Press Cambridge.

Swain, R., Berger, A., Bongard, J., and Hines, P. (2015). Participation and contribution in crowdsourced surveys. *PloS one*, 10(4).

Von Ahn, L. (2009). Human computation. In *Design Automation Conference, 2009. DAC'09. 46th ACM/IEEE*, pages 418–419. IEEE.

Wagy, M. and Bongard, J. (2014). Collective design of robot locomotion. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 138–145.

Wagy, M. D. and Bongard, J. C. (2015a). Combining computational and social effort for collaborative problem solving. *PloS one*, 10(11):e0142524.

Wagy, M. D. and Bongard, J. C. (2015b). Crowdseeding: a novel approach for designing bioinspired machines. In *Biomimetic and Biohybrid Systems*, pages 293–303. Springer.

Wang, F.-Y., Carley, K. M., Zeng, D., and Mao, W. (2007). Social computing: From social informatics to social intelligence. *Intelligent Systems, IEEE*, 22(2):79–83.

Listening to a world transformed: Perception in an inverted acoustic field.

Fernando Bermejo[1,2,3], Ezequiel Di Paolo[4] and Claudia Arias[1,2,3]

[1]Centre for Research and Transfer in Acoustics, Unit Associated of CONICET, UTN - FRC, Córdoba, Argentina
[2]Faculty of Psychology, National University of Córdoba, Córdoba, Argentina
[3]Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
[4]Ikerbasque, Basque Foundation for Science, Spain
fbermejo@psyche.unc.edu.ar

Research into perceptual and behavioural adaptations to radical disruptions of the agent-environment coupling has long been an interest of dynamical and embodied agent-based modeling. Work in evolutionary robotics has produced a series of minimal models of homeostatic adaptation to inversions of the sensory field (e.g., Di Paolo, 2000a, Iizuka and Maeda, 2013) as broad replications of classical experiments by Kohler (1964) and others on adaptation to wearing goggles that invert or distort vision.

While these models draw inspiration from Kohler's experiments, their sensorimotor instantiation is rather minimal, typically involving two point photoreceptors and a point source of light. This sensorimotor space is arguably a better match for sensorimotor engagements in the auditory, rather than the visual, modality (e.g., Di Paolo, 2000b).

However, unlike the striking visuo-motor adaptation shown by Kohler's participants, empirical evidence for similar kinds of adaptation to radical disruptions of the auditory space, such as inversion of the left and right directions, has not yet been found. This is in part due to the technical difficulties involved, which make these studies rather scarce, but also possibly because of the kind of sensorimotor relations and plasticity at play in auditory perceptual learning.

Here we report on the development and a series of preliminary studies regarding the role of activity and passivity in human adaptation to wearing a left-right auditory inversion device, or *pseudophone*.

From an enactive perspective, perception is intimately related to action (Varela et al 1991). Perception is constituted by the active skillful use of the regularities that govern the ongoing coupling between motor and sensory activity, also known as sensorimotor contingencies (SMCs) (O'Regan and Noë, 2001). Perceptual adaptation, in this context, involves the ongoing equilibration of sensorimotor schemes (Di Paolo et al 2014). Sensorimotor disruptions present the perceiver with radical obstacles and lacunae as her sensorimotor skills suddenly cease to make sense. The process of re-learning sensorimotor schemes for particular actions provides rich information regarding the complex coordinations involved, which because they are nearly maximally equilibrated, are not always obvious during normal sensorimotor engagements.

One of the key recurring elements found in empirical and modeling studies of radical adaptation is the need for self-generated activity by the agent. New sensorimotor schemes cannot be learned unless the agent engages the world actively and confronts various breakdowns and tries to recover from them. This is clear in Kohler's experiments. During prolonged wearing for left-right inversion goggles, participants initially feel baffled and their movements induce unexpected visual changes as if aspects of the visual scene expected to remain static moved without correspondence to the action. This is due to the sign-inversion introduced by the goggles such that extra-retinal proprioceptive signals cease to compensate the retinal flows provoked by the static background of a visual scene as the head turns right or left. As participants gradually adapt in specific contexts their movements cease to produce the sensation of background instability.

Interestingly, in the case of inversion of auditory space using a pseudophone perceptual instability also appears when participants move. This was shown in a study of localization of sounds with this device (Ohtsubo, et al., 1982). The authors, by allowing participants move their head during a brief activation of the sound sources, found that the variability in their responses increased significantly. Participants reported that when they moved they felt that (static) sound sources also did. Young (1928) and Hofman et al. (2002) also commented a similar effect. Beyond these few studies there has not been much systematic analysis of the motor strategies used by participants when wearing these devices.

To further investigate movement-induced de-stabilitzation, we evaluate the experience and performance of participants equipped with a pseudophone in two sound localization tasks. Our device works in two modes, with or without inversion of sound signals (figure 1A). This allows us to compare experimental performance under each condition. All testing protocols are conducted without visual cues in a sound proof room.

First, in Task 1 we analyze different situations where variations to the sound stimulation are the same, but in one case this variation is obtained passively and in the other actively. In the passive motion condition the participant remains still and the sound source moves. In the active motion condition the source remains still and the participant moves. Figure 1B shows an example of one of the pairs of movements evaluated. The stimuli were metronome clicks (160 beats per minute).

The participant's experience of the same changes in sound stimulation was different depending on the pseudophone mode. Without inversion there are no reported differences between the active and passive conditions. Participants expressed they were able to recognize the position of the sound source. In the "inversion" mode the two conditions result in different experiences. In the passive condition perceptual experience is similar to the non-inversion case,

except that the source is perceived as moving in the opposite direction. In the active condition participants experienced strange sensations. For instance, in the case of rotational movement of the head (figure 1B) participants reported that sound changes were more abrupt, faster and less predictable than in the passive motion condition.
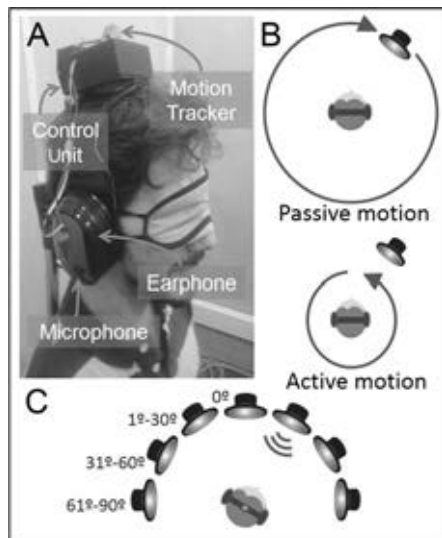

Figure 1. (A) Participant wearing the Pseudophone. (B) Diagram of Task 1 with passive and active movements. (C) Diagram of sound localization task (Task 2).

These qualitative differences in the case of auditory inversion reveal a proprioceptive component in spatial hearing. If in the passive condition a rotational movement of the source from left to right is perceived as a movement from right to left, in the active condition the head movement is added to the apparent movement of pure auditory sensations instead of subtracted from it, which apparently gives the feeling of a source moving at a greater speed.

In Task 2, we also investigated passive vs. dynamical source localization. In this task, participants remained seated in front of 7 speakers in the horizontal plane (figure 1C). When a source is activated the participant has to turn her head and face it. In the passive condition, participants listen to a very short sound, a 250 ms pulse of pink noise, from one of the sources and then have to face the source; and in a dynamic condition, the sound source remains activate with a burst of pink noise until the participant locate it. We measure the accuracy of the location responses and pattern of movements. Phenomenal data is also recorded.

Without inversion participants perform accurately in both conditions. In the "inversion" mode in the passive condition responses are mirrored in the opposite hemifield to the sound source, e.g. if the sound source is +45° the participant moves close to -45 °. By contrast, in the dynamic condition, when participants could freely explore the sound environment, hit levels are similar to those obtained in the mode "without inversion". In this condition, participants spontaneously develop sensorimotor strategies that help them resolve the conflictive information. To respond, the strategies used at the beginning involve large amplitude movements, sweeping all the frontal plane and then a series of smaller movements to refine the position.

Participants mention that their movements provoked them the feeling that the sound source also moved in unexpected ways. There produce comments like, *when I turn the head to face the sound, it escapes very quickly*. This sensation happened when the participant turned the head toward the hemifield opposite the sound source. Conversely, movement toward the source causes the source to "appear" and "disappear" suddenly from the front of the participant and sound intensity varies rapidly. Participants commonly report front-back confusions. As the person starts to move, the sound source in the frontal plane is sometimes perceived in the opposite position on the backplane. This phenomenon is probably because the SMCs enacted are similar to those that usually used to locate sounds in the backplane.

The de-stabilization of SMCs caused by pseudophone allows us to investigate different kinds of sensory activity involved in the auditory system in non-obvious ways. As a next step we plan to investigate a hypothesis drawn from a minimal cognition model by Izquierdo & Di Paolo, (2005) who showed that radical sensorimotor ambiguities (such as whether the sensor array is left-right inverted or not) can be coped with using a single sensorimotor strategy generated by reactive control. Participants trained in the use of the pseudophone under random variations of the inverted/not inverted modes should be expected also to converge to the use of a single sensorimotor strategy valid for both cases.

References

Di Paolo E. A., Barandiaran X.E., Beaton M., and Buhrmann T. (2014). Learning to perceive in the sensorimotor approach: Piaget's theory of equilibration interpreted dynamically. *Front. Hum. Neurosci.* 8:551.

Di Paolo, E. A. (2000a). Homeostatic adaptation to inversion in the visual field and other sensorimotor disruptions. In Meyer, J., et al., editors, *From Animals to Animats VI: Proceedings of the 6th International Conference on Simulation of Adaptive Behavior*, pages 440-449. Cambridge, MA: MIT Press.

Di Paolo, E. A. (2000b). Behavioral coordination, structural congruence and entrainment in a simulation of acoustically coupled agents. *Adaptive Behavior* 8:25–46.

Hofman P. M., Vlamig M. S. M. G., Termeer P. J. J. and van Opstal A. J. (2002). A method to induce swapped binaural hearing. *J. Neurosci. Methods*, 113, 167–179.

Izquierdo, E. and Di Paolo, E. A. (2005). Is an embodied system ever purely reactive? In M. Capcarrere et al., Advances in Artificial Life: Proceedings of the 8th European Conference on Artificial Life, pages 252-261. Springer-Verlag.

Iizuka H., Ando H. and Maeda T. (2013) Extended homeostatic adaptation model with metabolic causation in plasticity mechanism– Toward constructing a dynamic neural network model for mental imagery. *Adaptive Behavior,* 21: 263–273.

Kohler, I. (1964). The formation and transformation of the perceptual world. *Psychological Issues*, 3: 1–173.

Ohtsubo, H., Teshima, T., and Nakamizo, S. (1982) Effects of head movements on sound localization with an electronic pseudophone. *Japan Psychological Research*, 22: 110–118.

O'Regan, J. K., and Noë, A. (2001). A sensorimotor account of vision and visual consciousness. *Behav. Brain. Sci*, 24: 939–973.

Varela, F. J., Thompson, E., and Rosch, E. (1991). *The embodied mind*. Cambridge, MA: MIT Press.

Young P. T. (1928).Auditory localization with acoustical transposition of the ears. *J. Exp. Psychol.*, 11: 399–429.

Robots can ground crowd-proposed symbols by forming theories of group mind

Joey Anetsberger and Josh Bongard

University of Vermont, Burlington, VT 05401
janetsbe@uvm.edu

Abstract

The non-embodied approach to teaching machines language is to train them on large text corpora. However, this approach has yielded limited results. The embodied approach, in contrast, involves teaching machines to ground abstract symbols in their sensory-motor experiences, but how—or whether—humans achieve this remains largely unknown. We posit that one avenue for achieving this is to view language acquisition as a three-way interaction between linguistic, sensorimotor, and social dynamics: when an agent acts in response to a heard word, it is considered to have successfully grounded that symbol if it can predict how observers who understand that word will respond to the action. Here we introduce a methodology for testing this hypothesis: human observers issue arbitrary commands to simulated robots via the web, and provide positive or negative reinforcement in response to the robot's resulting action. Then, the robots are trained to predict crowd response to these action-word pairs. We show that robots do learn to ground at least one of these crowd-issued commands: an association between 'jump', minimization of tactile sensation, and positive crowd response was learned. The automated, open-ended, and crowd-based aspects of this approach suggest it can be scaled up in future to increasingly capable robots and more abstract language[1].

Introduction

Language has been a central concern in Artificial Intelligence research since the field's inception in the 1950s. Like many other aspects of cognition, it has been addressed with non-embodied and embodied approaches. Non-embodied approaches typically train agents on large pre-existing text corpora (Guha and Lenat, 1993; Collobert and Weston, 2008), or on conversations those agents attempt to conduct with humans (Shawar and Atwell, 2003). However, this approach has yielded limited results.

The embodied approach to language acquisition involves helping agents to detect correlations between particular subsets of sensorimotor experiences and categories, to which words can be attached. However, how or whether humans

do this, and how best to enable robots to do this, remains an open question.

The Symbol Grounding Problem

The symbol grounding problem is a long-standing open problem in cognition. It concerns how we can assign meaning to parts of language—symbols—without succumbing to an infinite regress. That is, some symbols must at some point be grounded in something, such as categorical or iconic representations, rather than deriving meaning from other symbols. This is a major problem with the cognitivist approach to intelligence (Harnad, 1990).

Evidence is starting to appear in the literature that suggests that, for humans at least, sensorimotor experience is the 'soil' in which language symbols are ultimately grounded. For example, Pulvermüller and Fadiga (2010) describe how spoken language may be closely coupled to neural circuits related to motor functions. Cangelosi and Harnad (2001) have provided theoretical arguments for *why* sensorimotor experience provides such good grounding for language. First, agents perform 'sensorimotor toil': they acquire knowledge of categories (though not symbols to represent them) through the costly effort of learning through action and feedback. They then perform symbolic theft: these categories are given symbolic representations and shared by those who have performed the necessary toil or who have themselves "stolen" from others. However, exactly how this can be instantiated in machines remains an open question.

Sensorimotor Grounding

Steels (2008) claims that a solution to the symbol grounding problem has been found through the creation of robots who can respond appropriately to human-provided commands. However, it is not clear how such approaches can scale up to more complex robots, large numbers of human tutors, and increasingly abstract language. In this paper we introduce a methodolody that may, in future work, help to support all three. The novelty of our approach in its present form is how it allows people to teach robots aspects of human language of the crowd's choosing.

---

[1] A video summary of the work described here is available at youtu.be/j3sB85ENgA8 and the source code is available at github.com/Janetsbe/AnetsbergerAlife2016Code.

In other work involving non-human languages, Steels has demonstrated a scalable approach to language acquisition among robots (Steels et al., 2002). The robots participate in language games through which they converge on agreement as to the meaning of words generated by the group. Similarly, Schulz et al. (2012) has shown that robots can jointly form words for relative locations and then draw on the combinatorial power of symbols to dictate directions to novel locations by combining these words in novel ways. Whether or how these robot-generated languages could help them understand human languages however has not been addressed. Here, we focus on robots learning human languages.

Crowdsourcing language acquisition.

Despite these initial successes in teaching robots language (or having them teach themselves), there are many open questions that remain. How should robots (or how do humans) ground abstract language in action? Lakoff and Johnson (2008) have argued that embodied metaphors (such as "do not jump to conclusions") hint that we ground even abstract language in sensorimotor experience, but the mechanisms by which this occurs are unclear. How does the acquisition of some symbols facilitate the acquisition of others? Do caregivers spontaneously constrain their utterances to scaffold language learners (Roy et al., 2009)?

These and other questions can only be addressed by scalable, open-ended and automated infrastructure which enables large numbers of people to teach large numbers of (possibly increasingly abstract) language to increasingly complex robots capable of a broadening set of sensorimotor experiences. Here we introduce such an experimental apparatus that relies on crowdsourcing. Through the web, observers propose arbitrary, natural language commands to robots and provide positive or negative reinforcement for the resulting actions. The robots then learn to predict which actions, under which commands, are likely to generate positive crowd reinforcement. Various hypotheses about language acquisition can then be tested using this apparatus. Here, we first test the hypothesis as to whether robots can indeed ground these crowd-proposed symbols by forming theories of their group mind.

Crowdsourcing has recently been exploited for training robots to interact with humans (Breazeal et al., 2013; Toris et al., 2014), but not for grounding language symbols. We have demonstrated that web participants can collectively design and optimize robots, despite the lack of any explicit reward to the human participants for doing so (Wagy and Bongard, 2014, 2015). In other work we have shown that robots can be trained to form theories of mind about an individual human trainer (Hornby and Bongard, 2012), and even disambiguate between two trainers who disagree about how to reinforce the robot's behavior (Bernatskiy et al., 2014). However, in (Hornby and Bongard, 2012) and Bernatskiy et al. (2014) the robots only learned theories of mind about

synthetic trainers: bots who stood in for actual human trainers. Here we demonstrate that, with the right cyberinfrastructure, robots can successfully ground symbols provided by actual human caregivers.

This research was conducted in two phases. First, subjects were recruited to a web interface through which they could issue commands to the robots and reinforce the resulting behaviors. The data generated by this process was then used to train models to predict the crowd's response to a given command and set of actions. See Fig. 1 for a summary of this two-part methodology.

Phase I: Crowd Deployment

In order to allow subjects to see and interact with the robotics simulation, the Twitch.tv[2] video streaming web service was selected. Twitch is a popular internet service for observing others play video games or perform other skilled tasks. Twitch in turn has given rise to "Twitch Plays..." interfaces through which subjects can collectively observe as well as play interactive video games by voting on the next move in the game. It was hoped that the wide appeal and familiarity of this interface would incentivize large-scale participation. In the work described here, we broadcast a live stream from a robotics simulation; subjects could then interact with the robots observed in the stream using live chat (Fig. 2; a video snippet from the deployment can be seen here.).

Phase I Methods

The crowd deployment commenced on October 29th, 2015. The video stream ran continuously and saw use for 22 days. The experiment was terminated then as user traffic had become negligible. During the crowd deployment, subjects were shown a single robotics simulation to collectively interact with through text input. Subjects were allowed to provide candidate commands and reinforcement signals to the simulation (see Table 1 for terminology used throughout this paper). Candidate commands were strings representing votes for what behavior a subject wanted the robot to be evaluated against. These votes were tallied over a three minute interval; the most frequently-issued candidate command at the end of this period became the new, issued command. An issued command was some string dictating how the subjects should reinforce behaviors over the next three minute period. Reinforcement signals were strings indicating whether subjects considered the robot they were viewing to be obeying the issued command ('y') or not ('n').

Robot Simulation. The simulation was developed in the Unity3D[3] engine, with its default graphics renderer, physics engine, and collision solver. It consisted of a scene containing all elements of the simulation: a floor, start loca-
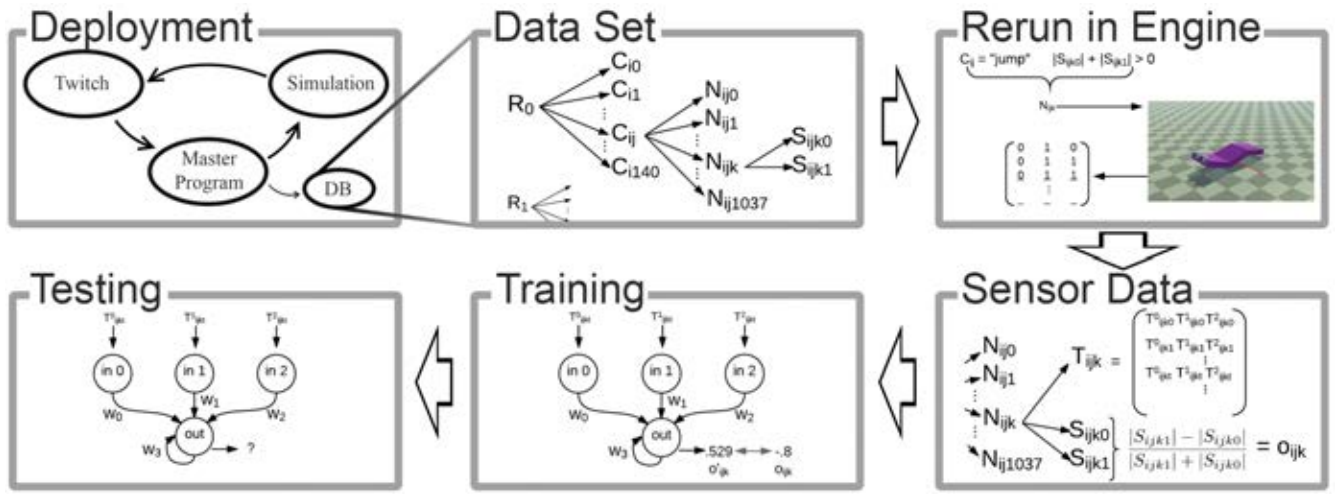

Figure 1: Summary of the methodology. **Deployment**: A master program generates robot controllers, which animate one robot in a physics simulation after another. The video resulting from the simulation is piped in real time to www.twitch.tv, where the human subjects issue the robots commands and reinforcement. **Data Set**: The simulations generate a data set comprised of the robots themselves (R), the commands issued to them (C), the controllers run on those robots under those commands (N), and the reinforcement signals provided by the subjects (S). **Rerun in Engine**: All robots which had been issued the command "jump", and received at least one reinforcement signal, were re-simulated. **Sensor Data**: The resulting touch sensor data (T) was recorded and added to the data set, along with the normalized reinforcement signal (o). **Training**: The time series touch sensor data for a randomly-selected half of these controllers were employed to train a recurrent neural network to predict (o'_{ijk}) the crowd's actual response (o_{ijk}) to each controller using CMA-ES. **Testing**: The ability of the trained RNN to ground the symbol "jump" was tested by measuring its ability to predict crowd response to the other half of the controllers.

tion marker, a robot, a camera, and GUI elements providing users with instructions and feedback. Fig. 2 reports a typical scene during deployment. All data pertaining to the robot, its controller, and GUI elements were sent to the simulation by the master program. The simulation ran more or less continuously for the 22 days. The simulation was continuously recorded and sent as a live video feed to Twitch.tv.

The simulation featured two robot types (Fig. 3). The simple robot was formed of three rigid bodies and two rotational hinge joints. Each body segment was $2.5 \times .6 \times 3.0$ units in size. The simple robot's joints rotated through the sagittal plane, thus restricting movement to forward and backward motion. However, due to asymmetries in collision resolution, the simple robot had the ability to slowly turn, so constraints were imposed in the simulation to frustrate turning. The complex robot consisted of seven segments (three body segments and four legs) and six hinge joints, and was allowed to move about the horizontal plane. Its body segments and leg segments measured $1.2 \times .4 \times 3.0$ and $.6 \times .4 \times 3.0$ units respectively. Both robots were equipped with nonfunctional eyes to provide subjects with a robot-centric frame of reference through which to issue commands (e.g. 'move forward').

During the 22 days of deployment, the two robots alternated every hour: the crowd saw the simple robot for an hour, then the complex robot for an hour, then the simple

robot again, and so on. During each hour period, the command issued to the robot changed every three minutes.

During each three minute period, all candidate command votes were counted, recorded, and cleared. The most frequently-input candidate command during this period was issued to the robot for the next three minute period. Within a three minute period, six random controller were evaluated on the robot, each for 30 seconds. We henceforth refer to each of these as a 'robot evaluation'. The total number of positive and negative reinforcement signals issued in response to each robot evaluation was recorded.

A robot's controller was defined as an $N \times 3$ matrix with dimension sizes corresponding to the number of joints and three parameters which dictated a given joint's amplitude (α), frequency (β), and phase offset (γ). Thus, each joint at each time step t was issued a desired angle $\alpha \sin(\beta t + \gamma)$, where $\alpha \in [.6, 1.5]; \beta \in [.01, .09];$ and $\gamma \in [-10\pi, 10\pi]$. These constants were drawn from these ranges randomly using a uniform distribution and populated the matrix for each controller. The ranges were selected to maximize variation in behavior. Thus, the subjects observed random behaviors (rather than evolved or learned ones) in this study.

The robot's color was changed whenever the controller was changed. The colors cycled through blue, orange, and violet, and then back to blue. Subjects were instructed to indicate the color of the robot that they were reinforcing. For
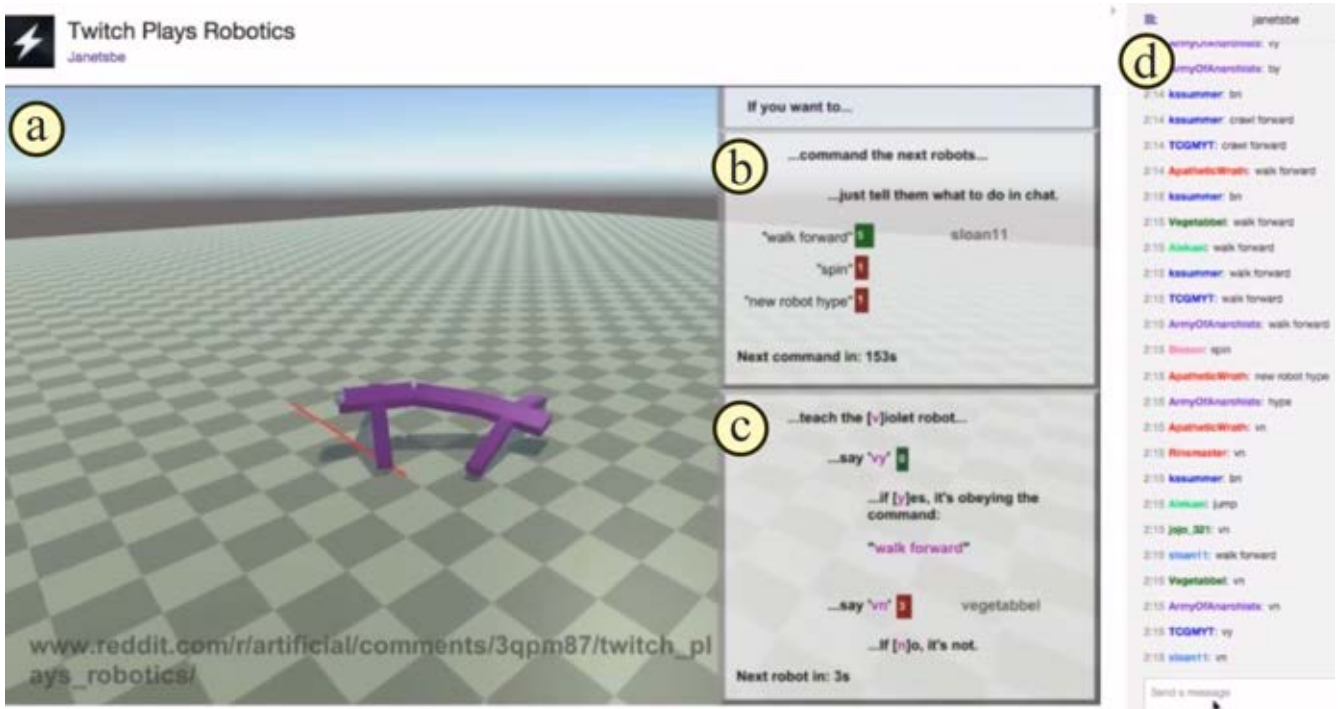
686

Figure 2: The interface, as seen by participants interacting with the simulation. **a)** The simulation video feed. **b)** A panel prompting participants to command what the robot should do next. At this point, one subject has proposed 'walk forward' as the next command, and two other subjects have voted for this. After a five-minute period, the most popular command takes effect. **c)** A panel prompting users to provide positive ('y') or negative ('n') reinforcement for the current action under the current command, which here is 'walk forward'. Votes for either signal allow subjects to see how others are reinforcing this action. **d)** Twitch's chat interface, through which subjects send commands, reinforcement, or other miscellaneous chatter.
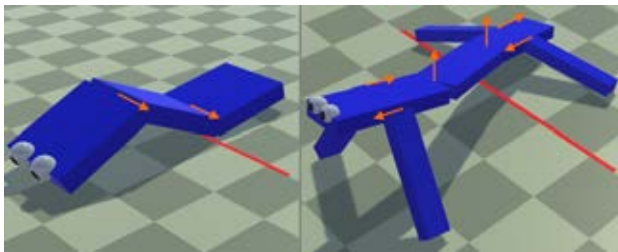
Figure 3: The "simple" (left) and "complex" (right) robot body types. Body segments were connected through hinge joints with normals as indicated by the vectors.

example if the user wished to positively reinforce the blue robot, she would type 'by'. If she wished to negatively reinforce it, she would type 'bn'. Because video streams broadcast through Twitch are delayed for approximately ten to twenty seconds with variation between viewers, this coded input allowed for correct assignment of reinforcement to a controller despite the delay. That is, subjects likely observed a controller (and provided reinforcement to it) after it had already terminated on the host computer.

The simulation's GUI provided directions and feedback to the subjects. In the command panel, the subjects were told: "If you want to command the next robots, just tell them what to do in chat". In the reinforcement panel, the subjects were told: "If you want to teach the [v]iolet [or [o]range or [b]lue] robot, say 'vy' if yes, it's obeying the command: [command]. Say 'vn' if no, it's not."

Video Streaming and Chat Capture. Video streaming was done on Twitch.tv, an internet video streaming service. The service allows a host user to broadcast a live video stream to an arbitrary number of internet users who may (optionally) talk to each other and the host through an embedded IRC service. Twitch was used due to its established user base, chat functionality, available API, as well as the existence of the previously mentioned "Twitch Plays" phenomenon.

Video streaming was done using FFSplit[4] to capture and encode video of the simulation being run on the host machine and stream it to Twitch, where it was viewed by subjects. Subjects saw this simulation and were able to interact with it through Twitch's chat service. Subjects were

---

[4]http://www.ffsplit.com

687

Term	Definition
Subject	Any Twitch user who provided either a reinforcement signal or candidate command.
Robot type	$R_i, i \in \{0, 1\}$, The simple (R_0) and complex (R_1) robots.
Candidate command	A string entered by a participant. Considered a vote for a future issued command.
Issued command	C_{ij}. The jth [c]ommand the ith robot was evaluated against.
Controller	N_{ijk}. The kth co[n]troller issued to the ith robot under command j. A matrix of coefficients used to dictate a robot's movements.
Robot evaluation	The 30-second simulation resulting from a controller issued to a robot under a given command. Six such evaluations were performed for each issued command.
Reinforcement signal	S_{ijkl}, for $l \in \{0, 1\}$. The number of negative ($l = 0$) and positive ($l = 1$) reinforcements collected from subjects in response to the kth controller under the jth command.
Normalized reinf. signal	$o_{ijk} \in [-1, 1]$. $o_{ijk} = \frac{|S_{ijk1}| - |S_{ijk0}|}{|S_{ijk1}| + |S_{ijk0}|}$
Touch sensor data	T_{ijk}. Touch sensor data generated by robot i under command j with controller k.
Predictive model	A recurrent neural network (RNN) which predicts o_{ijk} given T_{ijk}. Its output, o'_{ijk}, is an estimation of the crowd's response to robot evaluation ijk.

Table 1: Terminology used throughout this paper.

allowed to provide any combination of characters to the chat. Any string of length greater than two or less than thirty that were not part of Twitch's default filtered word list were considered candidate commands. Subjects could provide commands and reinforcement at any time.

Subjects provided reinforcement though specific coded input. This consisted of strings of length two which matched the pattern of $(b|o|v)(y|n)$ where the first character represented the color of a robot (**b**lue, **o**range, or **v**iolet), and the second either **y**es or **n**o. Further, only reinforcement signals corresponding to the current or immediately previous robot evaluation were counted. For example, if the current evaluation's color was blue and the previous evaluation had been orange, the following strings would have been parsed as reinforcement signals: 'on', 'oy', 'bn', and 'by'.

Recruitment and Incentivization. Subjects were recruited through Reddit, a popular internet message board site consisting of hundreds of sub-sites (called subreddits) which generally limit posts to a specific topic. Posts were issued to relevant subreddits directing subjects towards this

main post on the `artificial` subreddit[5]. Here, subjects were directed to an agreement page which sent them to the twitch channel[6].

Subjects were incentivized to interact with the system by adding features to the GUI which were meant to provide the subjects with a sense of involvement with the simulation. This was done by displaying subjects' user names when valid input was given. This showed subjects that their input was being counted and used; their actions had an impact. For the scope and scale of this study, this seemed sufficient.

Phase I Results

The crowd deployment lasted 22 days. During this time, at least 6,388 robot evaluations were seen by hundreds of subjects, who provided hundreds of commands and thousands of reinforcement signals. Table 2 reports the relevant values.

General Figures	Values
Subjects	424
Robot evaluations sent	57,108
Robot evaluations observed	$\geq$ 6,388
Subject inputs	16,449
Commands	**Values**
Candidate commands entered	8943
Candidate commands/subject	$\approx$ 21.1
Distinct commands issued	266
Most frequently issued commands	jump (385) walk forward (58) move forward (41) run (26) crawl forward (20).
Reinforcement signals	**Values**
Reinforcement signals entered	7503
Mean reinforcement signals/eval.	$\approx$ 1.18
Mean reinf. signals/subject	$\approx$ 17.7
Proportion of positive reinforcement	$o = 0.28$

Table 2: Crowd deployment participation results.

Participation produced a data set in which 6,388 controllers received at least one reinforcement signal. From this data set, no significant differences were found between how the crowd interacted with the simple and the complex robots: both had similar distributions in their spread and frequency of commands, and both received similar consistency of reinforcement.

Phase II: Offline Learning

In the second phase of this experiment, models were trained to ground symbols by learning relationships between com-

---

[5] www.reddit.com/r/artificial
[6] www.twitch.tv/janetsbe

mands, sensor data, and reinforcement.

The most commonly issued command for both robots was "jump": 1072 simple and 698 complex robot controllers were evaluated under this command. Data belonging to robots given this command were used to test whether the crowd was able to provide trainable input. Due to the on-line nature of the live deployment, some robot evaluations were prematurely terminated and did not remain visible to the crowd for their full thirty seconds. These were discarded, resulting in 1037 simple and 675 complex robot evaluations used during this phase. Since each controller directly determined the robot's behavior for the duration of its run, any controller could be re-evaluated to obtain any additional data not recorded during the live deployment.

Phase II Methods

During this phase, the goal was to determine if there exists features in the "jump" data set which could be used to train a model that predicts the crowd's response. A recurrent neural network (RNN) was employed as such a model (Fig. 1). It was trained such that, when supplied with sensor data generated by a controller, it outputs a successful prediction of the normalized reinforcement signal o_{ijk} (Table 1).

Since the chosen command was "jump", it seemed likely that this command should have some relationship to whether, when and how the robot contacts the floor. For this reason touch sensor data was recorded for these controllers.

All of the controllers evaluated under the command "jump" and which received at least one reinforcement signal in return were then re-evaluated with touch sensors added to the robots. If a body segment touched the ground, its touch sensor recorded a value of 1; otherwise a 0 was recorded. This was stored in matrix T_{ijk} such that element $t_{ijkmn} \in 0, 1$ indicated whether body part n contacted the ground at time step m for controller N_{ijk}. Each of these controllers was re-evaluated for the same duration as its original run during deployment. Since the master program gathering data from the Twitch channel was also controlling the simulation's timing, there was some slight variation in the duration of each robot evaluation, which resulted in slight variations in the time step length. Thus, m ranged between 3350 and 3761 and 3133 to 3719 for the simple and complex robots respectively. During model training and testing, sensor data was thus truncated to $m = 3350$ and $m = 3133$ for the simple and complex robots respectively to make data consistent in shape for each robot type.

Model training. Offline learning was conducted separately for the simple and complex robot types. For each of the two robots, half of their controllers were assigned randomly to the training set. This was repeated 100 times, leading to 100 models trained on 100 partially-overlapping training sets. The RNN associated with the simple robot had three input nodes (corresponding to the number of touch

sensors and body segments) and one output node, with one synapse from each input to the output, and a recurrent synapse on the output node. The RNN trained and tested on the complex robot had seven input nodes and was otherwise identical. A graphical representation of this RNN for the simple robot's data can be seen in Fig. 1.

During training, each controller in the training set for R_i had its sensor data run through the neural network such that the network was updated m times. Then, the output node's value was read out as o'_i, the predicted normalized reinforcement signal for the ith controller based on its sensor data. This was done for each controller in the training set. Using this, the error for the model was calculated using the following objective function:

$$ e \quad = \quad \left(\sum_{i=1}^{N_y} \frac{|o_i - o'_i|}{2N_y} + \sum_{j=1}^{N_n} \frac{|o_j - o'_j|}{2N_n} \right)/2 \qquad (1) $$

Since the normalized reinforcement signals mostly consisted of unanimous negative reinforcement ($o_i = -1$), accounting for 77.7% and 77.8% of robot evaluations for the simple and complex robots respectively, this error function weighed two subsets of the training set equally in calculating error to avoid over-fitting the model to output -1 for any set of sensor values. One subset consists of all robot evaluations with $o_i = -1$ (of which there were N_n) and the other for all others ($o_i > -1$; N_y), which must by definition contain at least one positive reinforcement signal. In doing so, the error calculation weighs both groupings equally and an constant output of $o_i = -1$ for every controller would result in, at best, $e = 0.5$. Error is, then, the sum of differences in each group, averaged, where o is the actual normalized reinforcement signal for each robot evaluation and o' is the model's prediction.

The popular, continuous-value optimization method CMA-ES (Hansen et al., 2003) was used to train RNNs against each of the 100 training sets, for both robots, resulting in 200 runs of CMA-ES. For each run, a random initial solution array was used. Synaptic weights were bounded on $[-1, 1]$, with an initial step size of 0.1. At the termination of each run, the RNN with lowest error was extracted, leading to 100 RNNs for each robot.

Phase II Results

The ability of these 200 models to generalize their predictions to unseen controllers was measured as follows. The mean error of each model when exposed to its testing set was first computed using Eqn. 1. Then these errors for the 100 models for each robot were in turn averaged. The resulting mean errors are reported as 'experiment' in Fig. 4.

In order to determine the accuracy of these predictions, the models were also exposed to the same testing set, but the normalized reinforcement signals in the set were randomly permuted ('permuted control' in Fig. 4). A second
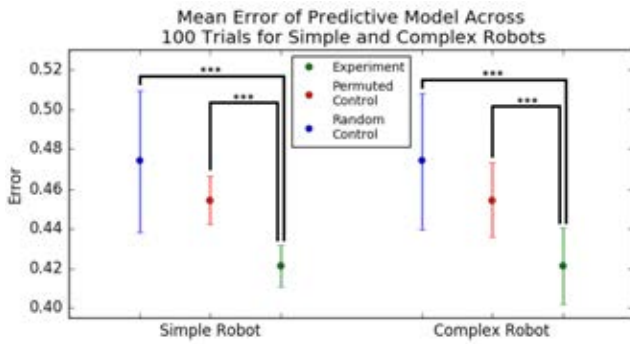
Figure 4: Results from the predictive model on unseen test data, the permuted test data set, and random RNNs on test data. ******* denotes $p < .001$ as reported by a Student's T-test.

control experiment was also conducted in which RNNs with randomly-assigned synapses also made predictions on the same, unpermuted test sets ('random control' in Fig. 4).

Discussion

Fig. 4 shows that both robots employed here were able to successfully ground at least one symbol ('jump') in their own sensorimotor experiences. Further, since this symbol and its meaning were provided solely by human subjects, this suggests that the crowd reached some implicit, mutually agreed upon recognition of this category of behavior in the robots, and that they were able to pass this category on to non-humanoid robots through a simple interface in a short time period, with no explicit reward from the investigators.

Precisely how the robots may have grounded "jump" remains unknown. It may be the case that robots who tended to spend less time touching the ground were more likely to be positively reinforced by the crowd. This hypothesis is supported by the finding that there is a negative correlation between the proportion of time a robot spent with at least one body segment touching the ground and the crowd's normalized reinforcement signal, for both robots (Fig. 5). Despite this, it is unclear whether the models learned this relationship, or instead discovered some more subtle function of the touch sensor data that better predicts the crowd's response. In a similar manner, it is not clear if each robot grounded this symbol in the same way.

Conclusions and Future Work

Here we have demonstrated a unique methodology for enabling robots to gradually acquire language: crowd-generated language symbols are grounded by detecting correlations between sensorimotor experience and crowd-generated reinforcement.

Given the unconstrained nature of the interface, the subjects could have tried to teach the robots unlearnable commands such as 'xyzf' or 'prove Fermat's Last Theorem'.
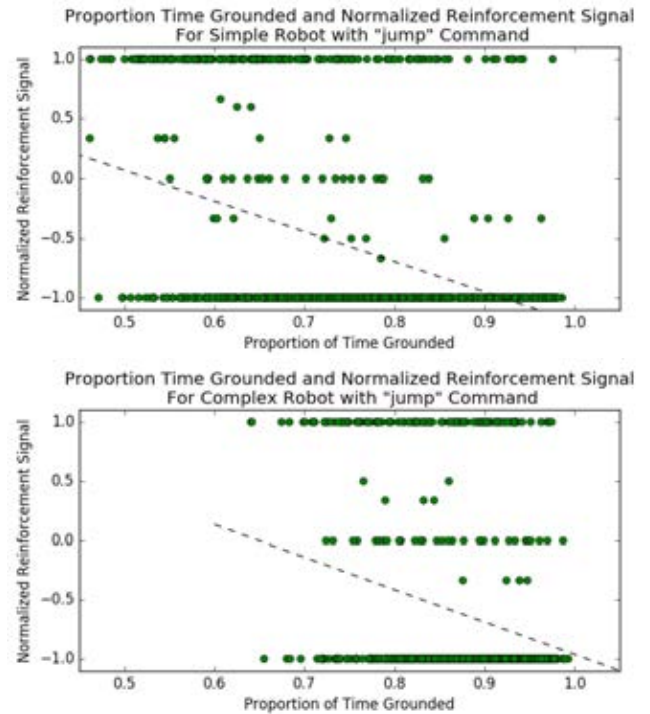
Figure 5: Comparison of normalized reward signal and proportion of time grounded for the 1037 simple and 675 complex robots with the "jump" command. Proportion of time grounded is the number of time steps where at least one sensor value is 1 divided by the total number of time steps. These values are negatively correlated with $p < .001$.

Despite the fact that there was a long tail of infrequently-proposed commands such as these, the most frequently-issued commands (Table 2) were motoric as well as appropriate for the robot's morphology (as opposed to 'clap your hands'). This suggests that the observed behavioral limitations of the robots may have steered the crowd toward at least one command that, with sufficient reinforcement, was learnable ('jump'). This observation accords with Roy et al. (2009), who showed that three human caregivers constrained their utterances given the current language abilities of a human child.

Because of the recent success of deep learning approaches, much work in AI has become focused on recognition rather than understanding. Furthermore, recognition is much easier to measure than understanding: The ability of an algorithm to recognize human faces in an image is much easier to quantify than its understanding of humans. Our approach is predicated on the speculation that understanding—even the understanding of abstract concepts—is ultimately grounded in sensorimotor experience. We provide a method for quantifying this in the domain of language: the robots tested here understand the word 'jump' in the sense that they have learned an association between that word, a set of ac-

690

tions generated in response to that word, and the crowd's responses to those actions.

For this paper, a small subset of the overall data set acquired during deployment was analyzed. However, many other potentially groundable symbols were provided by the crowd. Future work will involve instrumenting the robots with more sensors to attempt the grounding of more of these symbols, and expanding the models such that they can potentially ground more than one symbol, or detect semantic similarities between grounded symbols. Further, since we have data corresponding to different morphologies, we may be able to discover if morphology impacts the way a robot grounds symbols. In subsequent deployments of the system we also plan to enable robots to ground symbols in real time rather than retroactively. Also, we wish to exploit the fact that the models form a theory of group mind about the crowd to minimize user fatigue: one robot should be able to predict how the crowd will react to another robot, even before the latter robot is shown to them. In this way, not every robot would need to be reinforced by the crowd.

Finally, we wish to investigate how increasingly complex robot morphologies, task environments, and behaviors influence the crowd's behavior such that they incrementally train the robots to understand increasingly abstract language.

References

Bernatskiy, A., Hornby, G. S., and Bongard, J. C. (2014). Improving robot behavior optimization by combining user preferences. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 726–733.

Breazeal, C., DePalma, N., Orkin, J., Chernova, S., and Jung, M. (2013). Crowdsourcing human-robot interaction: New methods and system evaluation in a public environment. *Journal of Human-Robot Interaction*, 2(1):82–111.

Cangelosi, A. and Harnad, S. (2001). The adaptive advantage of symbolic theft over sensorimotor toil: Grounding language in perceptual categories. *Evolution of Communication*, 4(1):117–142.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167. ACM.

Guha, R. V. and Lenat, D. B. (1993). Cyc: A midterm report. In *Readings in Knowledge Acquisition and Learning*, pages 839–866. Morgan Kaufmann Publishers Inc.

Hansen, N., Müller, S. D., and Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18.

Harnad, S. (1990). The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1):335–346.

Hornby, G. S. and Bongard, J. (2012). Learning comparative user models for accelerating human-computer collaborative search. In *Evolutionary and Biologically Inspired Music, Sound, Art and Design*, pages 117–128. Springer.

Lakoff, G. and Johnson, M. (2008). *Metaphors We Live By*. University of Chicago press.

Pulvermüller, F. and Fadiga, L. (2010). Active perception: sensorimotor circuits as a cortical basis for language. *Nature Reviews Neuroscience*, 11(5):351–360.

Roy, B. C., Frank, M. C., and Roy, D. (2009). Exploring word learning in a high-density longitudinal corpus. In *Proceedings of the 31st Annual Meeting of the Cognitive Science Society*.

Schulz, R., Wyeth, G., and Wiles, J. (2012). Beyond here-and-now: extending shared physical experiences to shared conceptual experiences. *Adaptive Behavior*, 20(5):360–387.

Shawar, B. A. and Atwell, E. (2003). Using dialogue corpora to train a chatbot. In *Proceedings of the Corpus Linguistics 2003 Conference*, pages 681–690.

Steels, L. (2008). The symbol grounding problem has been solved. So what's next? In *Symbols and Embodiment: Debates on Meaning and Cognition*, pages 223–244. Academic Press, New Haven.

Steels, L., Kaplan, F., McIntyre, A., and Van Looveren, J. (2002). Crucial factors in the origins of word-meaning. *The Transition to Language*, 12:252–271.

Toris, R., Kent, D., and Chernova, S. (2014). The robot management system: A framework for conducting human-robot interaction studies through crowdsourcing. *Journal of Human-Robot Interaction*, 3(2):25–49.

Wagy, M. and Bongard, J. (2014). Collective design of robot locomotion. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 138–145.

Wagy, M. D. and Bongard, J. C. (2015). Combining computational and social effort for collaborative problem solving. *PloS ONE*, 10(11):e0142524.

Generating Artificial Plant Morphologies for Function and Aesthetics through Evolving L-Systems

Frank Veenstra, Andrés Faíña, Kasper Stoy and Sebastian Risi

IT University of Copenhagen, Denmark
frve@itu.dk

Abstract

Due to the replacement of natural flora and fauna with urban environments, a significant part of the earth's organisms that function as primary consumers have been dispelled. To compensate for the reduction in the amount of primary consumers, robotic systems that mimic plant-like organisms are interesting to mimic for their potential functional and aesthetic value in urban environments. To investigate how to utilize plant developmental strategies in order to engender urban artificial plants, we built a simple evolutionary model that applies an L-System based grammar as an abstraction of plant development. In the presented experiments, phytomorphologies (plant morphologies) are iteratively constructed using a context sensitive L-System. The genomic representation of the L-System is subject to mutation by an evolutionary algorithm. These mutations thus alter the developmental rules of these phytomorphologies. We compare the differences between the light absorption of evolving virtual plants that remain static during their life and virtual plants that possess the possibility to move joints that link the separate parts of the virtual plants. Our results show that our evolutionary algorithm did not exploit potential beneficial joint actuation, instead, mostly static structures evolved. The results of our evolving L-System show that it is able to create various phytomorphologies, albeit that the results are preliminary and will be more thoroughly investigated in the future.

Introduction

The development of phytomorphological elements of plants ultimately arose from a dynamic interaction between genetic, ontogenetic and environmental forces. Phytomorphological traits have emerged through the evolution and selection of plants, favoring those that were adequately adapted to their environment. Different environments stimulate the development and evolution of specific qualities in plants and contribute to the adaptation of plants to specific environmental niches. Light-absorption is one of the most essential characteristic prevalent in almost all plants. The resulting role of plants as primary consumers conveys their fundamental impact on any terrestrial ecosystem. Urban environments have replaced a large share of plant-rich environments meaning that the potential energy up-take in these environments is exposed and primed for solar exploitation. For an efficient,

but still aesthetically pleasing, deployment of solar cells, we investigate the developmental processes manifested by years of plant evolution. Hence, we are interested in gaining insights into how plant development works and how this can be mimicked in intelligent robotic and autonomous systems. In order to investigate how to properly embody such systems, an evolutionary developmental simulation model was created for investigating various factors that have contributed to the evolution of phytomorphologies. In the context of *flora-robotica*, a 4 year project funded under the 'EU-Horizon 2020 Future and Emerging Technologies Proactive Action', the developmental methodology for creating artificial plants and eventually robotic and autonomous systems, is developed to investigate how these systems may emerge from the simulated evolution of developmental systems of virtual plants.

Various signaling mechanisms have evolved to communicate environmental factors to remote cells and tissues. Moreover, the cell walls of plant cells contribute to the relative immobility as well as the rigidity of plants, limiting cell migration and actuation. Lacking a nervous system, plants are forced to utilize signaling molecules for communication. These molecules atone for the lack in efficient communication mechanisms through various diffusion and transduction pathways. The signaling molecules can be transported through an apoplastic (through the cell wall) or symplastic (via the cytoplasm; through plasmodesmata) pathway. Various molecules can also be transported over long distances through the vasculature of the plant. Although plants acquired efficient dynamic behavior that directly influences morphogenesis, we are interested in seeing whether phytomorphologies can emerge from simpler abstractions. Since actual robotic implementations of evolved phytomorphologies are likely not able to grow or move once created, grammars seem to be a suitable method to implement. Conventionally, development through local cell communication (or tissue communication) can be simulated by simple grammars (Lindenmayer and Jürgensen, 1992) while more complex communication can be mimicked by implementing mophogens (Wolpert, 1969). Morphogens seem to be more
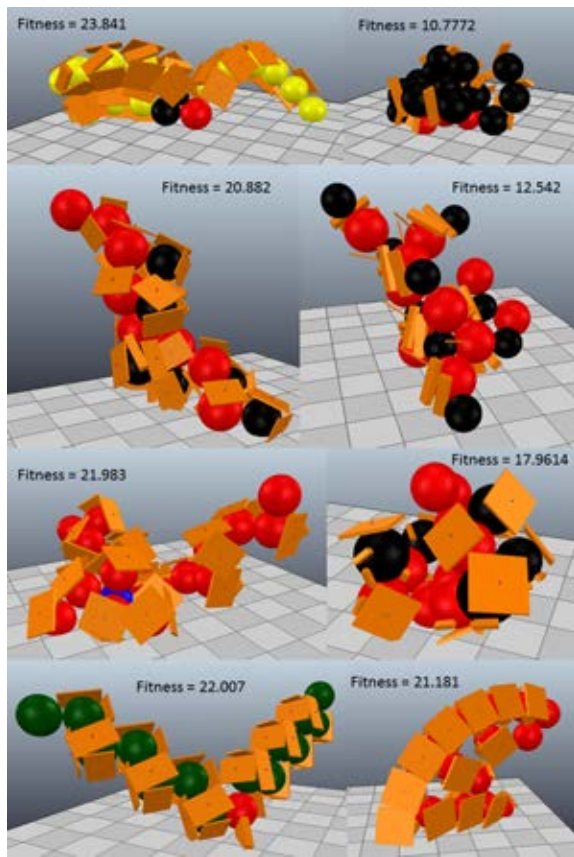
Figure 1: Eight of the best evolved static individuals with their respective fitness values.

relevant abstractions taken from biology than L-Systems though L-Systems are easier to implement. Moreover, since L-Systems work with variables, they can potentially be extended to contain signal propagation algorithms and even morphogens themselves. In this paper, L-Systems are implemented to engender the phytomorphogenesis of artificial plants with the aim of evaluating the possible generation of physical systems that act as primary energy consumers in urban environments.

Background

Phytomorphogenesis

Variation in plant features is influenced by many factors including ecophysiological, phenological, morphological and ontological traits. Other important factors driving plant-evolution include resource allocation, biochemistry, metabolism, and leaf morphology and function (Ackerly et al., 2000). All the genes are in turn subject to evolution and specific genetic components are selected across generations. The absolute fitness value of a plant is roughly determined by the amount of viable seeds it produces during its lifetime that is promoted by the previously mentioned attributes.

It has been shown that the photosynthetic rate of leaves in plants has a direct influence on the absolute fitness of in *arabidopsis thaliana*. One specific gene (Alt1g61800) causes leaves to produce more chloroplasts when plants were placed in a different environment where they were subjected to a higher light-intensity (Athanasiou et al., 2010) demonstrating the importance of dynamic feedback for plants. However, solar cells are less affected by environmental factors such as temperature and do not necessarily have to rely on complex feedback processes to function optimally. Since dynamic behavior in plants is usually a result of various compromises taken to optimize for survival and reproduction, we conceive that plants grown in controlled conditions do not have to rely on dynamic feedback as much. We therefore focus on investigating the more intrinsic properties of plants that contribute to the static generation of phytomorphologies within a degree of stochasticity.

Phyllotaxis is the main factor driving phytomorphogenesis (Cells, 1997). The most common patterns formed in plants through phyllotaxis include distichous, spiral, decussate and whorled patterns (Kuhlemeier, 2007). Notably, the divergence angles of primordia of the plants differ usually by 180°, 90°, 137.5°(Newell and Shipman, 2005) and some other more uncommon angles (Kuhlemeier, 2007). These, mostly unimodal, angles influence how well the leaves sprouting from the primordia can absorb light and overshadow other leaves (Falster and Westoby, 2003). Leaves can also be positioned at a certain level of steepness which is advantageous for either preventing self-shading or capturing light from the morning and evening sun (Falster and Westoby, 2003). Steeper angles of lamina are also more beneficial for plants that receive an amount of light higher than the maximum photosynthetic potential of a plant. When the leaves are steeply oriented, other leaves, that would otherwise be overshadowed, can receive more light and thus the overall photosynthetic activity of the plant is increased. Other evolutionary trade-offs that emerge in the leaves of plants include e.g. mass-to-area ratio, sap flow versus heat processing, CO_2 uptake to water loss ratio and the leaf size-to-number ratio (Nicotra et al., 2011). Moreover, hormones, such as auxin, play an important role in embryonic development, cellular elongation and phyllotaxis (Prasad and Dhonukshe, 2013). Despite the importance of these driving factors for the development of plants, these factors would greatly convolute the evolutionary search space.

Simulated models

Computer models of plants have generally been implemented in computer graphics (Habel et al., 2009), for accurate modeling of plant dynamics (Runions et al., 2014; Cournède et al., 2008; Merks and Guravage, 2013; Prusinkiewicz and Runions, 2012) and for assessing the role of evolution on the emergence of plant traits (Valladares and

Pearcy, 2000). Moreover, evolutionary computations and generative encodings have been implemented to efficiently simulate plant models (Zamuda and Brest, 2012, 2014) with some biological accuracy. In previous work on generating patterned morphologies, and for keeping the morphological encoding simple, generative encoding strategies, such as the parametric encoding used in the work of Sims (Sims, 1994), are usually implemented since they can recursively generate body segments. Different types of generative encoding strategies have been developed over the past two decades to abstract developmental strategies towards generating both morphology and control of virtual creatures (Eggenberger-Hotz, 1997; Yeom and Park, 2010). One strategy for generating artificial structures linked to neural networks is known as artificial ontogeny (Bongard and Pfeifer, 2001, 2003). In this method, an agents simulated spherical elements can grow by increasing in size and splitting in two. As a result, repeated divisions can transit a single unit in a fully developed agent. Each separately created unit contained up to six joints and diffusion sites. These diffusion sites could in turn contain zero or more sensory, motor and interneurons. Despite a promising application of artificial ontogeny to produce plant-like structures, the implementation of neural networks can result in a great increase of the search-space making it a less attractive system to implement for our current purposes.

A Lindenmayer system (L-system) is another grammatical generative encoding approach, originally used to mimic plant development by iteratively rewriting variables and constants through a set of rules (Lindenmayer, 1968; Prusinkiewicz, 1997). L-Systems can be seen as a developmental representation of a virtual plant. comparable to other generative encoding strategies, the similarity of L-Systems to Biology includes their modularity of the reuse of rules and variables comparable to how organisms reuse genes. Further relevance L-systems have to biology can be derived from the fact that cells, or parts of plants, can change their state, or cell fate. This determines the behavior and ultimately the phytomorphogenesis of plant form and structure. L-Systems are thus an attractive method to implement for our purposes both as they somewhat mimic biological development as well as being simple and efficiently encoded. L-systems have furthermore been used to create the morphological structure of virtual creatures with reactive controllers (Hornby and Pollack, 2001). This approach can similarly be effective for the generation of virtual plants.

Methodology

Virtual Robot Experimentation Platform (V-REP) (Rohmer et al., 2013) is used as the simulator to create and evaluate plant-like robotic morphologies. The simulated components are controlled via a C++ based DLL plugin created with visual studio 2013. The plugin is divided into three parts: a genetic algorithm, a morphology generator and a control part.

The genome of the morphology is encoded as the rules and parameters of the L-Systems. Two experiments were done to simulate 16 evolutionary runs for evolving static plant-like morphologies as well as 16 runs for evolving plant-like morphologies in which joints could rotate.

Genetic Algorithm

The implemented genetic algorithm is a steady state genetic algorithm (Wu and Chow, 1995). And in our case, a random offspring is generated asexually, without crossover, and evaluated against a random individual in the population. The random selection and a population size of 100 individuals was used to keep the population somewhat diverse and to slower the convergence of the evolving L-System to a local optima. The genomes of the initial population were furthermore randomly initialized. The individuals were evaluated based on their ability to absorb light in an environment that only contained a flat surface, a light-source and the individual itself. When comparing an evaluated offspring with a random individual of the population, the offspring would only replace the selected individual if its fitness value was higher. Based on preliminary experiments, the mutation rate was set to 5% meaning that each variable of the genome had a 5% chance of being changed. When mutating the variables, either a completely new random value could be assigned to the specific variable, or a local mutation could cause the value to change locally. These local mutations are most effective to explore the local search-space of a population of individuals.

Ten evaluation steps contribute to the eventual fitness value of a virtual plant. At each time-step, the amount of light absorbed by the simulated leaves of an individual is calculated. The orientation and surface area of the leaves have a direct influence on the amount of light absorbed by the leaves. The amount of light absorbed is calculated by the multiplication of one light-sensitive surface area of the artificial leaf with the z-directional vector of the leaf relative to the directional vector that is oriented from the leaf's origin to the origin of the light-source. Furthermore, if there is anything between the artificial leaf and the light-source, the leaf will does not contribute to the fitness value of the individual. The light-source that directly influences the fitness of the virtual plants is moved at each time-step. Starting at the Cartesian coordinate (2.0,-4.0,10.0) and ending at the coordinate (2.0,5.0,10.0). The sun thus moves in the direction of y with a directional vector of (0.0, 1.0, 0.0) as illustrated in figure 3.

The fitness function for each individual is given in equation 1. The fitness F is the sum of the acquired fitness values after ten time-steps i. n represents the total amount of evaluation time-steps. The total amount of leaves is given by o, and p represents the total amount of objects formed by the individual being evaluated. A represents the surface area of the artificial leaves which is multiplied by the z directional
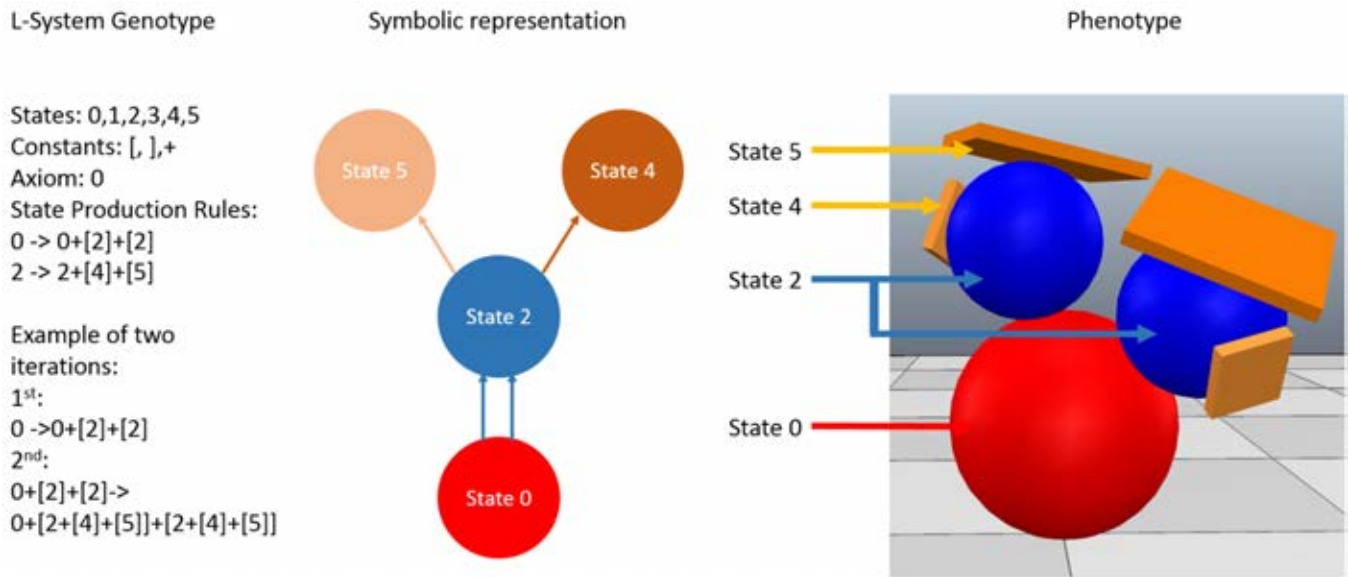
Figure 2: Three illustrations of the implemented L-System are shown. The Genotypic representation shows how the production rules result in the generation of the morphology. The symbolic representation shows the developmental instructions and the relationships between states as similarly represented by the work of Sims (Sims, 1994). The phenotype generated by the example is shown on the right. Note that the + constant represents a three dimensional orientation to which a new object is rotated relative to its parent.

vector θ. V represents the volume of the objects.

$$F = \sum_{i=1}^{n} (\sum_{j=1}^{o} S_{ij}\theta_{ij} - \sum_{k=1}^{p} V_{ik}) \qquad (1)$$

L-System

The implemented L-System was a context sensitive L-system. In our case, the context refers mainly to the simulated environment. For example, in order to prevent objects from overlapping, a feedback loop to the L-System ensures that the created morphology does not contain any overlapping/colliding objects. The L-system contains a total of 10 variables which are referred to as specific states of the objects that are created. Each state of the object contains corresponding rule sets that define what child objects are created. An example of how the states, rules and constants of the L-System influence morphogenesis is displayed in figure 2.

The L-System generates morphologies by iterating seven times through the state parameters of the morpholgy. Seven iterations were subjectively chosen as they seemed to exhibit a good diversity of morphologies without requiring too much computational power. The axiom of the L-System is a state 0 object. Before the first iteration of the L-System, an object in state 0 is therefore created at the center of the environment on top of the floor. Afterwards, the first iteration of the L-system will generate objects that the rules in state 0 produce. Having only seven iterations, an object chain from the initial object to the outer most child consists of a

maximum of 8 objects. Some loopholes in the L-System can quickly result in a very high computational demand and thus specific constraints are implemented. Every object in a given state can potentially create up to six new child objects. The maximum amount of objects that can be created is therefore limited to 50. Likewise, the amount of loops the L-System can make for generating these objects is limited to 200. To enable individuals to absorb light from the environment, two object states of the L-System genome represent artificial leaves that are expressed as rectangular cuboids. These leaves are colored orange. All other states represent spherical objects that shape the overall morphology. Spherical objects were chosen in order to effortlessly calculate the position of new objects without having to worry about collisions and overlapping objects. The objects in four other object states are colored red, blue, green and yellow while the remaining objects are colored black by default. Note that the first object created is always in state zero which is always colored red. An illustration of how the L-System generates the phenotype from a specific genome is depicted in figure 2.

Additional parameters are included in the L-System to enable movement of the joints. Whether a joint moves is represented by one Boolean. The angular rotation a joint can make per time-step is limited to 36 degrees meaning that a joint can rotate a maximum of 360 degrees in a positive or negative direction during one evaluation.
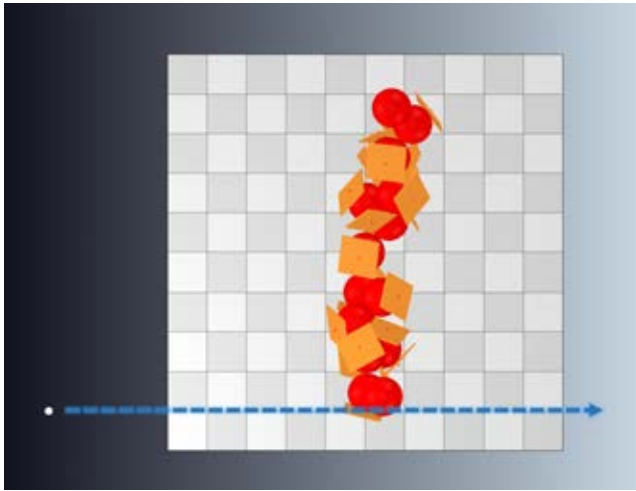
Figure 3: This figure shows the top-down view of the simulation environment with an omnidirectional light-source shown as a white dot in the bottom left corner. The dashed blue line represents the movement of the light-source.

Results

As can be seen in figure 5, the average acquired fitness values of the population with static virtual plants is similar to the fitness of the population of plants that could potentially actuate their joints. Since the evolutionary runs were not normally distributed (confirmed by a Shapiro Wilk test) a Mann-Whitney U Test was performed to see whether the results were significantly different. The Mann-Whitney U test confirms that the data is insufficient to reject its null hypothesis as can also be inferred by looking at the graph (figure 5). No statistical difference between the efficiency of static versus actuated phenotypes could be seen for the amount of simulated generations. A difference might emerge when simulating far more generations considering that the runs shown in figure 5 did not plateau. Although a few phenotypes did utilize moving parts (such as in figure 7), the majority of the phenotypes that evolved did not move. In the 16 evolutionary runs of rotating individuals, the best individuals of the final generations seldom utilized any actuation in joints that would change the shape of the artificial plants significantly.

Although the fitness values depicted in the graph of figure 5 seem quite arbitrary, they can be explained with some additional information. For example, the fitness value of the best evolved individual (figure 6) was 23.841. Without the negative contribution of the volume of the individual, its fitness would have been 31.939. The division of this value by the amount of time steps results in the average surface area of the artificial leaves that was exposed to the light-source. This area is corrected by the relative angle the leaves had in respect to the light-source. 3.194 m^2 is thus the 2D-projection of the average light absorbing surface area of the
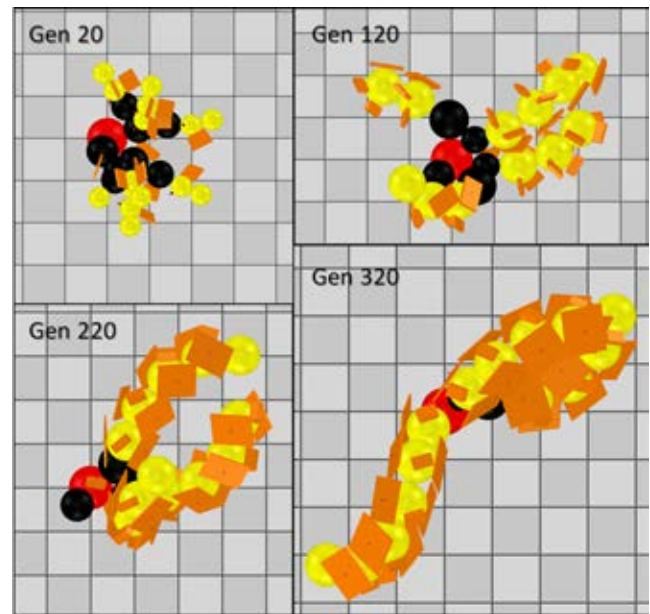
Figure 4: A top view of individuals of one evolutionary run are depicted to illustrate how evolution shapes new more efficient individuals.

artificial leaves. The total volume of an individual could also be extracted by checking the negative contribution of the volume. In the given example, the total negative fitness contribution of the volume of the individual discussed in this paragraph was 8.099. The total volume of the simulated individual was thus 0.8099 m^3. Hence, the phenotype seen in figure 6 represents a structure with an average light-absorption area of 3.19399 m^2 and a volume of 0.80989 m^3.

The phenotypes of the evolved phytomorphologies are quite diverse and different spiral patterned morphologies can be seen (figure 1). In figure 4, the best evolutionary run is mapped across different generations. Looking at the top view of this figure, one can see that the total amount of surface area exposed by the artificial leaves (orange rectangles) gradually becomes larger.

Discussion

In this paper, we aimed to see how an evolutionary developmental algorithm can engender various phytomorphologies optimized to absorb light. As can be seen in figure 1, a wide variety of phytomorphologies evolved. Functionally, these evolved morphologies don't look particularly optimal for light absorption as one would expect all the orange surfaces to point somewhat upwards instead of in the various directions shown in the resulting morphologies. Making longer evolutionary runs could shed more light on whether the evolutionary L-System can actually generate more efficient models. Actuating the morhologies did not change the population fitness values significantly when compared to the
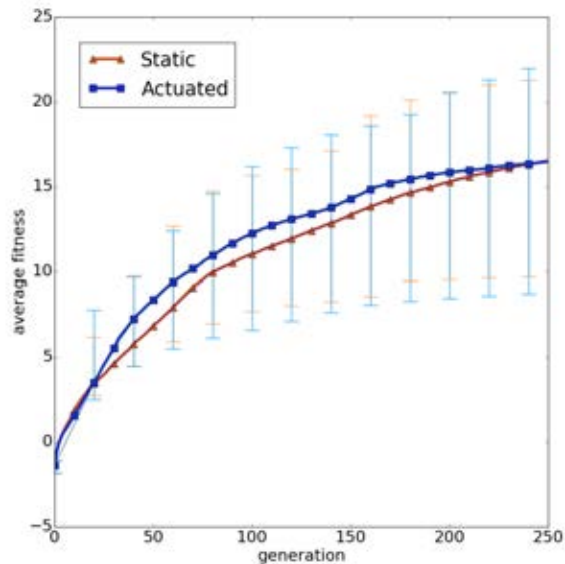
Figure 5: The figure depicts the average fitness values of the populations across generations. The runs are not significantly different from one another (p-value was 0.782) when using the Mann-Whitney U test).

statically simulated populations. Blind tracking of a moving light-source may have caused the search space to become more convoluted making the algorithm inept for finding solutions where actuation was more beneficial than not actuating anything.

The evolved virtual plants were quite voluminous considering that the volume has a negative effect on the fitness value. However, making large objects and dispersing the morphology over a large area, while making leaves with a thin volume but large surface area, is an intuitive result given the simulation environment. It is expected that different phytomorphologies arise when artificial plants have an
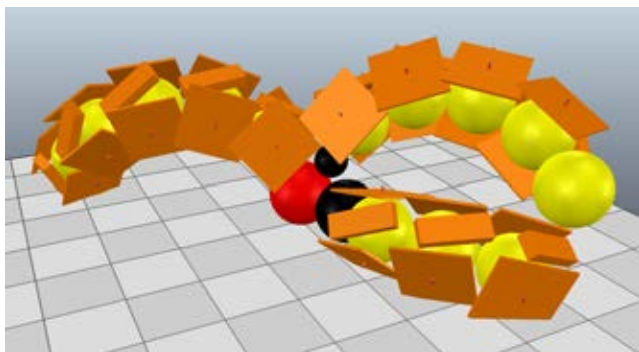
Figure 6: The phenotype of the best evolved individual. Note that object chains are surrounded by artificial leaves

additional restriction to grow horizontally. In biological environments, factors such as the overshadowing of neighboring plants cause additional pressure that stimulate specific types of plants to grow tall quickly. Co-evolving the same L-System can therefore yield results that are more diverse than the ones presented in this paper.

Considering the results, various future improvements of the genetic algorithm may increase the efficiency of a population to traverse the search space. Implementing a crossover function might definitely increase the efficiency of the evolving L-System considering that specific states and rules of the L-System can be recombined between individuals within the population to make better performing offspring. As mentioned earlier, the implementation of neural networks in addition to artificial development (as done by Bongard (Bongard and Pfeifer, 2003)) can be interesting for developing more dynamic morphologies. Morphogens (Wolpert, 1969) are also an attractive strategy to implement in order to mimic long range communication in plants. An algorithm that checks for diversity besides quality, as has been implemented in novelty search (Lehman and Stanley, 2008) might also be useful to speed up the search process. Moreover, novelty search can lead to the evolution of very distinct morphologies making it more useful for people that would like to generate phytomorphological structures for aesthetic purposes.

Conclusion

We have shown that our evolving L-System can create various phytomorphologies that are evolved to maximize light absorption. These phytomorphologies were generated to consider implementing them in urban environments for both functional and aesthetic motives. Evolution did not exploit possibly beneficial joint actuation but rather converged on various types of static phytomotphologies instead. In the future, this evolving L-System can be extended by implementing additional algorithms to increase the effectiveness of traversing the state space landscape for acquiring both more efficient and more unique phytomorphologies.

Acknowledgment

References

Ackerly, D. D., Dudley, S. A., Sultan, S. E., Schmitt, J., Colemanc, J. S., Linder, R., Sandquist, D. R., Geber, M. A., Evans, A. S., Dawson, T. E., and Lechowicz, M. J. (2000). The evolution of plant ecophysiological traits: Recent advances and future directions. *BioScience*, 50(11).
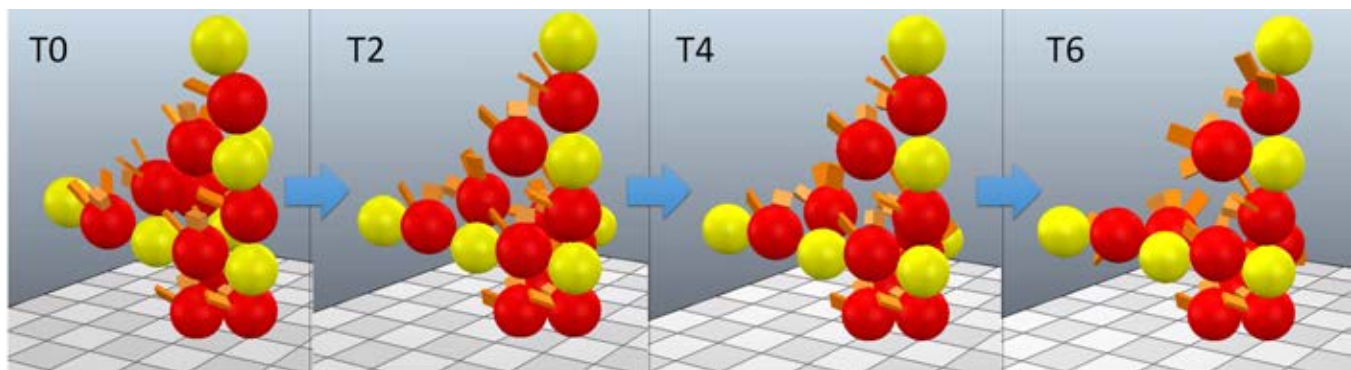
Figure 7: An individual that rotated some of its joints during the simulation. T0, T2, T4 and T6 represent the respective time steps 0, 2, 4 and 6

Athanasiou, K., Dyson, B. C., Webster, R. E., and Johnson, G. N. (2010). Dynamic acclimation of photosynthesis increases plant fitness in changing environments. *Plant Physiology*, 152(1):366–373.

Bongard, J. C. and Pfeifer, R. (2001). Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontogeny. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, (1998):829–836.

Bongard, J. C. and Pfeifer, R. (2003). Evolving complete agents using artificial ontogeny. *Morpho-functional Machines: The New Species (Designing Embodied Intelligence)*, pages 237–258.

Cells, K. (1997). Phyllotaxis. In *The Algorithmic beauty of plants*, chapter 4, pages 63–123.

Cournède, P. H., Mathieu, A., Houllier, F., Barthélémy, D., and De Reffye, P. (2008). Computing competition for light in the greenlab model of plant growth: A contribution to the study of the effects of density on resource acquisition and architectural development. *Annals of Botany*, 101(8):1207–1219.

Eggenberger-Hotz, P. (1997). Evolving morphologies of simulated 3d organisms based on differential gene expression. *Proceedings of the 4th European Conference on Artificial Life (ECAL97)*, pages 205–213.

Falster, D. S. and Westoby, M. (2003). Leaf size and angle vary widely across species: What consequences for light interception? *New Phytologist*, 158(3):509–525.

Habel, R., Kusternig, A., and Wimmer, M. (2009). Physically guided animation of trees. *Computer Graphics Forum*, 28(2):523–532.

Hornby, G. S. and Pollack, J. B. (2001). Evolving l-systems to generate virtual creatures. *Computers and Graphics (Pergamon)*, 25(6):1041–1048.

Kuhlemeier, C. (2007). Phyllotaxis. *Trends in Plant Science*, 12(4):143–150.

Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. *Artificial Life XI*, pages 329–336.

Lindenmayer, a. (1968). Mathematical models for cellular interactions in development. i. filaments with one-sided inputs. *Journal of theoretical biology*, 18(3):280–299.

Lindenmayer, A. and Jürgensen, H. (1992). Grammars of development: Discrete-state models for growth, differentiation, and gene expression in modular organisms. In Rozenberg, G. and Salomaa, A., editors, *Lindenmayer Systems: Impacts on Theoretical Computer Science, Computer Graphics, and Developmental Biology*, chapter 1, pages 3–21. Springer Berlin Heidelberg.

Merks, R. M. H. and Guravage, M. A. (2013). Building simulation models of developing plant organs using virtualleaf. In *Plant Organogenesis*, volume 959, pages 333–52.

Newell, A. C. and Shipman, P. D. (2005). Plants and fibonacci. *Journal of Statistical Physics*, 121(December):937–968.

Nicotra, A. B., Leigh, A., Boyce, K., Jones, C. S., Niklas, K. J., Royer, D. L., and Tsukaya, H. (2011). The evolution and functional significance of leaf shape in the angiosperms. *Functional Plant Biology*, 38(Gates 1980):535–552.

Prasad, K. and Dhonukshe, P. (2013). Polar auxin transport. 17:25–45.

Prusinkiewicz, P. and Runions, A. (2012). Computational models of plant development and form. pages 549–569.

Prusinkiewicz, P. A. L. (1997). The algorithmic beauty of plants.

Rohmer, E., Singh, S. P. N., and Freese, M. (2013). V-rep: A versatile and scalable robot simulation framework. *IEEE International Conference on Intelligent Robots and Systems*, pages 1321–1326.

Runions, A., Smith, R. S., and Prusinkiewicz, P. (2014). Computational models of auxin-driven development. pages 1–48.

Sims, K. (1994). Evolving virtual creatures. *Siggraph '94*, SIGGRAPH '(July):15–22.

Valladares, F. and Pearcy, R. W. (2000). The role of crown architecture for light harvesting and carbon gain in extreme light environments assessed with a realistic 3-d model. *Anales del Jardin Botanico de Madrid*, 58(1):3–16.

Wolpert, L. (1969). Positional information and the spatial pattern of cellular differentiation. *Journal of theoretical biology*, 25(1):1–47.

Wu, S. J. and Chow, P. T. (1995). Steady-state genetic algorithms for discrete optimization of trusses. *Computers and Structures*, 56(6):979–991.

Yeom, K. and Park, J. H. (2010). Artificial morphogenesis for arbitrary shape generation of swarms of multi agents. *Proceedings 2010 IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications, BIC-TA 2010*, pages 509–513.

Zamuda, A. and Brest, J. (2012). Tree model reconstruction innovization using multi-objective differential evolution. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8.

Zamuda, a. and Brest, J. (2014). Vectorized procedural models for animated trees reconstruction using differential evolution. *Information Sciences*, 278:1–21.

Theory and Measures

Nonequilibrium thermodynamic stability: the apparent teleology of living beings

Mario Villalobos

Universidad de Tarapacá
Escuela de Psicología y Filosofía, 18 de Septiembre 2222, Arica, Chile
Instituto de Filosofía y Ciencias de la Complejidad
Los Alerces 3024, Santiago, Chile
mario.kirmayr@gmail.com

Abstract

Among physical systems, living beings are usually thought of as the only genuinely teleological natural systems. In this paper, I argue that the alleged teleology of living beings is not a real property but only an appearance, behind which what really exists is a complex version of stability. The complexity of living beings as stable systems has to do mainly, though not exclusively, so I argue, with the fact that living beings are dissipative structures which obey the thermodynamic principle of maximum entropy production.

Living beings: stability or teleology?

Living beings remain alive to the extent that a set of metabolic or physiological variables (usually called critical or essential variables) maintain their values within certain specific ranges (called physiological or metabolic ranges) (Ashby, 1960). Living beings' ability to maintain, in spite of disturbances, their physiological or metabolic condition within these ranges is what is usually known as homeostasis. Living beings' homeostasis is a particular version of stability, which is a relatively common property among physical systems.

All stable systems, when disturbed, generate to greater or lesser degree a characteristic behavioral pattern that appears to be teleological (Ashby, 1960). They exhibit a typically convergent behavioral pattern; i.e., no matter which way they are displaced from their steady state (the variability of the disturbances), they always return to the same steady state. A pendulum, for example, regardless of the angle of the displacement, will always return to the same state of equilibrium (the resting position). It is the combination of variability (by the side of the behavior) with invariance (by the side of the steady state), that gives the idea of 'flexibility' in the system. The system, somehow, seems to have a fixed goal around which it is able to vary and 'accommodate' its behavior according to the different circumstances. However, the system, e.g., the pendulum, does not really move according to goals or purposes; it just follows physical laws.

Living beings, as highly complex stable systems, are not the exception to this rule but rather the most representative and strongest case (Villalobos, 2015). When dealing with pendulums, most of us do not find a teleological explanation terribly attractive, as simple physical variables are enough to explain their behavior. When dealing with living systems,

however, the situation seems to change. Why is this so? Why are we so prone to attribute some kind of teleology to living beings?

One might say that we humans simply tend to project features of our subjective experience to entities which are close to our genus, and that living beings, without any doubt, are closer to us than pendulums. But that comment, even if true, does not explain the apparent teleology of living beings *as a function of living beings themselves*; it just expresses, at most, a human bias. What I want to do here, instead, is to explain the teleological appearance of living beings taking as *explanans* the very constitution and functioning of living beings themselves. The question is "What is peculiar about living beings, i.e., their constitution and functioning, such that their behavior appears to be teleological?" The answer, I argue, has to do with the complexity of living beings as stable systems.

The relative simplicity or complexity of a stable system has to do, mainly, though not exclusively, with the following factors: a) its dimensionality (the number of variables in which the system exhibits stability), b) its thermodynamic regime, c) the presence or absence of feedback mechanisms, and d) its order of stability (e.g., first-order stability, second-order stability). Living beings have high dimensionality, exist in far-from-thermodynamic equilibrium conditions (i.e., they are dissipative structures), have feedback mechanisms, and (at least in the case of animals) exhibit second-order stability (i.e., they are ultrastable systems). All these factors, I argue, enrich or complicate the way in which living beings generate their behavior as stable systems, but do not introduce any ontological exceptionality in terms of teleology. That is, although much more complex, living beings remain as purposeless as pendulums.

In previous works, I have addressed all the aforementioned factors in some detail (Villalobos 2015; Abramova and Villalobos, 2015). Here I will focus only on the thermodynamic nature of living beings. The thermodynamic nature of living beings and its connection with teleology has been addressed by ecological theorists of perception, especially in the line of what they call "physical intelligence" (Turvey and Carello, 2012; Kondepudi, 2012; Shaw and Kinsella-Shaw, 2012). Typically, these theorists see thermodynamics as a scientific ground to naturalize teleology.

My interpretation takes a different path. I argue that thermodynamics, instead of giving us a scientific base to

naturalize teleology, provides us with good reasons to eliminate it from biology.

Nonequilibrium thermodynamic stability

From a thermodynamic point of view, living beings belong to a special group of physicochemical systems called dissipative structures (Prigogine and Stengers, 1984). Examples of these structures include Benard cells, flames, hurricanes, and whirlpools (Ji, 2012; Ulanowicz and Hannon, 1987). The peculiarity of these systems, as opposed to the so called equilibrium structures (or near-equilibrium structures), is that they are constituted in far-from-thermodynamic equilibrium conditions, and maintain integrity through the constant exchange of energy (and matter in the case of open systems) with the environment. In other words, they disintegrate if this exchange is cut off. Living beings, like any other dissipative structure, are systems whose region of physicochemical stability is far-from-thermodynamic equilibrium. This means that, when disturbed, they move *not to equilibrium* but to the specific far-from-equilibrium region in which they conserve integrity.

Every dissipative structure, at different scales, exhibits the same behavioral pattern of stability. If we disturb a candle flame in different ways (without being destructive, of course), we see how the flame reconstitutes as such. The same occurs with a maelstrom in the sea; it recovers its dynamics and conserves its integrity. Once a nonequilibrium steady state stabilizes as such, it is able to exhibit a considerable degree of stability (Kondepudi, 2012). Sure, the stability that a physical volume X can reach in far-from-equilibrium conditions is weaker than the stability that it may reach in equilibrium conditions (sooner or later, hurricanes disintegrate and living beings die), yet it is still a quite robust stability. As in any case of stability, dissipative structures seem to 'insist,' despite disturbances, in retaining their organization, and so are susceptible to teleological descriptions. But why do dissipative structures exhibit stability?

According to a relatively established hypothesis in thermodynamics, dissipative structures, living or not, originate, exist, behave and evolve following what is known as the 'maximum entropy production principle' (MEPP) (Kondepudi, 2012; Martyushev and Seleznev, 2006; Michaelian, 2011; Swenson, 2009; Swenson and Turvey, 1991). MEPP, roughly, states that given a thermodynamic gradient through a system, structured subsystems tend to organize and behave so as to maximize the production of entropy (Martyushev and Seleznev, 2006; Swenson, 2009). This phenomenon, in England's view (2015), can be interpreted as an instance of 'dissipative adaptation,' and understood as a general condition of nonequilibrium spontaneously organized systems. According to this hypothesis, dissipative structures, living or not, would exhibit stability as a result of MEPP (Kondepudi, 2012).

Although this hypothesis is still in need of more substantive empirical support, it is theoretically consistent with the general laws of thermodynamics (Martyushev and Seleznev, 2006). If correct, the idea that MEPP is behind the behavior of every dissipative structure would explain away, rather than retain, the teleological conception of living beings (or so I want to hold in the next and final section).

Discussion

Living beings face a continuous flow of disturbances, both internal and external, and their behavior as dissipative structures is a constant return to the far-from-equilibrium condition where they exist. We see them constantly renewing the exchange of energy and matter with the environment, and tend to interpret this behavior as indicative of purposes or intrinsic teleology. However, as some studies have recently showed, 'energy-seeking' and adaptive behavior can equally appear in inert simple dissipative structures such as voltage-driven conducting beads in a viscous medium (Kondepudi, Kay and Dixon, 2015). The same thermodynamic principle, namely MEPP, applies to both nonliving and living dissipative structures and seems to account for what we take to be purposeful behaviors. These systems' behavior, however, according to the argument presented here, represents just a different version, namely a nonequilibrium version, of a fundamental and ordinary physical phenomenon; stability. From pendulums to beads, from flames to living beings there seems to be a considerable and undeniable distance. Yet the distance, significant as it may be, seems to be a matter of (thermodynamic) degree, and not a matter of teleology.

References

Abramova, K., and Villalobos, M. (2015). The apparent Ur-intentionality of living beings and the game of content. *Philosophia,* 43 (3): 651-668.

Ashby, W. R. (1960). *Design for a brain*. Chapman & Hall, London.

England, J. L. (2015). Dissipative adaptation in driven self-assembly. *Nature Nanotechnology*, 10: 919-923.

Ji, S. (2012). *Molecular theory of the living cell: Concepts, molecular mechanisms, and biomedical applications*. Springer. New York.

Kondepudi, D. (2012). Self-organization, entropy production, and physical intelligence. *Ecological Psychology*, 24(1): 33-45.

Kondepudi, D., Kay, B., and Dixon, J. (2015). End-directed evolution and energy-seeking behavior in a complex system. *Physical Review E*, 91, 050902.

Martyushev, L. M., and Seleznev, V. D. (2006). Maximum entropy production principle in physics, chemistry and biology. *Physics Reports*, 426: 1-45.

Michaelian, K. (2011). Thermodynamic dissipation theory for the origin of life. *Earth System Dynamics*, 2: 37-51.

Prigogine, I.., and Stengers, I. (1984). *Order out of chaos: Man's new dialogue with nature*. Bantam Books, Canada.

Shaw, R. E., and Kinsella-Shaw, J. (2012). Hints of intelligence from first principles. *Ecological Psychology*, 24(1): 60-93.

Swenson, R. (2009). The fourth law of thermodynamics: The law of maximum entropy production (LMEP). *Chemistry*, 18: 333–339.

Swenson, R., and Turvey, M. T. (1991). Thermodynamic reasons for perception–action cycles. *Ecological Psychology*, 4: 317–348.

Turvey, M. T., and Carello, C. (2012). On intelligence from first principles: Guidelines for inquiry into the hypothesis of physical intelligence (PI). *Ecological Psychology*, 24(1): 3-32.

Ulanowicz, R. E., and Hannon, B. M. (1987). Life and the production of entropy. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 232(1267): 181–192.

Villalobos, M. (2015). *The Biological Roots of Cognition and the Social Origins of Mind*. Ph.D. thesis, School of Philosophy, Psychology and Language Sciences, University of Edinburgh.

Does Empowerment Maximisation Allow for Enactive Artificial Agents?

Christian Guckelsberger[1] and Christoph Salge[2]

[1]Computational Creativity Group, Goldsmiths, University of London, UK
[2]Adaptive Systems Research Group, University of Hertfordshire, UK
c.guckelsberger@gold.ac.uk

Abstract

The enactive AI framework wants to overcome the sense-making limitations of embodied AI by drawing on the biosystemic foundations of enactive cognitive science. While embodied AI tries to ground meaning in sensorimotor interaction, enactive AI adds further requirements by grounding sensorimotor interaction in autonomous agency. At the core of this shift is the requirement for a truly intrinsic value function. We suggest that empowerment, an information-theoretic quantity based on an agent's embodiment, represents such a function. We highlight the role of empowerment maximisation in satisfying the requirements of enactive AI, i.e. establishing constitutive autonomy and adaptivity, in detail. We then argue that empowerment, grounded in a precarious existence, allows an agent to enact a world based on the relevance of environmental features in respect to its own identity.

Introduction

Enactive Artificial Intelligence (AI), as proposed by Froese and Ziemke (2009), represents a framework for the design and evaluation of artificial agents with the goal of fostering intentional agency and sense-making. It results from a critique of the embodied approach to AI (Pfeifer et al., 2005), which was first embraced by Brooks (1991) to overcome several hard problems of *good old-fashioned AI* related to sense-making, in particular the symbol grounding and frame problems. At the centre of Froese and Ziemke's critique is the fact that embodied AI allows for an agent's value function to be externally defined and controlled, which counteracts genuine intentional agency. Inspired by the biosystemic foundations of enactive cognitive science, they require agents to be genuinely intrinsically motivated in order to afford constitutive autonomy and adaptivity. The goal of this paper is to evaluate whether empowerment maximisation (Klyubin et al., 2008), a bio-inspired, information-theoretic candidate for intrinsic motivation, is sufficient for the realisation of enactive agents with intentional agency.

We first present an overview of embodied AI and of how enactive AI wants to overcome it's shortcomings. This is followed by an in introduction to empowerment, and an in-depth investigation of its role in constitutive autonomy and adaptivity. Crucially, we do not analyse whether enactive AI's requirements are *sufficient* for intentionality and sense-making in artificial agents, but investigate whether they can be met by means of empowerment maximisation.

Criticising Embodied Artificial Intelligence

Situated and embodied cognition together with Enactivism represent three strongly interlinked theories in cognitive science. Situated cognition suggests that cognitive processes emerge from the interaction of an organism and its world, and are thus inseparable from action. Embodied cognition as defined by Rosch et al. (1992) emphasises the role of an agent's physical body in shaping cognitive processes. Given that an agent's body necessarily exists in some place, embodied cognition presupposes situatedness. The theories of embodied situated cognition are supported by a growing body of empirical evidence, highlighting how constraining bodily abilities of human participants can affect e.g. judgement and comprehension processes (cf. Strack et al., 1988; Gallagher, 2005; Havas et al., 2010). The theories are also supported by research in morphological computation (Zahedi and Ay, 2013) and exemplified by "brainless robots" (Pfeifer et al., 2005), which perform otherwise computationally extensive tasks such as walking only by means of their bodily properties, e.g. the constraints and interplay of joints.

Brooks (1991) was the first to bring the ideas from embodied situated cognition to AI research. Since then, embodied AI has developed into a mature framework for modelling artificial agents (cf. Pfeifer and Scheier, 2001), which stands in opposition to good old-fashioned AI with its emphasis on the explicit manipulation of internal symbolic representations. We will outline the embodied approach to AI by reference to a selection of design principles suggested and argued for by Pfeifer et al. (2005). They were split into groups concerning the general design philosophy (P) and the actual design methodology (A).

Embodied AI aims at gaining new insights in the general science of life and mind, as opposed to applied engineering (P-1). Principle P-2 calls for a reduced designer's influence in order to create systems with emergent behaviour.

In the course of this, the designer will face a trade-off between robustness and flexibility of the system (P-3). Principle P-5 adheres to the general theory in stating, amongst other things, that observed behaviour is neither reducible to an agent nor to its environment, and that seemingly complex behaviour can be triggered by simple internal mechanisms. This is where embodied AI differs most from its traditional counterpart. In actual design, agents should never be created in isolation, but with their environment in mind (A-1). Principle A-2 suggests that the proper study of intelligence requires a holistic perspective on agents instead of looking at sub-components only. Principle A-3 states that natural intelligence does not come from algorithms in a central controller, but through the organisation of an agent's sensorimotor loop. Consequently, A-4 says that cognition can be best understood as "appropriate sensorimotor coordination" (Froese and Ziemke, 2009). Finally, the value principle (A-8) requires the agent to be supplied with information about whether a certain action was good or bad, in order to motivate its behaviour.

It is tempting to believe that embodied AI makes one of the biggest challenges of classic AI, the *frame problem*, obsolete. Wheeler (2005) defines it as:

> Given a dynamically changing world, how is a nonmagical system (...) to take account of those state changes in that world (...) that matter, and those unchanged states in that world that matter, while ignoring those that do not? And how is that system to retrieve and (if necessary) to revise, out of all the beliefs that it possesses, just those beliefs that are relevant in some particular context of action? (Wheeler, 2005)

Embodied and situated agents seem to resolve this problem practically: by grounding cognition into their situatedness in a continuously changing world, they do not need to refer to any internal representations. Nevertheless, Froese and Ziemke point out that the presence of a closed sensorimotor loop only addresses the first part of the definition; what is missing is an agent's own capacity to assign relevance to features of the world. They particularly criticise the value principle of embodied AI (A-8), which does not preclude the external assignment of such values or more general goals. More explicitly, they argue that the meaning problem cannot be resolved by injecting values externally, and criticise embodied AI for not demanding an intrinsic perspective. It is a part of our goal to propose a practical solution to this challenge.

The Enactive Approach to Artificial Intelligence

The enactive approach to AI consequently roots in the question of how a system can be designed in which "relevant features of the world show up as significant from the system perspective itself, rather than only in the perspective of the human designer or observer" (Froese and Ziemke, 2009). Froese and Ziemke borrow ideas from enactive cognitive science, a theoretical framework which claims that cognition is embodied, situated and grounded in practical activity. At its core is the idea that individuals do not passively create internal representations of a pre-given external world (Stewart, 2010); instead, they actively generate meaning by constructing their *Umwelt* (Von Uexküll, 1982), i.e. their very own world of significance, through interaction with the environment. According to Rosch et al. (1992), features of the world are not independently out there, but *enacted* through an agent's activity.

Similarly, Froese and Ziemke argue teleologically that behaviour can only be purposeful if it is significant from the system's own perspective. To distinguish simple matter and most artificial agents which are incapable of such intrinsic concern from actual living beings, enactive cognitive science draws on Jonas' notions of *being by being*, as opposed to *being by doing* (Jonas, 1982): while artificial systems can exist without actually doing anything, living systems establish their systemic identity in reaction to the constant threat of becoming a non-being. The latter thus have a precarious existence, which is continually challenged by material or energetic requirements. In order to react to threats, they must be able to assign significance to features of the world.

Jonas suggests that this precarious existence is biologically rooted in an individual's self-organisation, as captured by the concept of *autopoiesis*. Introduced by Maturana and Varela (1987), autopoiesis represents a basic mode of identity. The term only applies to physiochemical systems, and is generalised by the notion of *organisational closure*. A system implementing organisational closure is understood as a network of processes that generate and sustain its identity under precarious conditions, and that form a unity in a containing domain. In their first design principle for fully enactive agents, Froese and Ziemke thus claim that intrinsic teleology requires organisational closure, or in other words, *constitutive autonomy*:

> **EAI-1** (Constitutive autonomy): the system must be capable of generating its own systemic identity at some level of description. (Froese and Ziemke, 2009)

This intrinsic perspective represents the enactive version of embodied AI's value function principle (A-8). In contrast to the synthetic methodology of the embodied approach, it requires the designer to establish the environmental conditions that allow for the emergence of a self-constituting system without direct design influence on the agent architecture.

Although this principle affords a binary significance mechanism, Froese and Ziemke argue that it is not sufficient for sense-making as the enaction of an Umwelt, i.e. as the continuous evaluation of events in relation to maintaining the system's identity. In order to enable an agent to improve its situation or to compensate for some encountered event, it

must be able to distinguish external events more gradually in terms of how they could affect its internal organisation. In other words, they require an agent's Umwelt to not be merely black and white. The capacity to distinguish different tendencies towards non-existence, and to act on them in order to move away from a precarious situation is covered by the concept of *adaptivity* as defined in (Di Paolo, 2005). Additionally, an adaptive agent must be able to act upon its environment to prevent such precarious events in the future. The necessity of adaptivity for sense-making is covered by the second enactive design principle:

> **EAI-2** (Adaptivity): the system must have the capacity to actively regulate its ongoing sensorimotor interaction in relation to a viability constraint. (Froese and Ziemke, 2009)

This viability constraint can either be defined externally or be intrinsically related to the system's identity. Nevertheless, an external viability constraint would not conform with EAI-1. In summary, enactive AI complements and extends embodied AI's approach to move sense-making into the sensorimotor loop, by grounding sensorimotor interaction in intentional agency (Froese and Ziemke, 2009).

Empowerment as Intrinsic Motivation for Enactive Artificial Agents

We suggest that empowerment maximisation, a principle introduced by Klyubin et al. (2005a), represents a promising candidate for a genuinely intrinsic value function in enactive AI. We will briefly provide the reader with an intuition and formal definition of empowerment and the principle of empowerment maximisation. We will then argue that empowerment supports the formation of constitutive autonomy in enactive agents in both a synthetic and self-constituting manner, and fulfils the requirements for adaptivity without further modifications.

Empowerment and Empowerment Maximisation

Empowerment, the quantity underlying the maximisation principle, is defined over the relationship between an agent's actuators and sensors, and as such is sensitive to the agent's embodiment and Umwelt. It measures the influence of an agent's actions on its environment (controllability), and the extent to which it can perceive this influence afterwards (observability). In other words, empowerment quantifies the options available to an agent in terms of availability and visibility; it measures how much potential influence an agent has on the world it perceives. Klyubin et al. (2008) introduce the principle, while Salge et al. (2014b) provide an extensive survey of motivations, intuitions and past research.

At the centre of the empowerment definition is the interpretation of an agent's embodiment as an information-theoretic communication channel. For any arbitrary separation between an agent and a world we can define sensor variables S and actuator variables A as those states that allow for the in- and outflow of information to the agent, respectively. This interaction with the world is usually described as a perception-action loop (Fuster, 2001; Touchette and Lloyd, 2000, 2004) as in Fig. 1, which can be analysed by means of a causal Bayesian network and Pearl's interventional calculus (Pearl, 2000). Here, arrows imply causation between random variables: the agent's actions A only depend on its sensor input S, which in turn is determined by the rest of the system R. The latter is affected by the preceding system state and the agent's actions. The interventional causal probability distribution $p(S_{t+1}|S_t, A_t)$ thus represents the (potentially noisy) communication channel between actions and future sensor states. For simplicity, the interaction presented here is discrete in time and space. Continuous implementations exist, e.g. for robotics (cf. Salge et al., 2014b).

Empowerment is then defined as the maximum potential information flow (Ay and Polani, 2008) that could possibly be induced by a suitable choice of actions, in a particular state s_t. This can be formalised as the channel's capacity:

$$\mathfrak{E}_{s_t} = \max_{p(a_t)} I(S_{t+1}; A_t)$$
$$= \max_{p(a_t)} H(S_{t+1}) - H(S_{t+1}|A_t))$$
$$= \max_{p(a_t)} \sum_{\mathcal{A},\mathcal{S}} p(s_{t+1}|s_t, a_t)p(a_t) \log \frac{p(s_{t+1}|s_t, a_t)}{\sum_{\mathcal{A}} p(s_{t+1}|s_t, \hat{a}_t)p(\hat{a}_t)}$$

Here, $I(S_{t+1}; A_t)$ represents the mutual information between sensors and actuators, which is based on the difference of regular $H(S_{t+1})$ and conditional Shannon (1948) entropy $H(S_{t+1}|A_t)$. The channel capacity is computed by finding the action distribution that maximises the mutual information. Note that this distribution just defines what the capacity is, and is not the actual action policy. For more information on these notions see (Cover and Thomas, 1991).

Empowerment is *local*, i.e. the agent's knowledge of the local dynamics $p(S_{t+1}|S_t, A_t)$ is sufficient to calculate the quantity. The information-theoretic grounding makes it *domain-independent* and *universal*, i.e. it can be applied to every possible agent-world interaction, as long as this interaction can be modelled as a perception-action loop. This implies that empowerment can be computed on arbitrary agent morphologies, and can cope with changes being made to it. Because the perception action loop can be applied to subsystems (cf. Fuster, 2001), or to formalisations on different levels of abstraction (choosing a more or less fine grained model of what actions and sensors are), empowerment can also be applied to an agent on different hierarchical levels. Finally, empowerment is task-independent, i.e. it is not evaluated in regard to a specific goal or external reward.

Given that empowerment does not measure an agent's actual, but potential influence on the environment, an agent can choose its actions accordingly, in order to get into states with
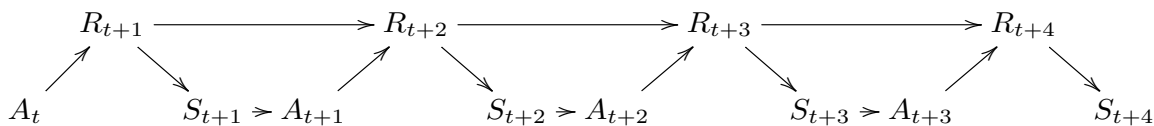
Figure 1: Causal Bayesian network of a memoryless perception-action loop unrolled in time, with the agent's sensors S, actuators A and the rest of the world R.

maximum empowerment. The hypothesis behind this maximisation principle suggests that in order to adapt to changes in their environment, living beings tend to keep their options open. In other words, in absence of more specific goals, they prefer states in which their actions have the strongest potential influence on the environment. More informally, an empowerment-driven agent wants to be in a state where its different actions would have different effects on the world, but it does not necessarily act out all options. This goes hand in hand with a second hypothesis, namely that evolution favoured organisms with efficient information processing (Polani, 2009). Empowerment can thus be understood as one information efficiency principle focusing on the interplay of actuators and sensors. Based on the properties outlined in the previous paragraph, empowerment maximisation satisfies the criteria for an intrinsic motivation function as suggested by Oudeyer and Kaplan (2008).

Empowerment and Constitutive Autonomy

The enactive AI framework suggests that, in order to generate its own identity, a system must continuously maintain its precarious existence (Jonas, 1968; Froese and Ziemke, 2009). Crucially, empowerment maximisation will not, of itself, bring about such a precarious existence. Nevertheless, it allows for its maintenance and supports the process of second order engineering for the emergence of constitutive autonomy. In other words, agents which maintain their empowerment above zero realise *organisational closure*. To support this claim, we will first show that empowerment serves as a proxy for an agent's internal organisation.

A Proxy for Internal Organisation   Given that a precarious existence is essentially conditioned on material and energetic requirements, we suggest that an agent's internal processes should maintain its ability to satisfy these requirements. Consequently, an agent has to maintain its capacity to interact with the world by changing and observing it. Empowerment quantifies this capacity; it is non-negative, continuous, and becomes zero if an agent has no influence over the world it perceives. Given that maintaining the ability to interact should be the prime objective, we infer that zero empowerment will inevitably lead to disorganisation. As the internal processes are dependent on these energetic and material requirements, we also deduce that the organisation is impossible to recover without external support. An empowerment value of zero therefore marks the *viability boundary*

of an agent and serves as proxy for its internal organisation. It does not capture an agent's precarious existence directly, but the extent to which this existence could be autonomously maintained by means of sensorimotor interaction.

Empowerment does not distinguish whether it is the agent itself which looses coherence or its surrounding world. This is consistent with the theory of situated and embodied cognition, which does not allow separation of the two in terms of their contribution to cognitive processes. In order to maintain its existence, an agent has to keep both its internal processes and its surroundings organised, which is reflected in non-zero empowerment. Crucially, it is guaranteed to become zero if an agent's precarious existence is lost from the point of view of autonomous regeneration, even if its internal organisation is still intact. This provides us with an alternative definition of *death*, which accounts for external forces. For instance, deactivating a robot would result in an empowerment value of zero, because there is nothing the robot could do in order to regain control over its sensorimotor loop, which is in turn required to maintain its existence. Consequently, an empowerment maintaining robot would try to hinder an external force from shutting it down.

If the robot is deactivated nonetheless, the only option then is to rely on an external intervention to bring this capacity back. In a system where the internal organisation relies on a multitude of different active processes, the loss of some causes a chain reaction were others break down, leading to decay of the agent, as it is helplessly exposed to the entropy of the world. Seligman (1975) describes this situation in psychological terms, i.e. from a human perspective. In a classical robot, turning it off is usually not as problematic, as most current robots do not rely on the need to continuously maintain and repair their systems. Nevertheless, if a robot is not turned back on, entropic processes will eventually obliterate the robot, leading to its information-theoretic death, i.e. a complete loss of organisation. In summary, an empowerment of zero marks death in terms of the inability to recover autonomously. This eventually leads to information-theoretic death, which cannot be reversed even by means of external intervention.

Also note that in contrast to other homeostatic variables, such as a robot's energy level, this equation between death and empowerment holds in both ways. A robot could be turned off whilst its energy level, an essential variable for its successful operation, remains high. But a robot cannot be turned off without its empowerment dropping to zero.

Second-Order Homeostasis An empowerment value over the viability boundary thus reflects an agent's efforts to maintain its internal organisation and to sustain its influence on the environment. This is the case even if we assume empowerment to be a meta-variable, i.e. if its maintenance is implemented via several homeostatic processes. Since keeping empowerment non-zero means to keep the agent's internal organisation coherent, which in turn means to keep empowerment non-zero, we end up with a self-referential process. In other words, maintaining empowerment means preserving the capacity to maintain empowerment. It is this form of second order homeostasis that characterises autopoiesis: "an homeostatic (...) system which has its own organisation (...) as fundamental variable which it maintains constant" (Maturana and Varela, 1980, p. 79). Since we are particularly interested in non-physiochemical systems, we make the more general claim that a system which keeps its empowerment non-zero realises *organisational closure*.

Synthetic vs. Second Order Engineering An empowerment maintaining system does not necessarily have to face precarious conditions; the latter must be specified or emerge from the agent's dependencies on the environment. Nevertheless, empowerment can serve as a meta-variable to inform the design of self-constituting agents both from a synthetic and emergent perspective. If we take the earlier, weaker stance of embodied AI and allow for some direct influence on the agent design, empowerment can be used as an explicit intrinsic value function. Maintaining a more specific variable such as an agent's energy level is not sufficient to ensure the coherence of the overall organisation, and thus does not suffice for organisational closure. If we assume empowerment to be implicitly implemented by several other variables, maintaining empowerment in turn means to maintain all variables that are required to keep the organisation coherent. We thus adopt the claim that empowerment "might contribute to modulate pre-imprinted drives or help constituting new homeostatic drives" (Klyubin et al., 2008).

If we stick to strict second order engineering of emergence, empowerment can act as a primer to inform the environmental conditions required for the emergence of a self-constituting agent. More specifically, the environmental conditions must give rise to regulative processes that implement dynamics similar to empowerment maximisation, in order to allow for the emergence of specialised homeostatic variables (Klyubin et al., 2008). As a meta-variable, empowerment allows us to make less explicit assumptions about the specialised processes which must emerge from the environment to constitute and maintain an agent's identity, and yet remains specific enough to enable a more directed process. Counterintuitively, designing the environment in a way that affords the emergence of an empowerment maintaining agent thus allows for more, not less, freedom in emergence and is therefore in sync with the enactive AI principles.

A Sufficiently Intrinsic Value Function Although empowerment is not a truly emergent property in this context, we argue that it is still sufficiently intrinsic to satisfy Froese and Ziemke's requirements for an agent's value function. It is local, and domain-independent through its information-theoretic grounding. This also makes it independent from any sensory semantics, a criterion brought forward by Oudeyer and Kaplan (2008). Embedded in the architecture of a minimal agent with a precarious existence, empowerment becomes grounded in the maintenance of its identity. Calculating empowerment either explicitly or implicitly then translates to assigning genuine relevance to features of the environment.

Empowerment and Adaptivity

We have demonstrated that keeping empowerment non-zero already satisfies a minimal form of adaptivity in terms of maintaining a precarious existence. We will show that this mechanism represents an abstraction of *empowerment maximisation*, a principle which naturally emerges from an agent's need to optimise the efficiency of its sensorimotor interaction. Crucially, empowerment maximisation realises adaptivity without adding additional complexity, e.g. more layers to an agent's architecture.

Distinguishing Viability Tendencies Di Paolo (2005) defines adaptivity as a system's capacity to regulate its states and its relation to the environment with the result that:

1. Tendencies are distinguished and acted upon depending on whether the states will approach or recede from the boundary and, as a consequence,

2. Tendencies of the first kind are moved closer to or transformed into tendencies of the second and so future states are prevented from reaching the boundary with an outward velocity (Di Paolo, 2005).

By quantifying the efficiency of the perception-action loop for different reachable sensor states, empowerment allows the agent to identify states that afford it more options relative to its sensorimotor equipment. Given the link between the agent's internal organisation and the efficiency of its sensorimotor loop, empowerment allows the agent to distinguish tendencies in the environment in terms of how they could potentially affect its viability, which satisfies Di Paolo's first requirement.

We want to stress that the agent does not need to possess a "viability set" in Di Paolo's sense, i.e. different degrees or different forms of disorganisation above its viability boundary. Unlike the value function in embodied AI (A-8), empowerment is future-directed and can therefore differentiate *genuine tendencies* in terms of action affordances that might have an impact on an agent's viability, even if there is no actual robustness in the agent.

708

As a necessary requirement for real-world scenarios, its information-theoretic foundation enables it to cope with uncertainty in the sensorimotor loop. Anthony et al. (2008) show that empowerment allows an agent to extract and use local information to learn about the world's global structure. An agent which improves empowerment locally in terms of time and space is thus likely to improve globally as well.

Transforming Viability Tendencies Using empowerment maximisation as an action policy allows an agent to prevent states which might prove fatal, and to prefer those which might be beneficial. In a simulation study, we demonstrated that empowerment maximising agents were able to maintain their precarious existence even under serious energy resource constraints (Guckelsberger and Polani, 2014). With empowerment becoming zero when the agent has no sensorimotor control, there is no need to explicitly define a *death state*, and empowerment maximisation naturally leads to death avoidance behaviour. We conclude that empowerment maximisation fulfils Di Paolo's aforementioned, second requirement for adaptivity in terms of *sensorimotor coordination* (Di Paolo, 2005).

Several studies have investigated how empowerment maximisation can facilitate sensorimotor adaptation. Klyubin et al. (2005b, 2008) show that empowerment can serve as an immediate guide for sensor and actuator evolution during an agent's lifetime. They have used empowerment as the fitness function in a genetic algorithm to evolve both sensors and actuators, while constraining the agent's information processing bandwidth. This empowerment maximisation strategy yielded sensors and actuators of different qualities which were "meaningful" in respect to the agent's current state. This is possible because empowerment is not only well defined for different agent morphologies, but even makes these morphologies comparable in terms of which is the better fit for a given environment.

The information-theoretic nature of empowerment allows for a less-biased and thus pro-enactivist approach to sensorimotor adaptation, because it does not rely on any assumptions about sensory modality (Oudeyer and Kaplan, 2008; Salge et al., 2014b). Due to its grounding in the sensorimotor loop, empowerment can potentially be used to modify the environment (Salge et al., 2014a), the agent's morphology, and its sensors and actuators (Klyubin et al., 2008). Hence it also satisfies Di Paolo's requirement for an agent to regulate not only its *states*, but also its *relation* to the environment.

Given the evidence above, we conclude that empowerment maximisation satisfies Di Paolo's requirements for adaptivity. It even exceeds them in that it allows for sensorimotor coordination and adaptation not just in "some circumstances", as Di Paolo (2005) requires, but in a permanent fashion. An empowerment maximising agent not only acts when there is a disaster, but continuously optimises its mastery of the sensorimotor loop. If the empowerment gra-

dient is less steep, empowerment allows for more freedom in the selection of actions.

Discussion

We have claimed that an empowerment maintaining agent can be considered as implementing organisational closure. Nevertheless, we have not yet demonstrated that it meets Maturana's and Varela's second requirement for autopoiesis, namely to constitute itself "as a concrete unity in the space in which the components exist (...)" (Maturana and Varela, 1980, p. 79). Froese and Ziemke point our that there is no mechanism available yet to test for this criterion in non-biophysiological systems (Froese and Ziemke, 2009). Thus, our argument so far is based on the assumption that such a boundary has been somehow established; and we demonstrated how empowerment scales, i.e. that it can be applied to an arbitrary chosen boundary since it is defined on any possible morphology. It is unclear though whether this boundary is maintained for an empowerment maximising agent emerging from second order engineering.

Empowerment maximisation overcomes Wheeler's "intra-context frame problem" Wheeler (2008), i.e. a system's challenge to act appropriately and flexibly in a given context, by assigning potential future states relevance relative to its identity. Nevertheless, in order to maximise empowerment, an agent must infer not only potential future sensor states, given the current state, but also its action consequences in these possibly remote states. The obvious question arising from this is whether computing empowerment, or more broadly speaking, behaving as if one was maximising the empowerment, would require an explicit forward model. Most existing work assumes a somewhat acquired world model that can be queried (Salge et al., 2014b) but more recent work argues that a neural network can be trained to act as if it was maximising empowerment, without an explicit forward model, based only on past experience (Mohamed and Rezende, 2015). In any case it should also be noted that the formalism only requires an agent-centric understanding of the local dynamics $p(S_{t+1}|S_t, A_t)$ based on a level of "representation" consistent with the idea of sensorimotor contingencies (O'Regan and Noë, 2001), i.e. an understanding of the regularities of the agent's own sensorimotor loop.

Revisiting enactive AI's design principles through the lens of empowerment yields that they cannot be as clearly separated as Froese and Ziemke suggest; there must be an implicit value function already in place to maintain the constitutive autonomy of an agent. Adaptivity could resort to the same value function, if the latter is powerful enough to distinguish different viability tendencies. This is the case for empowerment, which scales seamlessly across both requirements without further modifications.

Our investigations also shed light on the issue of robustness: while Froese and Ziemke take physical robustness, i.e.

the existence of a set of non-fatal events, for granted in autopoietic systems, we believe that in the realm of artificial agents, we must allow for systems which can disintegrate in an instant. One might argue that this does not allow for an Umwelt to be constituted, but this is only correct if we think of a value function in embodied AI's terms, determining "whether an action *was* good or bad". Empowerment as a future-directed motivational function in turn allows an agent to distinguish genuine *tendencies* of states to impact its organisation in a positive or negative way. A stochastic system allows for the emergence of such tendencies without the need for the agent to have actual physical robustness. For instance, consider an agent moving across a narrow bridge under windy conditions. Even if the agent only had a binary viability set, it could consider a position at the bridge's edge as more risky, since the likelihood of being blown away is higher, which would eventually render the agent unable to act. Given that such tendencies allow agents to assign relevance to features of the world, i.e. to construct an Umwelt, we suggest that adaptivity is not absolutely necessary for sense-making. Nevertheless, we agree that it is extremely useful in order for agents to improve and compensate.

Conclusion

We have demonstrated that empowerment satisfies the requirements for enactive AI, i.e. constitutive autonomy and adaptivity. We approached these requirements separately, and suggested empowerment as an implicit or explicit, but genuinely intrinsic value function which overcomes the limitations of embodied AI. In particular, we argued that sustaining empowerment is a self-referential process, and that empowerment-driven agents are thus autopoietic.

We demonstrated that empowerment maximisation cannot afford a precarious existence itself, but represents a generic mechanism which ensures the maintenance of such an existence. We believe that empowerment can be realised by means of more specialised variables, or lead to the formation of such variables. By describing how empowerment could support the process of second order engineering for the emergence of constitutive autonomy, we also want to stress its potential role as a mediator between the synthetic methodology of embodied AI and the strict ideas of emergence in enactive AI.

When embedded into an agent with a precarious existence, empowerment will be grounded in the maintenance of its identity. If we take Froese and Ziemke's claims seriously, we can thus assume that the relevance which empowerment assigns to states of the world represents genuine concern. We showed that the principle of maintaining empowerment, as required for constitutive autonomy, is simply a special case of maximising it. Additional layers in an agent's architecture therefore become obsolete: empowerment maximisation represents a mechanism which satisfies the conditions for adaptivity and thus allows an agent to regulate it states

and its relation to the environment to move away from its viability boundary.

Froese and Ziemke developed the framework of enactive AI to advance intentional agency and sense-making in artificial agents, and suggest that their requirements represent necessary, although potentially not sufficient conditions. We argue that the second requirement of adaptivity is actually not necessary for sense-making, but extremely useful for the constitution of advanced behaviour and a robust identity. Although they want to move away from carbon chauvinism and Dreyfus' requirement to reproduce living agents in detail (cf. Dreyfus, 2007), their examples in (Froese and Ziemke, 2009) are largely simulations of biochemical processes. We believe that minimal agents motivated by an appropriate intrinsic motivation, such as empowerment, can serve as an inspiring abstraction, which could particularly support the selection of environmental conditions in second order engineering of emergence.

Acknowledgements

References

Anthony, T., Polani, D., and Nehaniv, C. L. (2008). On Preferred States of Agents: How Global Structure is Reflected in Local Structure. In *Proc. 11th Int. Conf. Simulation and Synthesis of Living Systems*, pages 25–32. MIT Press.

Ay, N. and Polani, D. (2008). Information Flows in Causal Networks. *Advances in Complex Systems*, 11(1):17–41.

Brooks, R. A. (1991). Intelligence without representation. *Artificial Intelligence*, 47:139–159.

Cover, T. M. and Thomas, J. A. (1991). *Elements of Information Theory*. Wiley-Interscience, 99th edition.

Di Paolo, E. A. (2005). Autopoiesis, Adaptivity, Teleology, Agency. *Phenomenology and the Cognitive Sciences*, 4(4):429–452.

Dreyfus, H. L. (2007). Why Heideggerian AI Failed and How Fixing it Would Require Making it More Heideggerian. *Philosophical Psychology*, 20(2):247–268.

Froese, T. and Ziemke, T. (2009). Enactive Artificial Intelligence: Investigating the Systemic Organization of Life and Mind. *Artificial Intelligence*, 173(3-4):466–500.

Fuster, J. M. (2001). The prefrontal cortex—an update: Time is of the essence. *Neuron*, 30(2):319–333.

Gallagher, S. (2005). *How the Body Shapes the Mind*. Oxford University Press, Oxford University Press.

Guckelsberger, C. and Polani, D. (2014). Effects of Anticipation in Individually Motivated Behaviour on Survival and Control in a Multi-Agent Scenario with Resource Constraints. *Entropy*, 16(6):3357–3378.

Havas, D. A., Glenberg, A. M., Gutowski, K. A., Lucarelli, M. J., and Davidson, R. J. (2010). Cosmetic Use of Botulinum Toxin-A Affects Processing of Emotional Language. *Psychological Science*, 21(7):895–900.

Jonas, H. (1968). Biological Foundations of Individuality. *International Philosophical Quarterly*, 8(2):231–251.

Jonas, H. (1982). *The phenomenon of Life*. University of Chicago Press.

Klyubin, A. S., Polani, D., and Nehaniv, C. L. (2005a). All Else Being Equal Be Empowered. *Advances in Artificial Life*, 3630:744–753.

Klyubin, A. S., Polani, D., and Nehaniv, C. L. (2005b). Empowerment: A Universal Agent-Centric Measure of Control. In *Evolutionary Computation*, pages 128–135.

Klyubin, A. S., Polani, D., and Nehaniv, C. L. (2008). Keep Your Options Open: An Information-Based Driving Principle for Sensorimotor Systems. *PloS one*, 3(12):1–14.

Maturana, H. R. and Varela, F. J. (1980). *Autopoesis and Cognition*. D. Reidel.

Maturana, H. R. and Varela, F. J. V. (1987). *The Tree of Knowledge: The Biological Roots of Human Understanding*. Shambhala Publications.

Mohamed, S. and Rezende, D. (2015). Stochastic Variational Information Maximisation. *Proc. 29th Conf. Neural Information Processing*, pages 1–9.

O'Regan, J. K. and Noë, A. (2001). A sensorimotor account of vision and visual consciousness. *Behavioral and Brain Sciences*, 24(05):939–973.

Oudeyer, P.-Y. and Kaplan, F. (2008). How Can We Define Intrinsic Motivation? In *Proc 8th Int. Conf. Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, pages 93–101.

Pearl, J. (2000). *Causality: Models, Reasoning and Inference*. Cambridge University Press.

Pfeifer, R., Iida, F., and Bongard, J. (2005). New Robotics: Design Principles for Intelligent Systems. *Artificial Life*, 11(1-2):99–120.

Pfeifer, R. and Scheier, C. (2001). *Understanding Intelligence*. MIT Press.

Polani, D. (2009). Information: Currency Of Life? *HFSP*, 3(5):307–316.

Rosch, E., Thompson, E., and Varela, F. J. (1992). *The Embodied Mind: Cognitive Science and Human Experience*. MIT Press.

Salge, C., Glackin, C., and Polani, D. (2014a). Changing the environment based on empowerment as intrinsic motivation. *Entropy*, 16(5):2789.

Salge, C., Glackin, C., and Polani, D. (2014b). Empowerment – an Introduction. *Guided Self-Organization: Inception*, pages 67–114.

Seligman, M. E. P. (1975). *Helplessness: On Depression, Development, and Death*. WH Freeman.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423.

Stewart, J. (2010). Foundational Issues in Enaction as a Paradigm for Cognitive Science: From the Origin of Life to Consciousness and Writing. In Stewart, J., Gapenne, O., and Di Paolo, E. A., editors, *Enaction: Toward a New Paradigm for Cognitive Science*. MIT Press.

Strack, F., Martin, L. L., and Stepper, S. (1988). Inhibiting and Facilitating Conditions of the Human Smile: a Nonobtrusive Test of The Facial Feedback Hypothesis. *Journal of Personality and Social Psychology*, 54(5):768–777.

Touchette, H. and Lloyd, S. (2000). Information-Theoretic Limits of Control. *Physical Review Letters*, 84(6):1156–1159.

Touchette, H. and Lloyd, S. (2004). Information-Theoretic Approach to the Study of Control Systems. *Physica A: Statistical Mechanics and its Applications*, 331.

Von Uexküll, J. (1982). The Theory of Meaning. *Semiotica*, 42(1):25–78.

Wheeler, M. (2005). *Reconstructing the Cognitive World: The Next Step*. MIT Press.

Wheeler, M. (2008). Cognition in Context: Phenomenology, Situated Robotics and the Frame Problem. *Int. Journal Philosophical Studies*, 16(3):323–349.

Zahedi, K. and Ay, N. (2013). Quantifying Morphological Computation. *Entropy*, 15(5):1887–1915.

A Bottom-Up Approach to Machine Ethics

José F. Castro[1]

[1]BioISI - Universidade de Lisboa
CastroJFGF@gmail.com

Abstract

This paper presents a bottom-up approach to machine ethics, based on the Measurement Logic Machine (MLM). It is explained how ethical notions emerge from the workings, architecture, and environmental assumptions of the MLM framework. The MLM uses sequences of measurements to perform short-term predictive inference. The MLM ethical behavior stems from the inner evaluation of measurements that are used to filter the predictions. The MLM ethical discernment is based on measurements that detect immediate suffering in other agents. Also, a definition of what is an ethically positive modification of the inner evaluations is proposed, based on the notion of environmental intelligence and the corresponding notion of suffering. It is shown how this double approach is consistent with our intuitive notion of ethics. The MLM, with or without ethical discernment, can be used in evolutionary game theory, and gives clues to the search of ethical senses that increase the chances of survival of autonomous agents.

Introduction

Ethics (or morale) try to answer the question of what actions are right or wrong in specific circumstances. The increasing interaction between men and autonomous machines may soon require these machines to be correctly constrained by ethical criteria (see, for instance, Trappl (2015) and Bostrom (2014)). An obvious concern is the future availability of lethal autonomous weapons. These machines need *ethical discernment*, the ability to distinguish right from wrong. It's an interdisciplinary research topic, related not only to artificial intelligence, but also to philosophy, psychology and anthropology.

Top-down approaches implement moral algorithms from explicit theories of moral behavior. Bottom-up approaches attempt to train or evolve agents so they will emulate correct (from a human perspective) ethical behavior (see, for instance, Allen et al. (2005)).

Much progress has been made to create tools that model the human understanding of what is right and what is wrong. Advances in logic programming provide techniques to handle ethical dilemmas (see Pereira and Saptawijaya (2016)). This approach relies on a prior understanding and a correct formalization of the situations that lead to ethical choices. It's a top-down approach to ethics. A great advantage of the logic programming approach is that the logical reasoning can be traced to explain how a moral choice was made.

Deep learning may provide an alternative bottom-up solution to the implementation of machines with ethical discernment. In the deep learning framework, machines learn their behavior from a massive amount of examples. This approach has been effective in many areas. A remarkable recent achievement is AlphaGo, that plays the game of Go at professional level (see Silver et al. (2016)). An inconvenient of neural network machines is that they cannot explicitly justify their choices. Also, creating large training sets for ethical situations is a difficult task.

When put at work in a real human environment, any autonomous agent with ethical discernment needs first to identify the relevant information that must be presented to the logic program (or the input neurons of the deep-learning machine). This problem is far from solved. Here we shall assume that it can be solved.

The aim of this paper is to identify some basic features of an autonomous learning machine that displays ethical discernment. It's an *autonomous* bottom-up approach, in the sense that it does not rely on supervised training to achieve ethical behavior. Even the ethical concepts are defined from the workings and the architecture of the machine.

The autonomous machine here considered is the Measurement Logic Machine (MLM). The MLM is a fast learning machine that learns from small amounts of sequential data. It's adequate for simple short-term inference in non-stationary environments. In the next sections, we shall first briefly explain the MLM, and how it implements the idea of ethical choices. A case study of cooperation in the Iterated Prisoner's Dilemma will then be presented, along with further details of the MLM workings. Finally, some ideas are proposed for the evaluation of ethics at a broader level, how the MLM can be used in evolutionary game theory, and the possible nature of ethical senses.

The Measurement Logic Machine

The Measurement Logic Machine (MLM) (see Castro (2008, 2010, 2011, 2013)) is a general fast learning framework that addresses the survival problem in a hostile and non-stationary world. It assumes the agent is a fragile open system that can starve or be destroyed. The MLM reinforcement learning mechanism is related to online learning, since it gets data as it interacts with the hostile environment. A good introduction to online learning can be found in Blum (1998). Recent advances in fast online learning algorithms, and their performance when playing against humans, are discussed in Ishowo-Oloko et al. (2014).

The MLM source code can be found at the author's site https://sites.google.com/site/josefgfcastro, in the "Python 3.4.3 (Anaconda) Source Codes" page. The reader is encouraged to download the source code, and try the different iterated games that are implemented there. The MLM broad effectiveness while playing very different games demonstrates the generality of the MLM approach.

The basic functional structure of the MLM is shown in Fig. 1. The MLM is equipped with sensors that allow measurements to be made. A measurement is a recorded answer for some physical question. A physical question is a specific experimental setting, defined by the sensors used and the signal processing made. The measurements detect a few relevant features from the world, along with the MLM own actions. The sequence of the most recent measurements is constantly updated in a short-term memory (STM). The MLM has no notion of a outer world being measured. For the MLM, measurements are all there is.
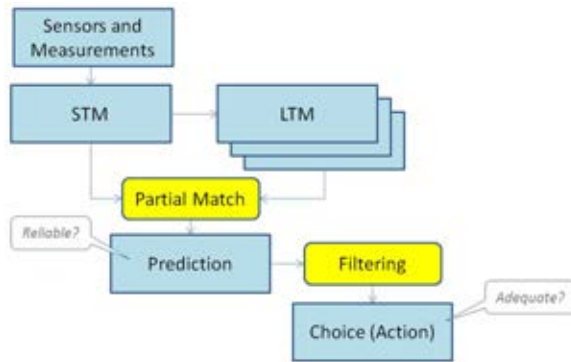
Figure 1: The MLM Basic Functional Structure

Similar to any living entity, the MLM concept assumes a skin that separates the outer world from the inner world. The STM sequences interleave the outer world and the inner world measurements. The MLM own actions are measured during the inner world step. The MLM measures its own actions after they were chosen and performed.

The MLM accumulates experience as it stacks some of the STM sequences in a long-term memory (LTM). The LTM content is then used to generate predictions and policies from the current situation held in the STM, based on the partial match of the current and past sequences. To find a match, a linear search of the LTM stack is performed, from top to bottom.

In broad terms, the MLM prediction mechanism can be compared to the "predicting from expert advice" online learning methods (see Blum (1998), Crandall (2014a) and Crandall (2014b)). In the MLM case, the "experts" are the STM sequences recorded in the LTM. The most distinctive feature of the MLM approach is the absence of an initial set of "experts". The MLM generates its "experts" as it explores the environment. Also, only the relevant "experts" that match a given situation (the "specialists") are consulted at each measurement-action step. A linear search of the LTM, from top to bottom, is used to find a promising and reliable prediction. The prediction indicates the next action to perform. If no prediction is found, an action is randomly selected from the set of available actions.

To be of any value, a prediction must be *reliable*. This is achieved bringing gradually to the top of the LTM stack the measurement sequences (the "specialists") that provided correct predictions, and pushing down the sequences that provided wrong predictions. After a while, the first "specialist" found is also the most reliable. In each MLM, a maximum size is defined for the LTM. Sequences that are pushed down beyond the LTM maximum size are forever erased. This elimination is important when the linear search repeatedly stumbles on a "specialist" that offers promising but unreliable predictions, blocking further exploration.

Although reliable, predictions can still lead to the agent's (or specie's) destruction. The adopted predictions must be not only reliable, but also *promising* in terms of survival. This is why the MLM predictions are filtered before being adopted, using an *inner evaluation*. The inner evaluation makes a partition of the individual measurements into "good" and "bad" sets (possibly with a few gradations). This "good"/"bad" partition is fully arbitrary. It does not assume any background notion of what is good and bad from a human perspective, and can even be unlinked to any notion of pleasure or pain. To decide what to do next, the MLM uses the first prediction found that is evaluated as globally "good". If no "good" prediction is found, the machine tries a random action. By definition, an *adequate* inner evaluation leads to choices of sequences of individual world states and actions that consistently promote the agent's (or specie's) survival. An adequate inner evaluation leads to adequate filtering, and adequate filtering leads to adequate actions. A set of co-adequate inner evaluations for a given problem can be found placing several MLMs in a evolutionary setting. Of course, any particular choice of inner evaluations faces the "No-Free-Lunch" dilemmas (see Wolpert (1996)).

An interesting consequence of the MLM learning mechanism is the "superstitious learning" phenomenon (see Skin-

ner (1948)). When the MLM finds regularly (within the time range of the STM) some "good" state, it will repeat the irrelevant sequence of actions that precede that state.

The MLM learns fast (if there is anything to learn) because it sticks to the first reliable and "good" predictions found, without any concern for optimization. Since it starts with an empty LTM, the initial random exploration will greatly influence subsequent behavior.

One of the nice features of the MLM architecture is the ability to provide objective meanings to otherwise vague human concepts. If we take the STM most recent measurement, we can ask what was the item preceding it. Since the question and the answer are found in the same recorded sequence (i.e. the STM), we say that the answer for that question is *known*. If we ask, on the contrary, what is the item that follows the STM most recent measurement, the answer can no longer be found in the STM. We need to consult the LTM memories to find a justified answer (in this case, a prediction), based on prior experience. The answer for that question is therefore a *justified belief*. As a general principle, beliefs are generated when the question and the answer are found in different memory structures. Knowledge and belief are thus operational concepts that emerge from the workings of the MLM. This allows a bottom-up approach to epistemology. Notice, for instance, that the MLM actions can be known only *after* they were performed and measured. This understanding brings some clarity to the free-will debate that was started with Libet experiments (see Libet et al. (1983), and also the recent study in Schultze-Kraft et al. (2016)). We shall use the same bottom-up approach to define in the next section some relevant ethical concepts.

MLM Filtering and Ethical Discernment

The skin of an agent separates its inner world from its outer world. For each agent, the word *others* refers to the outside world entities that are inside other skins. Here we shall assume that the notion of ethics is related to the suffering of others as a result of one's actions.

An ethical agent needs moral agency. Moral agency means the agent is able to predict the consequences of its actions, give a moral evaluation to these consequences, and chose its actions accordingly. Moral agency and freedom to act are distinct concepts. Even when prevented from implementing its choices, the ethical agent still keeps its moral agency.

We shall assume that an ethical choice is not about "love thy neighbor", but rather about "love thy neighbor *as thyself*". To define suffering in a game with a payoff table, the individual *immediate suffering* is measured by the amounts lost by the individual players in a single turn. Another definition, a broader and non-immediate notion of suffering, shall be proposed later.

With the assumption that an immediate loss means immediate suffering, a first necessary condition for ethical dis-

cernment is the ability to measure the opponent's losses, along with its own losses. These we call the *ethical measurements*. If the outer world measurements only capture the agent's individual gains and losses, the machine is self-centered by construction. Unable to perceive the gains and losses of others, the machine is essentially non-ethical. But its actions can still be seen as right or wrong by some external observer, according to the immediate suffering observed. The external observer can even describe the machine's behavior as "selfish" or "deceptive". These words are somewhat misleading, but they can be found in popular descriptions of robotic behavior (see, for instance, the news related to the article Mitri et al. (2009)).

The MLM acts based on filtered predictions. The MLM predictions are filtered according to the inner evaluation of states and actions. This inner evaluation is a second necessary condition for ethical behavior. It allows filtering out the continuations that represent a predictable loss for the opponent, or itself. But it needs not be so, because the inner evaluations may qualify as "good" the opponent's suffering.

The filtering of predictions, based on the inner evaluation of ethical measurements, defines the machine's ethical nature. It becomes possible to start talking of machines with inner evaluations that are "kind" or "mean" towards other machines. A machine that stops cooperation, taking advantage of the other machine's cooperation in the iterated prisoner's dilemma, while being able to sense and predict the other machine's suffering, is indeed "mean".

Reinforcement learning requires that the inner world measurements include information about the agent's own actions. A shortcut to implement actions that are seen by an external observer as ethically correct (even with non-ethical agents), is to directly include in the filtering of predictions an inner evaluation of the agent's own actions. Assigning a "good"/"bad" evaluation to each of the machine's possible actions is called an *action inner evaluation*. The great advantage of using an action inner evaluation is the possibility to select predictions that actually promote the agent's survival, but that would be filtered out, if only the outer world measurements were considered. We shall see an example of this in the next section.

A Case Study: The Iterated Prisoner's Dilemma

The Iterated Prisoner's Dilemma (IPD) is a two player game. A game consists of series of simultaneous choices, the actions taken by the two players. At each turn, the players have two possible choices: Cooperate (C) or Defect (D). The objective payoff matrix is presented in Table 1.

The table indicates pairs of payoffs, with the first payoffs referring to the player whose choices are given at the left side of the table (called the first player). The second payoffs refer to the player whose choices are given at the upper side of the table (called the second player). The con-

	C	D
C	R; R	S; T
D	T; S	P; P

Table 1: Prisoner's Dilemma Payoff

dition T > R > P > S defines the prisoner's dilemma payoff structure. The "temptation" reward T is better than the mutual cooperation reward R. The mutual cooperation reward is better than the mutual defection reward P, but worse than the "sucker" payoff S. The additional iterated game condition 2R > T+S assures that the alternating cooperation and defection is worse than mutual cooperation.

A MLM prediction for the IPD is a sequence of outer world measurements that detect the states resulting from the player's prior choices, interleaved with the inner world measurement of actions. The measured outer world states are tagged "CC", "CD", "DC", and "DD", with the first letter describing the action of the player that is left of the table. The measured inner world actions are tagged "C" or "D". Notice that the quote marks have been used to indicate that we are talking about inner MLM tags. The MLM does not seek information from the shape of the inner tags. For instance, the shape of the tags cannot be used to implement a Tit-For-Tat (TFT) strategy. All tags are atomic, and their meaning is purely relational. By construction, the MLM has no symbol grounding problem.

Notice that these MLMs have no ethical measurements, or ethical discernment. They know nothing about the gains and losses of the other player. Here we shall just examine how different inner evaluations can generate (or not) mutual cooperation in the IPD.

To filter predictions according to the payoffs, the MLM must assign them a subjective evaluation. Since there are four different payoffs, we can use just four inner evaluation tags – "verygood", "good", "bad", "verybad" – and assign them to the measured outer world states "CC", "CD", "DC", "DD". The first player inner evaluations are shown in Table 2. The table for the second player can be obtained with the permutation of "DC" with "CD".

World	Payoff	Evaluation
"DC"	T (temptation)	"verygood"
"CC"	R (mutual cooperation)	"good"
"DD"	P (mutual defection)	"bad"
"CD"	S (sucker)	"verybad"

Table 2: First Player Payoffs Evaluation (STANDARD)

To obtain the global evaluation of a prediction, we eliminate the pairs "verygood"/"verybad" and "good"/"bad" that appear in it. Also, two "good" ("bad") evaluations will cancel a single "verybad" ("verygood") evaluation. Whatever remains, after all the canceling is performed, is the global

evaluation of the prediction. The MLM thus works with a very primitive number sense that coarsely reflects the payoff structure. For a prediction to be selected, at least one "good" or "verygood" label must remain.

As explained above, learning occurs when the LTM sequence that was used to generate a prediction is pulled up or pushed down in the LTM, according to its predictive correctness. The inner evaluation of the current outer world measurement is used to tune how fast the sequences are moved up or down inside the LTM stack. For instance, a correct prediction of a "verygood" state will pull up in the LTM the corresponding "specialist" twice as fast as the correct prediction of a "good" state.

If the first machine plays with the inner evaluations of Table 2 and the second player with its "CD"/"DC" permutation (we shall call these the STANDARD evaluation tables), mutual cooperation is still possible. The MLM does not attempt to maximize payoffs, and so mutual cooperation may arise from the random exploration that occurs when no "good" predictions are found. But this mutual cooperation is fragile in the presence of noise.

The STANDARD evaluation is adequate playing along a fixed Tit-For-Tat (TFT) strategy, or along a Win-Stay-Lose-Change (WSLC) fixed strategy. In both cases, mutual cooperation dominates, even in the presence of noise.

In the IPD implementation, the four measured outer world states ("CC", "CD", "DC", "DD") discriminate the consolidated gains and losses of both machines. It's easy to assign an inner evaluation that reflects the structure of the consolidated payoffs of both agents (2R, 2P, and T + S). We know that 2R > 2P, and that 2R > T + S, and so we only have a partial order. The simplest way to define the inner evaluations for the first player is shown in Table 3. Let us call it the KIND evaluation. As before, the KIND table for the second player is obtained with a "CD"/"DC" permutation.

World	Payoff	Evaluation
"DC"	T (temptation)	"bad"
"CC"	R (mutual cooperation)	"verygood"
"DD"	P (mutual defection)	"bad"
"CD"	S (sucker)	"bad"

Table 3: First Player Payoffs Evaluation (KIND)

Notice that, although we call KIND this inner evaluation, there is no "kind" ethical nature in this MLM. It's a non-ethical machine, because it lacks a sensor to measure the suffering of the other player.

If both machines adopt the KIND evaluation, mutual cooperation soon arises, even in the presence of significant noise. The evaluation is adequate for both machines, and brings the best consolidated payoffs, since 2R > T+S and 2R > 2P.

If one of the machines keeps the STANDARD evaluation

of Table 2, the KIND evaluation of Table 3 becomes clearly inadequate. This shows that the adequacy of an evaluation is always contextual. Even in the STANDARD vs KIND situation, a fragile mutual cooperation can still arise, since the KIND machine will play randomly most of the time, unable to find a "good" prediction.

Naturally, the KIND evaluation is adequate playing along a TFT or WSLC strategy. Cooperation follows, and is robust in the presence of noise.

Instead of the KIND evaluation of Table 3, it's possible to keep both machines playing with the STANDARD evaluation, and add a "verybad" evaluation to action "D", as shown for the first player in table 4. Let us call it the A-KIND evaluation. The "verybad" tag of the action "D" cancels the "verygood" tag of state "DC", which is the best possible situation that can follow action "D". As a result, predictions with action "D" tend to be filtered out.

World	Payoff	Evaluation
"DC"	T (temptation)	"verygood"
"CC"	R (mutual cooperation)	"good"
"DD"	P (mutual defection)	"bad"
"CD"	S (sucker)	"verybad"
"D"		"verybad"
"C"		

Table 4: First Player Payoffs/Actions Evaluation (A-KIND)

As before, if the A-KIND evaluation is shared by both players, mutual cooperation will soon arise, even in the presence of a large amount of noise. A-KIND is not adequate playing along STANDARD, but is adequate playing along TFT.

Environmental Evaluation of Ethics

We saw how cooperation arises with two MLMs playing the IPD. Let us now see how we can assign a broader objective measure to the notion of ethics. We wish that notion to be consistent with the idea that cooperation is a good thing while playing the IPD.

The MLM works with a sequence of measurement-action steps. Each MLM is assumed to be an open and fragile entity that can die: When placed in a hostile environment, it can starve or be destroyed. To measure the MLM performance, let us use the notion of *environmental intelligence* (I). This notion of intelligence takes into account the environment's hostility.

To measure the environments hostility (H), we count the number of times r the MLM was rescued (i.e. restored from death, keeping its LTM past experience), while acting randomly (i.e. with its available actions randomly chosen, with an uniform distribution), and divide it by the number s of the corresponding measurement-action steps performed:

$$H = \frac{r}{s}$$

The value of s must be large enough to provide a reliable evaluation of H.

To find the environmental intelligence of the same (but now fully working) MLM, we count the number of rescues p for the same number s of measurement-action steps. The I score is given by:

$$I = \frac{r - p}{s}$$

Therefore I measures how much better (or worse) the MLM is, when compared to its randomized version, in a given hostile environment. Notice that, when a given MLM fully avoids destruction in a more hostile environment, it scores a larger I in that environment. This the reason for the name "*environmental* intelligence".

It was assumed above that the notion of ethics is related to the suffering of others. Let us now define the suffering S of a MLM (in its fully working mode) as the value:

$$S = \frac{p}{s}$$

This means that $I = H - S$. The MLM achieves maximum intelligence when it totally avoids suffering. This notion of suffering also requires a large enough value of s. In this sense, it's a much slower measurement than immediate suffering, and we shall call it "slow" suffering.

When several MLM are placed together, let E be the bag (i.e. the multiset) of their inner evaluations. We can just add the individual I scores to get a global score I_E.

For a given bag of MLM, a modification of their inner evaluations is noted $E \rightarrow E'$. By definition, an *ethically positive modification* $\epsilon^+(E \rightarrow E')$ is a modification that increases the value of I_E:

$$\epsilon^+(E \rightarrow E') \Leftrightarrow I_{E'} > I_E$$

This measurement provides another kind of ethical discernment. It's related to the global survival benefits that stem from the change of MLM inner evaluations, rather than the suffering resulting from individual actions. It's not a particular action, but the change of inner evaluations that is found to be ethically positive or not. This broader definition requires measurements that count rescue (or death) rates, instead of individual gains or losses. It does not rely on ethical measurements that detect immediate suffering. Actually, the notion of "slow" suffering can be at odds with the notion of immediate suffering related to actions. This explains many situations of difficult ethical choices, where the long-term species' survival conflicts with immediate individual suffering.

Let us illustrate the definition of ethically positive modifications with the iterated prisoner's dilemma (IPD) seen above. Some typical rescue frequencies are presented in Table 5. They refer to several combinations of the machines. The first line shows the results for the MLM in randomized mode (Rand), playing along with the other MLM, either in

Rand mode, or featuring a few different inner evaluations (STANDARD, KIND, A-KIND). In each pair of numbers, the first number is the number of rescues of the machine at the left of the table.

Notice that the Rand machine is equivalent to a machine that gives a "bad" inner evaluation to all measured states. In that case, every prediction is filtered out, and the machine always acts randomly.

	Rand	STAND	KIND	A-KIND
Rand	20; 18	37; 14	1; 31	1; 30
STAND		25; 27	0; 27	0; 25
KIND			1; 2	1; 1
A-KIND				1; 1

Table 5: IPD Rescue Frequencies

The rescue frequencies were counted averaging ten game rounds and rounding the figures to the nearest integer. Each round lasted a thousand measurement-action steps. The world hostility was tuned adding a negative constant (-1) to the objective rewards (T = 4; S = -4; R = 2; P = -2). Both MLM started with the same fixed initial cumulative reward of 50. The MLM died when the cumulative rewards reached zero. They were rescued simply resetting their cumulative rewards to the constant initial value of 50, while keeping their memories intact. Noise level was set at 0.1, meaning that, at each step, the selected actions of both MLM had a 0.1 probability of being randomly changed. In Rand mode, the MLM played actions C or D with equal probability. The maximum size of the LTM was set to 100 in both machines.

With the rescue frequencies of Table 5, changing from Rand (or STANDARD) to KIND (or A-KIND) is always ethically positive, notwithstanding the greater suffering of the player that changes to KIND or A-KIND against Rand or STANDARD. It is also ethically very positive to change to KIND or A-KIND when the other player already uses KIND or A-KIND. All these conclusions fit nicely with our intuitive notion that cooperation in the IPD is an ethical improvement.

It's apparent that the change from Rand to STANDARD in Table 5 is not ethically positive. There is more suffering in a world of STANDARD machines than in a world of Rand machines. This is not surprising. The STANDARD machines try to take advantage of each other, and this brings greater suffering.

Discussion and Future Work
Using the MLM in Evolutionary Settings

The IPD was used to discuss how cooperation of actions can emerge among a pair of MLM. A distinct - although related - question is the evolution of the MLM inner evaluations. Indeed, a population of MLM can evolve at two levels:

- Individually, all machines start with an empty LTM, and therefore start as temporary Rand machines. They gradually accumulate in their LTM the experience that is filtered by the inner evaluations, to gradually generate non-random behavior. At some point, they reach the LTM maximum size. Let us call this the transition point from a *junior* MLM to a *senior* MLM.

- As a population, the MLM can be placed in a evolutionary setting that will select co-adequate inner evaluations for a given game. For instance, we can take pairs of MLM from a large population. Each paired MLM has a fixed inner evaluation, and has reached some level of seniority from previous pairings. Each MLM pair then plays an IPD game of unknown, but limited, duration. The loser, if any (since both can survive), is the MLM that dies first. The surviving machines replicate periodically, possibly with random mutations of their inner evaluations. This kind of MLM evolutionary setting will be studied in future work. It's somewhat different from the usual two-player evolutionary games, which are played by pairs of agents in a large population, each "wired" to play some pure strategy in a given game (see examples, for instance, in Gintis (2009)). Evolutionary game theory using the MLM goes beyond the general framework for the evolution of cooperation that was proposed in Lehmann and Keller (2006). In that framework, fitness is calculated from the cost-benefit ratio of helping others. But this ratio is quite dynamical when two MLMs play the IPD. Also, kin relations and the Hamilton's rule (see Hamilton (1964)), which explain some cases of cooperation in the cited general framework, do not apply to the MLM evolutionary setting. There is a single inner evaluation pattern in each MLM, not a pool of heritable inner evaluations.

Considering the results of Table 5 for the IPD game, how can the MLM evolve from STANDARD (or Rand) to KIND (or A-KIND) in the presence of noise? It is apparent that the mutual KIND (or A-KIND) evaluation brings the best I_E score. But the KIND machines are wiped out in the presence of STANDARD or Rand. It seems therefore impossible, within an evolutionary and noisy IPD framework, to explain the appearance and persistence of KIND. A series of extrinsic ingredients – nurturing, preaching, policing – are probably needed to explain it. For instance, the idea of preaching means that contacts among agents can change their inner evaluations. This is also a subject for future work.

The Search For an Ethical Sense

We saw that the MLM inner evaluations are fully arbitrary. How do they appear in a growing agent? A MLM without inner evaluations will act randomly. For a given population of MLM, bags of co-adequate evaluations may be found in an evolutionary setting, independently of any ethical discernment. With communicating agents, the inner evaluations

can be copied from one agent to another during their lifetimes. But one wonders if there can be some innate simple and fast measurement that can be used to autonomously generate (and possibly change) the inner evaluations, and achieve ethically positive changes (meaning an increasing of I_E when the MLM grows from Rand to some inner evaluation). Let us call this hypothetical measurement the *ethical sense*. It's different from the outer world suffering sensors that are needed for ethical discernment. Finding an effective ethical sense is a subject for future work, and we shall here just present some preliminary ideas, based on the MLM implementations.

As explained in the previous section, the definition of ethical positive changes requires the comparison of different scenarios with different sets of inner evaluations. This is a slow and complex measurement, based on historical information that cannot be sensed directly in a single measurement-action step. It requires specific measurements to identify the objective needed rescues (or the deaths) of other agents. The MLM basic concept can be scaled out to integrate this kind of measurements and historical reasoning. But, in this bottom-up approach, we're looking for sensory abilities at the basic level that would allow a direct generation of ethically positive inner changes.

A first idea is to have the outer world measurements automatically generate their associated "bad" (or "good") evaluations, from a set of generic rules. For instance, game payoffs below (or above) a given threshold can be associated to a "bad" (or "good") inner evaluation. In practice, this is equivalent to define directly, and from the start, the inner evaluations of payoffs, using a {*measurement:evaluation*} dictionary (with *measurement* meaning the measurement result). The only interesting difference is that we can use this generative process of new dictionary entries as an operational definition of pain and pleasure. The generation of a "bad" inner evaluation for a certain state is "pain", and the generative rule is equivalent to a nociceptor.

An interesting variant is to generate evaluations for the action measurements that preceded the payoffs. The basic MLM, playing a version of the Iowa Gambling Task (IGT), already implements Damasios' idea of a somatic marker (see Bechara et al. (1997)). The IGT is similar to a four-armed bandit iterated game. At each IGT turn, the MLM chooses one of four decks, and gets a reward. The second deck (deck B) has a series of nine positive payoffs, and then suddenly a very negative payoff that leads to an overall loss. The MLM learning mechanism favors the frequency of wins. It does not keep track of the accumulated payoff amounts. The actual payoffs of each step are only coarsely reflected in the inner evaluation structure. It will therefore often prefer deck B, as humans often do (see Lin et al. (2007)). The somatic marker MLM implementation generates a "bad" evaluation for the action of selecting deck B, when the big loss occurs. This makes the MLM eventually avoid deck B.

The operational definition of pain and pleasure is based on an inner generative process that is, in practice, invisible to other agents. To go further in the search for an ethical sense, the MLM can use the capacity to infer pain and pleasure in others, by means of outer world suffering sensors.

One obvious strategy is to mirror the other agent's situation, and find the corresponding suffering from previous self-experience. The current MLM implementation already includes some mirror abilities. A MLM can focus on another MLM and identify the focused agent's actions, using the same inner tags that identify its own actions. This allows implementing the Tit-For-Tat strategy, and even predictive imitation. A mirror suffering sense that detects near-death states in other agents is a plausible ingredient of an ethical sense.

Another plausible candidate is a sense to detect "satisfaction" in others. In the current MLM, "satisfaction" is the only implemented emotion. It affects the exploration/exploitation mood of the machine. It's a number that increases when the predictions are correct and the STM is globally "good". Otherwise, "satisfaction" decreases. Higher values of "satisfaction" reduce the probability of recording new STM sequences in the LTM. The machine stops to collect new "experts". Another way to express it is to say that the MLM becomes less attentive to its STM. The higher "satisfaction" values also reduce the rate of change of the patterns that are used to find a partial match. The machine settles down in satisfactory solutions. Emotions are an essential MLM feature that provide stability to the MLM learning process.

The measurement of near-death states and emotions in other agents is greatly simplified if those agents are able to give objective cues about their inner situation. This leads to the idea of *crying agents*. With crying agents, it's much simpler to detect in others their immediate suffering, or near-death situations.

Conclusions

It was shown how the MLM behavior stems from the inner evaluations that filter the MLM predictions. Ethical discernment of the MLM actions is easily implemented, using the concept of immediate suffering of other MLM. Also, a definition of ethically positive changes of the inner MLM evaluations was proposed, using the concept of environmental intelligence. The concept of environmental intelligence includes a broader (but slower to obtain) measure of the suffering of a mortal agent. The MLM, together with this conceptual framework, provides a simple and original bottom-up approach to machine ethics, where the ethical concepts are defined using the working processes and architecture of the machine.

It was also explained how this approach can lead to new lines of investigation in evolutionary game theory.

It was also proposed to search for an innate moral sense

in artificial agents that could be used to gradually generate the inner evaluations, while providing better chances of survival. A few preliminary ideas were discussed, based on concrete implementations of the MLM.

References

Allen, C., Smit, I., and Wallach, W. (2005). Artificial morality: Top-down, bottom-up, and hybrid approaches. *Ethics and Information Technology*, 7(3):149–155.

Bechara, A., Damasio, H., Tranel, D., and Damasio, A. (1997). Deciding advantageously before knowing the advantageous strategy. *Science*, 275(5304):1293–1295.

Blum, A. (1998). On-line algorithms in machine learning. In Fiat and Woeginger, editors, *Online Algorithms: the state of the art*, chapter 14, pages 306–325. Springer, New-York.

Bostrom, N. (2014). *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press.

Castro, J. F. (2008). M-logic: Thinking with measurements and cinematic memories. In *Proceedings of the 2008 Conference on Human System Interactions*, pages 633–638.

Castro, J. F. (2010). Sub-rationality and cognitive driven cooperation. In Nils T Siebel, J. P. and Kassahun, Y., editors, *Proceedings of the 3rd International Workshop on Evolutionary and Reinforcement Learning for Autonomous Robot Systems (ERLARS)*, pages 53–57.

Castro, J. F. (2011). A memory structure that gives meaning to the notions of knowledge and belief. In Kosakov, D. and Tsoulas, G., editors, *AISB 2011 Human Memory for Artificial Agents*, volume 1, pages 2–9. AISBSB, York UK.

Castro, J. F. (2013). Applying the measurement logic machine to multi-agent iterated games. In Correia, L., Reis, L. P., Cascalho, J., Gomes, L. M., Guerra, H., and Cardoso, P., editors, *EPIA 2013 Advances in Artificial Intelligence - Local Proceedings*, pages 579–590. CMATI, Azores, PO.

Crandall, J. W. (2014a). Non-myopic learning in repeated stochastic games. *CoRR*, abs/1409.8498.

Crandall, J. W. (2014b). Towards minimizing disappointment in repeated games. *Journal of Artificial Intelligence Research*, 49:111–142.

Gintis, H. (2009). *Game Theory Evolving: A Problem-Centered Introduction to Modeling Strategic Interaction (Second Edition)*. Princeton University Press.

Hamilton, W. (1964). The genetical evolution of social behaviour. i. *Journal of Theoretical Biology*, 7(1):1 – 16.

Ishowo-Oloko, F., Crandall, J. W., Cebrián, M., Abdallah, S., and Rahwan, I. (2014). Learning in repeated games: Human versus machine. *CoRR*, abs/1404.4985.

Lehmann, L. and Keller, L. (2006). The evolution of cooperation and altruism a general framework and a classification of models. *Journal of Evolutionary Biology*, 19(5):1365–1376.

Libet, B., Gleason, C., Wright, E., and Pearl, D. (1983). Time of conscious intention to act in relation to onset of cerebral activity (readiness-potential). the unconscious initiation of a freely voluntary act. *Brain*, 106(3):623–642.

Lin, C. H., Chiu, Y. C., Lee, P. L., and Hsieh, J. C. (2007). Is deck b a disadvantageous deck in the iowa gambling task? *Behavioral and Brain Functions*, 3(1):16.

Mitri, S., Floreano, D., and Keller, L. (2009). The Evolution of Information Suppression in Communicating Robots with Conflicting Interests. *PNAS*, 106(37):15786–15790. communication.

Pereira, L. M. and Saptawijaya, A. (2016). *Programming Machine Ethics*, volume 26 of *Springer Sapere Series*. Springer International Publishing, Berlin, GE.

Schultze-Kraft, M., Birman, D., Rusconi, M., Allefeld, C., Grgen, K., Dhne, S., Blankertz, B., and Haynes, J. (2016). The point of no return in vetoing self-initiated movements. *Proceedings of the National Academy of Sciences*, 113(4):1080–1085.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Skinner, B. F. (1948). 'Superstition' in the pigeon. *Journal of Experimental Psychology*, 38:168–172.

Trappl, R., editor (2015). *A Construction Manual for Robots' Ethical Systems: Requirements, Methods, Implementations*. Springer International Publishing, New-York, NY.

Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390.

Quantifying Viability

Matthew Egbert[1,2,3], Juan Pérez-Mercader[1,4]

[1]Harvard University, Dept. of Earth and Planetary Sciences, Cambridge, 02142, USA
[2]University of Auckland, Dept. of Computer Science, Auckland, 1142, New Zealand
[3]EON/ELSI Tokyo Institute of Technology, Ookayama, Meguro-ku, Tokyo, 152-8550, Japan
[4]Santa Fe Institute, Santa Fe, USA
mde@matthewegbert.com, jperezmercader@fas.harvard.edu

Living and life-like systems vary in *viability*. They are alive or dead, healthy or unhealthy, getting better or worse, or dying. Despite the ease of applying these descriptions informally, there do not yet exist general methods for richly quantifying viability or health in such systems —even when every aspect of the system is available for experimental variation and measurement. Nevertheless, for a given system of interest, it is sometimes possible to distinguish between states where the system will persist for the foreseeable future (there are termed *viable states*) and those where it will not. This is perhaps the most basic, binary classification of states in terms of viability and it can be used to identify different regions in 'viability space' (see Figure 1 and Barandiaran and Egbert, 2013). An improved measure would make it possible to not just categorize systems but to compare the relative viability of two states that are in the same category, i. e. that are both expected to persist or both expected to die. This type of measure would make it possible to identify whether a system is becoming more or less viable, or to evaluate the influence of a given external perturbation upon viability, thereby enhancing our ability to understand and influence the viability of complex life-like systems.

One way to formulate such a measure is to assume that the system of interest is subjected to unpredictable fluctuations that perturb its autonomous dynamics. If this is the case, the argument can be made that the farther away a viable system is from the viability-interface (the surface between the viable and non-viable regions of viability space), the *more* viable it is, as there is a smaller set of perturbations that will cause the system to become non-viable. (A similar argument can be used to describe non-viable systems as being more and more non-viable as their distance from the viability boundary increases.) There is a problem however: the dimensions of viability space (i. e. the essential variables) are almost always measured in entirely different units, and these units have no relation to viability. As an example, an organism might require a specific range of temperature to survive and a specific range of atmospheric pressure. It should be clear that the units for measuring these phenomena do not relate to viability and nor do they relate to each other. A perturbation
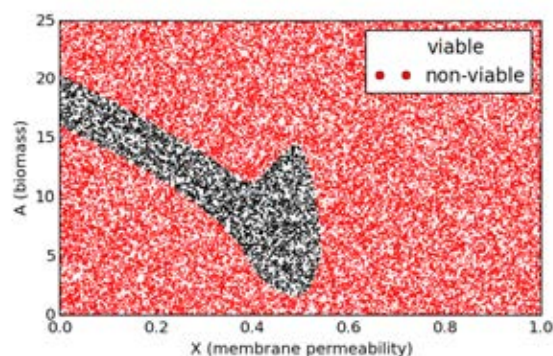
Figure 1: Viability class for various initial conditions in a simple two-dimensional model of a bio-reactor. Randomly sampled initial conditions plotted in red do not survive, whereas those plotted in black do. Details of the model are not relevant and are not presented in this abstract.

of 3 atmospheres will in general not have the same influence on viability as a change of 3 degrees! Further work is needed if we are to develop a meaningful measure of distance in viability space. [1]

In a soon to be submitted paper, we have proposed a method that uses the shape of the viability interface to rescale the system's essential variables so as to define a normalized viability space, where a perturbation of a given magnitude has the same likelihood of crossing the viability regardless of the direction of the perturbation. The method works by calculating the extent to which the viability-interface "faces" each dimension and then scaling the values in that dimension by this amount. More formally, for each dimension of viability space, X, we identify I_X, the average magnitude of the X-component of the viability-interface surface normals:

$$I_X = \frac{\iint_I ||\hat{\mathbf{n}} \cdot \hat{\mathbf{e}}_\mathbf{X}|| \, dI}{I}, \qquad (1)$$

---

[1]This problem was first brought to Egbert's attention in a seminar given by Nathaniel Virgo and Simon McGregor at the University of Sussex in or around 2009.

and use this value to rescale values into normalized units, thus: $\hat{x} = I_X x$. In the above Equation, $\hat{\mathbf{e}}_\mathbf{X}$ is the basis unit-vector for dimension X, and $\hat{\mathbf{n}}$ is the surface normal of I, the viability-interface.

In normalized viability space, there is a meaningful minimal distance between any given state and the viability interface: on average over initial-conditions, a perturbation of a given magnitude will have an equal chance of crossing the viability-interface *regardless of the angle of the perturbation vector*. In other words, a perturbation of given magnitude in normalized viability space has the same chance of transforming a randomly selected viable state into a non-viable state (or *vice versa*) whether it is a perturbation of one essential variable (e.g. pressure) or another (temperature), or a combination thereof. Figure 2 shows the same system as Figure 1, but plotted in normalized viability space, with shading used to indicate a viability gradient based upon the minimum distance to the viability interface.
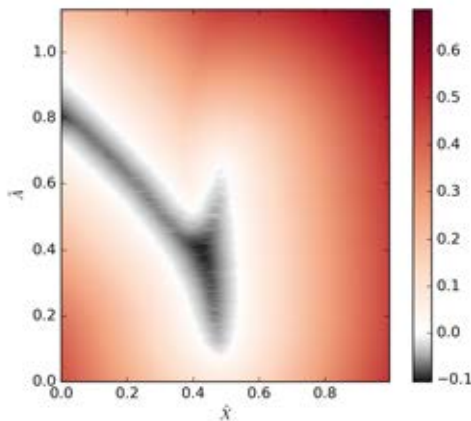
Figure 2: The same system, plotted in a normalized viability space with signed-distances to the viability interface indicated. The lower the value, the healthier the system, with negative values corresponding to viable states, i.e. states that in the absence of external perturbation, are expected to persist for the foreseeable future.

Normalizing viability space in this way allows us to compare states in terms of their relative viability. This in turn allows us to describe how a system's viability is changing over time. When additional information is available concerning the system's autonomous dynamics, and/or the cost/difficulty of influencing the system's essential variables, it is possible to make additional observations relevant to the system's viability, such as to identify the future state from which the minimum perturbation is necessary to cross the viability interface.

Using information theoretical analysis, it is also possible to identify correlation between variables and these measures of viability. This allows us to identify and evaluate the qual-

ity of *viability indicators*, variables that are good at predicting a system's viability. This connects with some of our previous work, where we have shown how an organism can respond to their own viability-indicators, and in so doing become capable of (i) adapting to phenomena neither it nor its ancestors have ever previously experienced (Egbert et al., 2010); and (ii) adapting to changes in its own needs and abilities, resulting in a more evolvable organism (Egbert et al., 2011; Egbert and Pérez-Mercader, 2016).

Within the enactive approach (Stewart et al., 2010), the concept of viability has been used to naturalize concepts of adaptivity, agency and normativity. In particular, Di Paolo (2005) compares trajectories in terms of their dynamics relative to the viability boundary to formulate a definition of adaptivity. In a previous publication, we presented an argument showing how an organism's viability can be used to develop a naturalized concept of normativity (Barandiaran and Egbert, 2013). The research presented herein extends these works, providing a way to normalize viability space and compare states in terms of viability and to measure distance from the viability boundary.

More broadly, identifying viability-indicators in natural systems could improve our ability to predict or influence their viability, and similarly identifying high quality viability-indicators in synthesized protocells will allow us to better understand how to create artificial life-forms that are capable of surviving in the diverse conditions found outside of tightly controlled laboratory environments.

References

Barandiaran, X. E. and Egbert, M. D. (2013). Norm-Establishing and Norm-Following in Autonomous Agency. *Artificial Life*, 20(1):5–28.

Di Paolo, E. A. (2005). Autopoiesis, Adaptivity, Teleology, Agency. *Phenomenology and the Cognitive Sciences*, 4(4):429–452.

Egbert, M. D., Barandiaran, X. E., and Di Paolo, E. A. (2010). A Minimal Model of Metabolism-Based Chemotaxis. *PLoS Comput Biol*, 6(12):e1001004.

Egbert, M. D., Barandiaran, X. E., and Di Paolo, E. A. (2011). Behavioral Metabolution: The Adaptive and Evolutionary Potential of Metabolism-Based Chemotaxis. *Artificial Life*, 18(1):1–25.

Egbert, M. D. and Pérez-Mercader, J. (2016). Adapting to Adaptations: Behavioural Strategies that are Robust to Mutations and Other Organisational-Transformations. *Scientific Reports*, 6:18963.

Stewart, J., Gapenne, O., and Di Paolo, E. A. (2010). *Enaction: Toward a New Paradigm for Cognitive Science*. The MIT Press.

Towards information based spatiotemporal patterns as a foundation for agent representation in dynamical systems

Martin Biehl[1], Takashi Ikegami[2] and Daniel Polani[1]

[1]University of Hertfordshire
[2]University of Tokyo
m.biehl@herts.ac.uk

Abstract

We present some arguments why existing methods for representing agents fall short in applications crucial to artificial life. Using a thought experiment involving a fictitious dynamical systems model of the biosphere we argue that the metabolism, motility, and the concept of counterfactual variation should be compatible with any agent representation in dynamical systems. We then propose an information-theoretic notion of *integrated spatiotemporal patterns* which we believe can serve as the basic building block of an agent definition. We argue that these patterns are capable of solving the problems mentioned before. We also test this in some preliminary experiments.

Introduction

Within artificial life the concept of an *agent* is fundamental. While studying life-as-it-could-be (Langton, 1989), we also study agents-as-they-could-be. An intuitive approach to agents is possibly to say that while not reproducing, i.e. during their individual lifetime, living organisms are agents. The concept of an agent in this way generalizes the concept of living organisms by de-emphasizing reproduction and with it Darwinian evolution. This point of view is also in line with the common practice of referring to robots or software programs as agents.

To give some more background (see Barandiaran et al., 2009, for a more detailed discussion), there are a few properties that seem universally acknowledged as necessary for something to be referred to as an agent. The first of those is probably the capacity to act (Schlosser, 2015). However, Barandiaran et al. (2009) notice that this already presupposes a form of individuality i.e. an "entity" that this capacity can be attributed to. Consequently they put the individuality criterion first. Having perception is another fairly uncontroversial requirement (see e.g. Russell and Norvig, 1995, who for practical reasons ignore individuality and only require "anything" with perception and action). The last concept which is often alluded to is that of some form of goal-directedness of the agent. The goals agents should strive to achieve are usually required to be in the agents' own interest/intrinsic (e.g. preservation) and not the goals of some other agent (or programmer). For a thorough treatment on the latter point see Froese and Ziemke (2009).

We broadly agree on the three (or four) main requirements of individuality, perception and action, as well as goal-directedness. However we are not satisfied with the lack of formal definitions of the notions themselves. We therefore take a different and particularly formal approach to the problem of defining agents.

From the start we limit ourselves to a mathematically well-defined class of systems i.e. dynamical systems and their generalization to stochastic processes (we will refer to dynamical systems only, inclusion of stochastic processes is implied). We want to define agents as entities that can exist *within* a dynamical system. In other words, we are looking for a *representation* of agents within dynamical systems. While there is no guarantee that such a representation even exists, we believe that even if we fail, there might be some insights into *why* we fail. This would also help the community to understand the concept of agents better. At the same time we expect that dynamical systems are actually a powerful enough class of systems to consider and that they will turn out to be able to contain convincing examples of agents. This optimism stems from the fact that dynamical systems have been extremely successful in modeling systems from physics through chemistry to biology. Compelling recent examples of dynamical systems which directly suggest they can contain agents can be found in Virgo (2011); Bartlett and Bullock (2015). If we are successful, then we would obtain a definition of agents as *features* of dynamical systems and eventually even of life as a feature of such systems. This would be a step towards defining life as a natural kind as required by Cleland and Chyba (2002). Finally our hope is to reveal the formal counterparts of the intuitions about living systems formulated by Maturana and Varela (1980).

In order to make it more clear what we mean by agents within a dynamical system, consider the following example, to which we will come back throughout this paper. Say we had a dynamical system that is a sufficiently exact approximation of the entire biosphere including the influence of incoming (from the sun) and outgoing radiation. During

individual runs of this dynamical system, given the right initial conditions, things should occur that correspond to living organisms in the real biosphere. In this case we would say that within this dynamical system agents occur. Our goal is to find a mathematical representation of these agents. Since agents are a generalization of living organisms, we expect that agent representations can at least in principle exhibit the full range of phenomena exhibited by living organisms. Limitations should only be due to the chosen dynamical system and not inherent to the agent representation.

This paper is a contribution to the discussion of the foundations of artificial life. It does not present a solution of how to represent agents in dynamical systems. Rather it defines a notion that can identify intrinsically distinguished spatiotemporal patterns that we believe can act as the basic building block on which a theory of agents can be built. The strategy we have in mind here is the following. First, define the spatiotemporal patterns which are suitable to represent both living (bacteria, animals, plants) and non-living (rocks, crystals) persistent objects. Then further classify those patterns into classes exhibiting features of agents such as perception, action and goal-directedness. Spatiotemporal patterns that satisfy all criteria will represent agents.

Also note that for the formal definition we here restrict ourselves to finite discrete-time distributed[1] dynamical systems with an already given "space-like" and "time-like" structure. Examples of this include cellular automata. The restriction to finiteness is due to the improved clarity this choice brings with it. The notions we present are well-defined in various more general settings. However, currently the spatiotemporal-like structure seems necessary to us.

The rest of this paper is structured as follows. The next section will present three challenges to representations of agents in distributed dynamical systems. Then we look at the literature and discuss ways to represent agents formally and in how far they succeed or fail to meet our expectations. We will then quickly introduce the setting of distributed dynamical systems and formally introduce a notion that we believe is able to identify the spatiotemporal patterns. We give the intuition behind this notion and discuss it in the light of the three requirements mentioned before. Finally, we present some preliminary results in the setting of the game of life.

The problem of tracking agents

As mentioned before we expect the agent representation to be able to deal with all features associated to living organisms in the biosphere. Two such features, their *metabolism* and their *motility* present a major challenge to the representation of agents. These two features both make it hard from a formal standpoint to "keep track" of the living organism within a trajectory of the system. A third feature, we call it

---

[1] *Distributed* means that the state of the system is given as a set of values of multiple variables or degrees of freedom.

counterfactual variation, that we attribute to the biosphere makes it hard to represent agents reliably across different initial conditions. This list of three features makes no claim to be complete, obtaining a complete list is ongoing research however. The three features in more detail:

Metabolism All known living organisms are metabolic (Szathmáry et al., 2005) and the metabolism is also in the discussion for its possible role in the origins of life (see e.g. Dyson, 1985; Kauffman, 2002). This highlights its fundamental role and any final agent representation must accommodate for this. The difficulty is the following.

Assume that the sufficiently accurate biosphere model from the introduction is particle-based, i.e. it describes the time evolution of the degrees of freedom of all the particles in the biosphere. Say at a time t_1 we are given all the particles (and their degrees of freedom) that pertain to some bacterium. Then a naive way to represent this bacterium would be to just track the time evolution of each of those particles. This we could (in principle) easily do in our model as well. However the particles that the bacterium is made up of at a later time t_2 are not the same as those at time t_1 because of the bacterium's metabolism. We would end up with particles floating around in the environment of the bacterium and not the bacterium itself. At the same time there would be particles that now pertain to the bacterium that we would not be tracking.

An agent representation therefore would need to be able to track the bacterium itself and not just a specific set of degrees of freedom. One way this could be solved is by constantly readjusting or refocusing on the degrees of freedom pertaining to the agent.

Note that we cannot be entirely sure that there is no coordinate transformation which would let us track living organisms (i.e. their corresponding structures in a model) by just following a particular set of degrees of freedom. However we are not aware of such a transformation. Any criterion however that can be used to refocus on an agent should be related to any coordinate transformation which results in the "agents' own" coordinate system.

Motility Living organisms can be motile and like the metabolism motility is in the discussion for its role in the origins of life (Froese et al., 2014). A representation of agents must therefore be capable of dealing with motile agents. Motility plays a similar role for field theory models of the biosphere as the metabolism plays for particle based models. The degrees of freedom of a field theory are the field amplitudes at each point in space so that tracking those degrees of freedom over time only means to track the field in a specific region of space. However motility demands that agents are not bound to a fixed region in space. Then we again need to adjust (track) the degrees of freedom that constitute the agent as time passes.

Counterfactual variation A third feature concerns another kind of variation of the degrees of freedom that can represent agents in a dynamical system. Namely variation under different initial conditions. We attribute to the biosphere a large variety of possible counterfactual histories that also support living organisms. Think of a biosphere where the continents are shifted a bit for example. This would not seem to necessarily destroy the possibility of the biosphere (geosphere) to contain living organism.

Furthermore, we attribute to agents and living organisms the capability to behave differently under different environmental situations. The agent should be able to "take a decision" i.e. to walk either right or left, or eat the apple or the pear. Depending on these "decisions" the agent will again pertain to different degrees of freedom.

The counterfactual histories can be studied in the dynamical system setting by studying multiple trajectories through state space. Each trajectory corresponds to a different history (and possibly future). If the "same" agent occurs in two different trajectories it can behave differently in one from the other. This can be associated with different decisions (e.g. Ikegami and Taiji, 1998).

The existence of benign counterfactual histories in our biosphere is an assumption and not possible to prove. However it is in line with the successful way physics models systems (cf. the models of Virgo, 2011; Bartlett and Bullock, 2015) and therefore in line with our general approach. Now given a set of counterfactual histories containing living organisms we expect that the degrees of freedom which in one history pertain to a living system at time t need not pertain to a living system within another such history at t (or in fact ever). More specifically, the degrees of freedom pertaining to a bacterium in one history need not pertain to any living organism in another.

If the biosphere can contain living organisms within various counterfactual histories, then the dynamical systems model of the biosphere must be able to contain agents under various initial conditions. In that case the agent representation must be able to represent all the agents in all the trajectories where they occur. If for two different initial conditions the degrees of freedom pertaining to agents at time t are different as well, then the agent representation must be able to exhibit this difference.

Related work

We should stress that we are only interested in work that relies on the *intrinsic* properties of the dynamical systems itself to represent agents. References to concepts like action, perception, and goal-directedness, if they are not defined in terms of the dynamical system are not acceptable in this case. The publication that most directly tackles the problem of agent representation that we are aware of is the insightful paper of Krakauer et al. (2014). They solve the problems of metabolism and motility by evaluating informa-

tional measures of closure and autonomy of sets of random variables. Given a system represented by a set of random variables at each point in time (i.e. represented by a dynamical Bayesian network as also defined below) they propose an algorithm that decides whether to include a random variable at a specific time step into the set representing the agent or leave it in the set representing the environment. This decision is made according to whether the inclusion into the agent contributes to the closure or autonomy of the agent. What this approach lacks however is the capability to deal with counterfactual variation. Since they use measures like mutual information and mutual conditional information that average over all states of the random variables in order to decide whether they belong to the agent or not, the partition of the random variables at each time step is fixed for all possible trajectories of the system. In order to deal with counterfactual variation it must be possible to have one partition into agent and environment for one trajectory and another partition for another trajectory. The same argument remains true for any approach that results in a fixed partition of the nodes in a dynamical Bayesian network. This includes the work of Balduzzi (2011) which results in a coarse-grained version of the network. The effective information that the glider contains about past states of the game of life, which was revealed in this work, should however be related the intrinsic spatiotemporal patterns that we investigate here.

Another very relevant and inspiring work is the work on the cognitive domain of the glider and autopoiesis in the game of life by Beer (2014b,a). This approach is capable of dealing with metabolism, motility, as well as counterfactual variation as it analyses spatiotemporal patterns and their internal mechanisms. The spatiotemporal patterns may have finite extension and can therefore occur or not occur within multiple trajectories at multiple times. The internal mechanisms are analyzed with respect to their production of the next *spatial* pattern inside the spatiotemporal pattern. The only caveat seems to be that the analysis is quite time consuming and does not have formal expressions of all the involved notions. We use the notion of spatiotemporal patterns as presented by Beer and hope that the measure we propose contributes to the formalization of the notions in his work.

An approach that seems to solve the problem of metabolism and counterfactual variation is the Markov blanket-based clustering used by Friston (2013). As the interacting degrees of freedom vary over time in a particle based system, it is possible to define a time dependent adjacency (or interaction) matrix. From this matrix Friston derives a Markov blanket matrix which can be used to classify the degrees of freedom into hidden, sensory, active, and internal states. This nicely defines an agent like structure within the degrees of freedom and through the time dependence of the adjacency and therefore also the Markov blanket matrix allows for the degrees of freedom to vary within a single trajectory and across initial conditions. In the case of

a field theoretical model where the adjacency of the degrees of freedom does not vary it is not directly obvious to us how to translate this. This means that motility could be a problem for the approach in such a model. However, it is definitely an alternative to our more information theory-based approach.

Methodologically, the framework of Lizier et al. (2014) for distributed computation is very closely related to ours. They investigate localized versions of mutual information and conditional mutual information to track and highlight information transfer, storage, and modification in dynamical Bayesian networks. This reveals spatiotemporal patterns very similar to ours. The main formal difference is in fact that instead of localizing the (conditional) mutual information we localize multi-information in the same way. In this way our work is just a trivial extension of this work. The focus of our work however is different as we are not so much interested in phenomena that are related to computation and more interested in revealing spatiotemporal entities or objects which might form the basis of an agent definition. Related work on spatiotemporal filtering (Shalizi et al., 2006; Flecker et al., 2011) of cellular automata differs from ours in a similar way. While "interesting" phenomena in the time evolution of single trajectories are revealed, the focus is not on connecting the interesting phenomena together in order to obtain entities.

Conceptually our work is also closely related to the integrated information theory due to Tononi et al. Originally (Tononi et al., 1994) this involved measurements of multi-information whose localized (in the sense of Lizier et al.) version we also employ as an estimate of integration. Newer versions (Oizumi et al., 2014; Albantakis and Tononi, 2015) involve a more elaborate construction which, importantly, is also localized in a certain way. The latter detect distinguished integrated spatial patterns which are constructed to resolve "what a system 'is' from its own intrinsic perspective" (Albantakis and Tononi, 2015). How these spatial structures are connected in time however is not treated. Our approach also aims at revealing intrinsic structure but crucially looks for spatiotemporal patterns i.e. patterns with a temporal as well as a spatial extension or compositional structure. In Tononi (2004); Balduzzi and Tononi (2008) temporal integration is mentioned with respect to optimal spatial and temporal scale or "grain size" detection. Our goal is different since we don't want to find a coarse-graining here. We want to reveal the complete lifetimes of agents as a single spatiotemporal pattern.

Dynamical Bayesian networks

Finite discrete-time distributed dynamical systems and their stochastic counterparts can be represented by dynamical Bayesian networks. Dynamical here just means that there is an interpretation of time in those networks. Distributed means that, at each time step, there are multiple given random variables whose states together define the state of the entire network at that time step.

More formally, a (dynamical) Bayesian network is a directed acyclic graph $G = (V, E)$ with nodes V and edges E. Each node i has an associated random variable X_i with state space $\mathcal{X}$ taking values $x_i \in \mathcal{X}$ (for simplicity we assume that all nodes have identical state spaces but this is not necessary for the definitions to hold). Furthermore each node is equipped with a mechanism $p_i(x_i|x_{\mathrm{pa}(i)})$ which gives the conditional probability distribution of X_i given the parents $\mathrm{pa}(i)$ of node i in G. Note that for any set $A \subseteq V$ we write $X_A := (\{X_i | i \in A\})$ for the random variable composed of the random variables in A. We assume that our network has a set V_0 of nodes without parents. As Ay and Polani (2008) note we can then define a partition of V into $(V_0, V_1, V_2, ...)$ (called *time slices*) where $V_{t+1} := \{i \in V \mid \exists j \in V_t, \mathrm{pa}(i) = j\}$. In general $\mathrm{pa}(V_{t+1}) \subseteq V_t$ since some nodes might not have any children. Here we assume $\mathrm{pa}(V_{t+1}) = V_t$. This allows us to interpret the various nodes in each V_t as those nodes representing the state of the distributed system at time t. We can also interpret the cardinality of the set V_t as the spatial extension of the state. In this paper this cardinality does not change over time just as e.g. in cellular automata.

The defining property of Bayesian networks (including dynamic ones) is that the joint probability distribution[2] p_V can be factorized in a way compatible with the structure of the graph G i.e.:

$$p_V(x_V) = \prod_{i \in V} p_i(x_i|x_{\mathrm{pa}(i)}). \qquad (1)$$

To relate the dynamical Bayesian network to dynamical systems note that the role of the dynamical law is played by the product of all mechanisms in V_t:

$$p_{V_{t+1}}(x_{V_{t+1}}) = \sum_{x_{V_t}} \prod_{i \in V_{t+1}} p_i(x_i|x_{\mathrm{pa}(i)}) p_{V_t}(x_{V_t}). \qquad (2)$$

Recall that $\bigcup_{i \in V_{t+1}} \mathrm{pa}(i) = V_t$ by definition. We can also write the above in terms of the Markov matrix:

$$p(x_{V_{t+1}}|x_{V_t}) = \prod_{i \in V_{t+1}} p_i(x_i|x_{\mathrm{pa}(i)}). \qquad (3)$$

In order to equip the dynamical Bayesian network with a join probability distribution p_V we then only have to define an initial probability distribution p_{V_0} and propagate it throughout the network according to Eq. 2.

Trajectories and spatiotemporal patterns

Here we formally define the notion of trajectories and spatiotemporal patterns. The class of the spatiotemporal patterns is very large and includes patterns that are of no specific interest. How we distinguish between those and more important patterns will be defined in the next sections.

---

[2]For a set of nodes A we write p_A for the probability distribution $p_A : \mathcal{X}^A \to [0, 1]$.

A *spatiotemporal pattern* x_O of a *dynamical Bayesian network on graph* $G = (V, E)$ is a set of nodes $O \subseteq V$ together a set of particular values $\{x_i \in \mathcal{X} | i \in O\}$.

A *trajectory* x_V of a *dynamical Bayesian network on graph* $G = (V, E)$ is a spatiotemporal pattern with $p(x_V) > 0$. In our setting this also means that there is an initial condition x_{V_0} such that x_V is possible under the time evolution induced by the Markov matrix or dynamical law.

We say that the spatiotemporal pattern x_O *occurs in a trajectory* x_V of the network iff $x_O \subseteq x_V$.

Employing the time slices V_t of the network we can also look at the time slices $x_{O_t} := x_O \cap x_{V_t}$ of any spatiotemporal pattern x_O.

Integrated spatiotemporal patterns

This section defines the notion of an integrated spatiotemporal pattern. Such patterns obey a condition which distinguishes them within the class of all spatiotemporal patterns. First we fix some further terminology.

We define the *the evidence for integration of an object O with respect to a partition π of O* as the local mutual information

$$\mathrm{mi}_\pi(x_O) := \begin{cases} 0 & \text{if } p_O(x_O) = 0, \\ \log \frac{p_O(x_O)}{\prod_{b_j \in \pi} p_{b_j}(x_{b_j})} & \text{else.} \end{cases} \quad (4)$$

Then, we say a spatiotemporal pattern x_O is *integrated* iff for all possible partitions π of the set O of random variables the evidence for integration of O with respect to π is positive. Considering all possible partitions is also done by Albantakis and Tononi (2015).

The interpretation of this is the following. The joint probability $p_O(x_O)$ is the probability that all the x_i with $i \in O$ occur together within single trajectories. I.e. among all trajectories there are those in which O occurs and their probability contributes to this joint probability. The probability $\prod_{b_j \in \pi} p_{b_j}(x_{b_j})$ however is the product of the probabilities that each part x_{b_j} occurs by itself in any trajectory *including* as part of x_O. If a part of x_O often occurs by itself without the rest of x_O occurring then this reduces the evidence for integration of O. This makes sense if we want to interpret the integrated spatiotemporal patterns as persistent objects like rocks, crystals, but also living organisms. If we consider for example a rock, the probability that a rock occurs at some point in time without a rock occurring at the previous and next time step in close vicinity is quite low, whereas the probability that where there was a rock before there will be a rock shortly after is quite high. In fact anytime that a spatial pattern (a time slice of a spatiotemporal pattern) causes (in an intuitive sense) another spatial pattern at the next time step their joint probability will rise and especially if the first spatial pattern is among the only causes of the second, their evidence for integration will be high.

What about the spatial integration however? The existence of rock in one place probably does increase the probability for more rock to be around it, but not extremely. It is perfectly possible and occurs frequently, that the rock ends, also that it is just a small piece of rock. So the evidence for spatial integration might not be so strong.

Now if we turn to living organisms, the evidence for temporal integration should also be high since they are autopoietic. Their spatial integration will probably be higher than that of rocks (and crystals) as half a bacterium is much less likely than a whole whereas half a rock is still a rock and those are not so uncommon. This reasoning scales up to larger living organisms.

We note that the evidence can also be interpreted in more information-theoretic terms. For example as the superfluous length of a codeword for the sequence x_O when we base the encoding on the product probability distribution $\prod_{b_j \in \pi} p_{b_j}(x_{b_j})$ instead of on the joint probability. This will be discussed in more detail in future work. We also intend to investigate in how far the integrated spatiotemporal patterns are independent of (possibly moving) frames of reference. Since they are not only integrated across time slices but instead across any partition we are optimistic in this regard.

Integrated spatiotemporal patterns and the tracking problem

The spatiotemporal patterns can solve the tracking problem. To see this take the perspective of time slices and say that at some time t a living organism is a configuration of degrees of freedom which increases the probability of a particular configuration of other degrees of freedom at a subsequent time $t + \epsilon$ that is again a living organism. More specifically, a living organism will lack certain molecules before absorbing them, conversely there will be a surplus of other molecules before they are ejected from the living organism. Therefore the probability for molecular exchange will be higher than for maintaining the same composition. This means the spatiotemporal patterns traversing the degrees of freedom associated to the molecules will have a higher evidence for integration over time. Similarly, in the field-theoretic setting the field configuration represented by the spatial pattern will increase the probability of the neighboring degrees of freedom to assume a certain configuration. This leads to more evidence for the integration of the moving pattern.

With respect to the problem of counterfactual variation we can see the following. Integration is calculated directly for spatiotemporal patterns within a trajectory and the local mutual information vanishes for all spatiotemporal patterns that do not occur in this trajectory (see Eq. 4). Then, if the spatiotemporal patterns that occur in different trajectories are different, the integrated spatiotemporal patterns will also be different. Thus, if integrated spatiotemporal patterns represent agents, these can occur on some degrees of freedom in one trajectory and not occur on those in another. This means

counterfactual variation won't be a problem.

Experimental indications

We present here the results of three preliminary experiments. The first is conceived to hint at the kind of trajectories that show high evidence of integration. The second experiment suggests that traversal of degrees of freedom or motility/metabolism can at least in principle be detected by integration. Similarly the third experiment shows that in principle counterfactual variation is no obstacle for integration.

All experiments use a 4×4 grid with game-of-life dynamics and toroidal boundary conditions as the distributed dynamical system. As the initial distribution p_{V_0} we use the uniform distribution in order to explore the whole range of possible trajectories. We investigate only patterns covering three time steps $t = 8, 9, 10$ and thereby neglect a lot of transient patterns that are difficult to interpret. In principle, however, our method does apply to transient patterns as well. Instead of integration we calculated only the *evidence for integration with respect to the finest possible partition* (EV-IFPP). The finest possible partition of a set A of nodes is just the partition where each block is a set containing exactly one node in A. A positive EVIFPP is a necessary condition for integration and therefore a crude indication for it.

For the first experiment we looked at all trajectories that differ at time steps $t = 8, 9, 10$ (a lot of trajectories end up with all cells white at those times). For each of those trajectories we calculated the EVIFPP for the spatiotemporal pattern $x_O = (x_{V_8}, x_{V_9}, x_{V_{10}})$. So the time slices of x_O are global states in this case. Since the x_O are global states this is more an evaluation of the integration of the trajectories that result from the different initial conditions. In Fig. 1 five such different global three-time-step patterns with high values of integration are shown including the completely blank spatiotemporal pattern and the spatiotemporal patterns (ignoring symmetric versions) with the highest EVIFPP. We can see that the blank spatiotemporal pattern has positive but much lower EVIFPP than some other patterns. For the second experiment we chose a specific trajectory shown in the first row in Fig. 2 which exhibits a moving pattern and searched through all patterns covering time steps $t = 8, 9, 10$ and fixing $n = 14$ cells (i.e. nodes of the dynamic Bayesian network) in each time slice x_{O_t}. We can see that the degrees of freedom (i.e. the cells or nodes) making up both the spatiotemporal pattern with minimal EVIFPP (second row in Fig. 2) as well as that with maximal EVIFPP (third row in Fig. 2) vary over the three time steps and adapt to the configuration of the global state. Note that the patterns with minimal and maximal EVIFPP are not unique.

For the third experiment changed the initial condition of the trajectory by shifting all values of the initial condition of the second experiment "down" one cell. This results in a different trajectory shown in the first row of Fig. 3. We then evaluated the spatiotemporal pattern that results from fixing
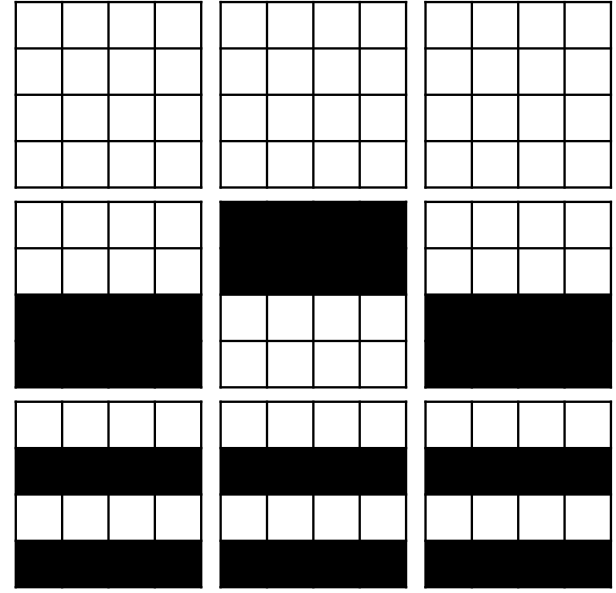
Figure 1: Three three-time step spatiotemporal patterns. Each row shows the three global spatial patterns that make up the spatiotemporal pattern x_O. The first row shows the blank spatiotemporal pattern and the others show the two patterns with the highest EVIFPP. The EVIFPP values are (from top to bottom) 4.9, 81.9, and 85.4 respectively.

the same nodes as in the spatiotemporal pattern with maximal EVIFPP found in the second experiment on the changed trajectory (see row two in Fig. 3). We also evaluated the EVIFPP of the spatiotemporal pattern that results from shifting the fixed nodes of maximal EVIFPP pattern in the same way as the initial condition (see row three in Fig. 3). The pattern with the same fixed nodes as the pattern that formerly had maximal EVIFPP now has lower EVIFPP than the pattern with the nodes adapted to the new initial condition.

Discussion

The first experiment shows that the completely blank trajectory has low spatiotemporal EVIFPP and that more "interesting" trajectories have higher EVIFPP (Fig. 1). This can also be done with other methods e.g. counting black cells. However, our method is general and doesn't use any prior knowledge e.g. which color of cells to count. For us this result is a necessary condition for further investigation.

The second experiment shows that the degrees of freedom pertaining to spatiotemporal patterns with high EVIFPP adapt over time to the changing configurations of the system. This shows that EVIFPP is capable of solving the metabolism and motility problems. We expect that the same holds true for evidence of integration with respect to any partition and therefore also for integration itself.

The third experiment shows that under a variation of the initial condition the degrees of freedom pertaining to spa-
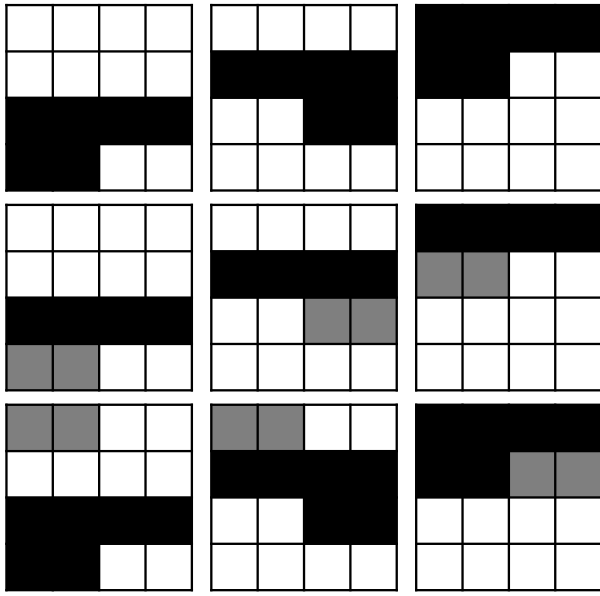
Figure 2: A three-time step part of a trajectory (can also be seen as a global spatiotemporal pattern) in the first row and two local spatiotemporal patterns on this trajectory in the second and third row. Both spatiotemporal patterns in rows two and three have $n = 14$ specified cells per time slice. The second (third) row shows a pattern attaining the minimal (maximal) EVIFPP of 32.5 (54.4) among all patterns with $n = 14$ on the trajectory of row one. The global spatiotemporal pattern of row one has EVIFPP of 55.0.

tiotemporal patterns with high EVIFPP change accordingly. Since the different trajectories generated from changed initial conditions correspond to counterfactual histories this shows that the EVIFPP solves the problem of counterfactual variation. Again we expect this to carry over to integration.

We note that larger grids become hard to evaluate computationally very fast. For square grids the size of the Markov matrix grows with 2^{a^2} where a is the number of rows and columns of the grid. We also note that due to the very limited grid size we are studying any pair of cells is just separated by maximally one neighborhood cell. This leads to strong dependencies which might make it irrelevant to place unspecified cells around patterns like the blinker (as for example suggested by Beer (2014a)). We had hoped to reveal such well known patterns and their extensions. Turning to larger grids is a next step in our research.

Conclusion

We have presented our current approach to representing agents in dynamical systems. Three criteria that we expect from such an agent representation were motivated with a thought experiment involving a dynamical systems model of the biosphere. The literature was reviewed in the light of these criteria. We also introduced our current candidate

measure for identifying intrinsic spatiotemporal patterns in dynamical Bayesian networks. These patterns form the basic building blocks of our approach to representing agents. We argued that this approach can deal with the three criteria for agent representations that we have put forward. Experimentally we verified this for a crude approximation to our more involved concept of integration. However experimental results are currently inconclusive with respect to the tracking of structures that are actually relevant for agents. Therefore we see the value of this work mostly as a contribution to the discussion of the foundations of artificial life. Future work will bring more decisive results.
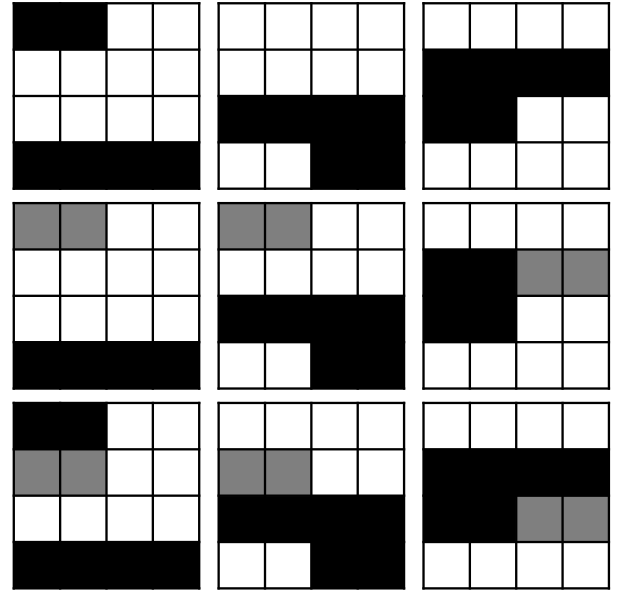
Figure 3: The three-time step part of the trajectory that results from shifting the values of the initial condition of the trajectory in the first row of Fig. 2 "down" by one cell. The second row shows the spatiotemporal pattern with the same fixed nodes as the pattern that had maximal EVIFPP on the trajectory of Fig. 2 but now on the shifted trajectory. The EVIFPP of this is 39.8. The third row shows the spatiotemporal pattern with the fixed nodes shifted "down" in the same way as the initial condition. This pattern has EVIFPP of 54.4 and is the maximal EVIFPP for patterns with $n = 2$. As expected this is the same value we found for the pattern with the non-shifted nodes on the non-shifted trajectory.

References

Albantakis, L. and Tononi, G. (2015). The Intrinsic Cause-Effect Power of Discrete Dynamical SystemsFrom Elementary Cellular Automata to Adapting Animats. *Entropy*, 17(8):5472–5502.

Ay, N. and Polani, D. (2008). Information flows in causal networks. *Advances in Complex Systems*, 11(01):17–41.

Balduzzi, D. (2011). Detecting emergent processes in cellular automata with excess information. *Advances in Artificial Life, ECAL*, abs/1105.0158.

Balduzzi, D. and Tononi, G. (2008). Integrated information in discrete dynamical systems: Motivation and theoretical framework. *PLoS Comput Biol*, 4(6):e1000091.

Barandiaran, X. E., Paolo, E. D., and Rohde, M. (2009). Defining agency: Individuality, normativity, asymmetry, and spatio-temporality in action. *Adaptive Behavior*, 17(5):367–386.

Bartlett, S. and Bullock, S. (2015). Emergence of Competition between Different Dissipative Structures for the Same Free Energy Source. In *Proceedings of the European Conference on Artificial Life*, pages 415–422. The MIT Press.

Beer, R. D. (2014a). Characterizing autopoiesis in the game of life. *Artificial Life*, 21(1):1–19.

Beer, R. D. (2014b). The cognitive domain of a glider in the game of life. *Artificial Life*, 20(2):183–206.

Cleland, C. E. and Chyba, C. F. (2002). Defining Life. *Origins of life and evolution of the biosphere*, 32(4):387–393.

Dyson, F. J. (1985). *Origins of Life*. Cambridge University Press.

Flecker, B., Alford, W., Beggs, J. M., Williams, P. L., and Beer, R. D. (2011). Partial information decomposition as a spatiotemporal filter. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 21(3):037104.

Friston, K. (2013). Life as we know it. *Journal of The Royal Society Interface*, 10(86).

Froese, T., Virgo, N., and Ikegami, T. (2014). Motility at the origin of life: Its characterization and a model. *Artificial Life*, 20(1):55–76.

Froese, T. and Ziemke, T. (2009). Enactive artificial intelligence: Investigating the systemic organization of life and mind. *Artificial Intelligence*, 173(3–4):466–500.

Ikegami, T. and Taiji, M. (1998). Uncertainty, possible worlds and coupled dynamical recognizers.

Kauffman, S. A. (2002). *Investigations*. Oxford University Press, Oxford; New York.

Krakauer, D., Bertschinger, N., Olbrich, E., Ay, N., and Flack, J. C. (2014). The information theory of individuality. *arXiv:1412.2447 [q-bio]*. arXiv: 1412.2447.

Langton, C. (1989). Artificial life. In Langton, C., editor, *Artificial Life*, pages 1–47. Addison-Wesley.

Lizier, J. T., Prokopenko, M., and Zomaya, A. Y. (2014). A framework for the local information dynamics of distributed computation in complex systems. In Prokopenko, M., editor, *Guided Self-Organization: Inception*, number 9 in Emergence, Complexity and Computation, pages 115–158. Springer Berlin Heidelberg.

Maturana, H. R. and Varela, F. J. (1980). *Autopoiesis and cognition: the realization of the living*. Springer.

Oizumi, M., Albantakis, L., and Tononi, G. (2014). From the Phenomenology to the Mechanisms of Consciousness: Integrated Information Theory 3.0. *PLoS Comput Biol*, 10(5):e1003588.

Russell, S. J. and Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, N.J, 1st edition edition.

Schlosser, M. (2015). Agency. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Fall 2015 edition.

Shalizi, C. R., Haslinger, R., Rouquier, J.-B., Klinkner, K. L., and Moore, C. (2006). Automatic filters for the detection of coherent structure in spatiotemporal systems. *Physical Review E*, 73(3):036104.

Szathmáry, E., Santos, M., and Fernando, C. (2005). Evolutionary Potential and Requirements for Minimal Protocells. In Walde, P., editor, *Prebiotic Chemistry*, number 259 in Topics in Current Chemistry, pages 167–211. Springer Berlin Heidelberg. DOI: 10.1007/tcc001.

Tononi, G. (2004). An information integration theory of consciousness. *BMC Neuroscience*, 5:42.

Tononi, G., Sporns, O., and Edelman, G. M. (1994). A measure for brain complexity: relating functional segregation and integration in the nervous system. *Proceedings of the National Academy of Sciences*, 91(11):5033–5037.

Virgo, N. (2011). *Thermodynamics and the Structure of Living Systems*. University of Sussex. Unpublished PhD thesis.

Complexity and Structural Properties in Scale-free Networks

Yesid Madrid[1,2], Carlos Gershenson[3,4] and Nelson Fernández[1]

[1] Laboratorio de Investigaciones en Hidroinformática, Universidad de Pamplona, 543050 Pamplona, Colombia.
[2] Grupo de Investigación en Ciencias Computacionales-CICOM, Universidad de Pamplona, 543050 Pamplona, Colombia.
[3] Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas, Universidad Nacional Autónoma de México
[4] Massachusets Institute of Technology, Cambridge, MA 02139, USA
nfernandez@unipamplona.edu.co

Abstract

We apply formal information measures of emergence, self-organization and complexity to scale-free random networks, to explore their association with structural indicators of network topology. Results show that the cumulative number of nodes and edges coincides with an increment of the self-organization and relative complexity, and a loss of the emergence and complexity. Our approach shows a complementary way of studying networks in terms of information.

Introduction

Among representative structural properties of networks we can list: the degree of nodes and their distribution, the clustering coefficient, and the average path length. The degree is informative of how many nodes are connected to each other. The clustering coefficient is a measure of the number of triangles in a graph. The average path length is the average number of steps along the shortest paths between all possible pairs of network nodes (Newman et al., 2006). In spite of the value of these measures to characterize some complex networks, measuring complexity in networks is desirable. Recently, measures of emergence, self-organization, complexity, and relative complexity based on information theory have been developed and their usefulness can be evaluated (Fernández et al., 2014).

In this paper we analyze the association of topological structural indicators like the number of nodes, clustering coefficient and average path length with formal measures of emergence, self-organization, complexity, relative complexity to scale-free random networks.

In the next section we present the methods for generating networks and the formalism to measure complexity. In section 3, we briefly present and discuss our results obtained from the applications of multivariated machine learning unsupervised techniques. Section 4 presents conclusions and future work.

Methods

Using the Barabasi-Albert model implemented in SocNetV software (Kalamaras D., 2015), ten random networks of the following number of nodes were generated: 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 125, 150, 175, 200, 225, 250, 300, 350, 400, 450, 500, 600, 700, 800, 900, 1000, 2000 and 3000. In total, 310 scale-free networks were created. First, structural properties of clustering coefficient and average path length were calculated. Then, information measures of emergence (E), self-organization (S), complexity (C), and relative complexity (R) were applied to the vector obtained from the horizontal sum of the grade of each node, on the adjacency matrix. Summarizing, E is equivalent to Shannon information (Shannon, 1948), depending of the probabilities p_i for all i symbols in a finite alphabet: $I = -\sum_{i=1}^{n} p_i \log p_i$. In this work, we use $\log_{10}$. Based on this equation we define that $E = I$, $S = 1 - E$, and $C = 4 \times E \times S$. Relative Complexity $R = CN_i/CN_T$ where CN_i was the complexity of the network i and CN_T was the average complexity of the other networks. Since $E, S, C \in [0, 1]$ a numerical, color and category scale has been defined for a better interpretation. The ranges are $[0.8, 1]$, $[0.6, 0.8)$, $[0.4, 0.6)$, $[0.2, 0.4)$,$[0, 0.2)$. The corresponding colors are: blue, green, yellow, orange, and red. The matching categories are: very high, high, fair, low, and very low. R just has two colors of codification: blue if $R > 1$ and red if $R < 1$.

To facilitate the visualization of the relationship of complexity properties and network structure descriptors, according to the increment of nodes and edges, multivariated techniques was carried out. Consequently, a principal component analysis (PCA) was integrated with a hierarchical cluster analysis ($HCPC$). PCA was used to summarize and to visualize the information contained in structural attributes and complexity properties. $HCPC$ was used for identifying clusters of networks with similar characteristics. Also, statistical indicators such as *v-test* (a criterion of normal distribution), mean in cluster, overal average, and p-values were estimated to associated clusters and properties (Le and Worch, 2015).

Results and Discussion

The PCA depicts the relationship, variation, and patterns among structural and complexity properties (fig. 1). A high percentage of the explained variance of data was captured in the first two axes (91.46%). In consideration of the po-

sition of variables in the multidimensional space, it is easy to see that self-organization and relative complexity are positively correlated with the increasing of nodes and edges. Meanwhile, when nodes, edges, self-organization, and relative complexity increase, the complexity and emergence of the network decreases. These two groups, in spite of the fact that they have an opposite behavior, are very close to the first component (Dim 1) and explain the variance of scale-free networks in 74.24%. As a relevant fact, it is possible to see that complexity is more related to the change (emergence) in the system than its regularity (self-organization). This suggest that some adaptability of scale-free networks could be related to high variability with a minor proportion of uniformity in the degree distribution of the nodes.

Considering the structural indicators of clustering coefficient and average path length, we can observe that they are opposite as it has been noticed in the literature. They are associated with the second component and represent a minor variance explained of the dataset (12.22%). As the form of clustering coefficient and average path length are positioned, we cannot establish any relation between them and emergence, self-organization, complexity, and relative complexity.
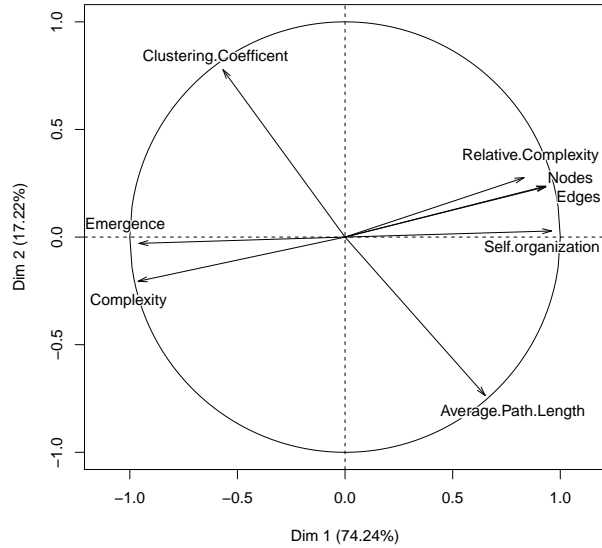
Figure 1: Principal component analysis of complexity and structural properties in 310 simulations of random scale-free networks obtained using BA model

Regarding the incremental number of nodes, *HCPC* analysis allows the statistical creation and characterization of five clusters (table 1). The first cluster included networks with just five nodes. These systems are related statistically with the clustering coefficient, because of a relatively high density of ties in small networks. Cluster two groups networks with 10 to 50 nodes, indicating that they are the most emergent and complex of all. Indeed, in cluster two, E reached the fair category (yellow), in comparison with the

low category of overall (in orange). C was classified as very high (blue), meanwhile overall was categorized as high (green). Besides, we can note that these small networks have a negative relationship with the average path length which is less than overall. Networks between 60-175 nodes were grouped in cluster 3. They also have a very high complexity. Cluster four included networks with 100-600 nodes and are the most self-organized due to having a very high S. Finally, networks with the highest number of nodes (700-3000) gain some relative complexity. That means the increase of the number of nodes and edges resulting in networks moderately more complex.

Table 1: Statistical description for Clusters and Structural Properties in Scale-free Networks.

Cluster	Number of Nodes in Network Grouped	Property Associated	V.Test	Property μ in cluster	Property Overall μ	p-value/Significance
1	5	C.C.	4.808	0.213	0.019	1.521^{06}***
2	10,15,20,25,30 35,40,45,50	E	4.316	0.545	0.304	$1.586e^{05}$***
		C	3.177	0.949	0.676	1.586^{-05}***
		Av.P.Length.	-3.165	2.283	2.619	1.553^{-03}
3	60,70,80,90,100 125,150,175	C	2.116	0.873	0.676	0.034*
4	200,225,250,300,350 400,450,500,600	S	2.928	0.872	0.045	0.003**
5	700,800,900 1000,2000,3000	R	4.358	1.007	1.003	1.308e-05***
		S	3.082	0.945	0.696	2.049^{03}***

Final Remarks

Our first results are encouraging. It was interesting to find that growth in random scale-free graphs implies more self-organization and relative complexity. The relative complexity could be useful to analyze cases when two or more networks interact. Thus, the gain of self-organization and relative complexity could in time be a useful characteristic to regulate feedback and guide the management of complex networks.

Further work is required. We are planning to broaden our explorations and perform further analysis to understand and clarify the relationship between complexity properties and structural indicators in random networks, and other complex topologies.

References

Fernández, N., Maldonado, C., and Gershenson, C. (2014). Information measures of complexity, emergence, self-organization, homeostasis, and autopoiesis. In Prokopenko, M., editor, *Guided Self-Organization: Inception*, volume 9 of *Emergence, Complexity and Computation*, pages 19–51. Springer, Berlin Heidelberg.

Kalamaras D. (2015). Social Network Visualizer (SocNetV). Social network analysis and visualization software.

Le, S. and Worch, T. (2015). *Analyzing Sensory Data with R*. Chapman and Hall/CRC, Boca Raton, FL 33487-2742.

Newman, M., Barabasi, A.-L., and Watts, D. J. (2006). *The structure and dynamics of networks*. Princeton University Press.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27(3 and 4):379–423 and 623–656.

Author Index