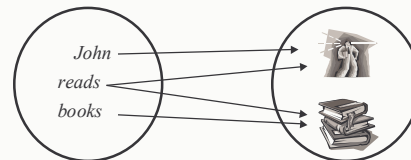


## Session 3

### Regular Languages and Expressions

## Syntax and Semantics

- Normally languages refer to individual objects, properties and relations in the world
- Linguistic symbols are *syntactic objects*
- The representation or reference relation:

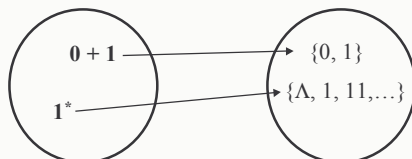


Language Refers to The world

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Syntax and Semantics

- If language is the object of study, we need a language to be able to talk about languages
- Sets of strings (languages) become *semantic objects*
- The representation or reference relation:



Language Refers to The world of languages

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## The language of Regular Expressions

- Syntax
  - Basic constants (for a given  $\Sigma$ )
    - $\Phi$  is regular expression (*RE*)
    - $\Lambda$  is *RE*
    - If  $a \in \Sigma$  then  $a$  is a *RE*
    - A variable, a italic capital letter (e.g.  $L$ ), is a *RE*
  - Composition rules:
    - If  $E$  and  $F$  are *RE* then  $E + F$  is *RE* (union)
    - If  $E$  and  $F$  are *RE* then  $EF$  is *RE* (concatenation)
    - If  $E$  is a *RE* then  $E^*$  is a *RE* (closure)
    - If  $E$  is a *RE* then  $(E)$  is a *RE* (introduction of parenthesis)
  - Only the expressions constructed by a FINITE application of the rules in this definition are *RE*

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## An alternative formulation

- Syntax
  - Basic constants (for a given  $\Sigma$ )
    - $\Phi$  is regular expression *RE*
    - $\Lambda$  is *RE*
    - If  $a \in \Sigma$  then  $a$  is a *RE*
    - A variable, a italic capital letter (e.g.  $L$ ), is a *RE*
  - Composition rules (parenthesis are obligatory):
    - If  $E$  and  $F$  are *RE* then  $(E + F)$  is *RE* (union)
    - If  $E$  and  $F$  are *RE* then  $(EF)$  is *RE* (concatenation)
    - If  $E$  is a *RE* then  $(E^*)$  is a *RE* (closure)
  - Only the expressions constructed by a FINITE application of the rules in this definition are *RE*

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

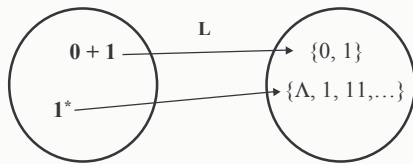
## The language of Regular Expressions

- Semantics
  - Let  $RE$  the set of all regular expressions over  $\Sigma$ ,  $R$  the set of regular languages and  $L$  an interpretation function from  $RE$  to  $R$
  - Interpretation of basic constants
    - $L(\Phi)$  is  $\Phi$  (i.e. the empty language)
    - $L(\Lambda)$  is  $\{\Lambda\}$  (i.e. the language with the empty string)
    - If  $a \in \Sigma$  then  $L(a)$  is  $\{a\}$  (i.e. the language with  $a$ )
    - $L(L)$  is any language
  - Interpretation of composite expressions:
    - $L(E+F)$  is the union of  $L(E)$  and  $L(F)$
    - $L(EF)$  or  $L(E.F)$  is the concatenation of  $L(E)$  and  $L(F)$
    - $L(E^*)$  is  $(L(E))^*$  (i.e. the closure of  $L(E)$ )
    - $L((E))$  is  $L(E)$  (i.e. the same language)

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Syntax and Semantics

- The representation or reference function  $L$ :



- The interpretation of a composite expression is a function of:
  - The interpretation of its constituent parts
  - The form of syntactic composition

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Examples

$RE$	Language
• $\Lambda$ :	$\{\Lambda\}$
• $0$ :	$\{0\}$
• $001$ :	$\{001\}$
• $0 + 1$ :	$\{0, 1\}$
• $0 + 10$ :	$\{0, 10\}$
• $(1 + \Lambda)001$ :	$\{1, \Lambda\} \{001\}$
• $(110)^*(0 + 1)$ :	$\{110\}^* \{0, 1\}$
• $1^*10$ :	$\{1\}^* \{10\}$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Examples

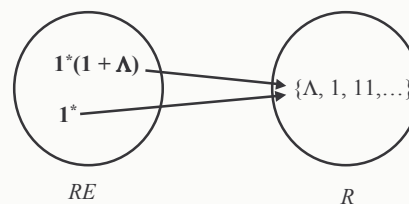
- $\dots (10 + 111 + 11010)^*$ :  $\{10, 111, 11010\}^*$
- $(0 + 10)^*((11)^* + 001 + \Lambda)$ :  $\{0, 10\}^* \{ \{11\}^* \cup \{001, \Lambda\} \}$
- $01^* + 1$ :  $\{0\} \{1\}^* \cup \{1\} = \{1, 0, 01, 011, \dots, 011\dots 1\}$
- $(01)^* + 1$ :  $\{01\}^* \cup \{1\} = \{1, \Lambda, 01, 0101, \dots, 0101\dots 01\}$
- $0(1^* + 1)$ :  $\{0\} \{ \{1\}^* \cup \{1\} \} = \{0, 01, 011, \dots, 011\dots 1\}$

Note that:  $0(1^* + 1) = 01^*$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Equality of Regular Expression

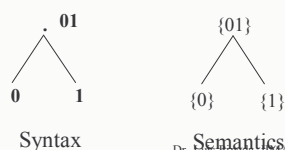
- Regular expressions are equal if they refer to the same language:



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

Example: The language of alternating 0's and 1's

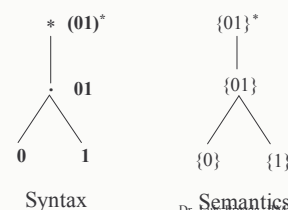
- First: The language  $\{01\}$ :  $01$



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

Example: The language of alternating 0's and 1's

- Second: The language  $\{01\}^*$ :  $(01)^*$

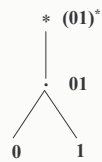


Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

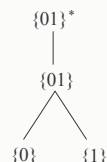
### Example: The language of alternating 0's and 1's

- But we also need:

- $\{0101\dots 0\}$
- $\{1010\dots 0\}$
- $\{1010\dots 1\}$



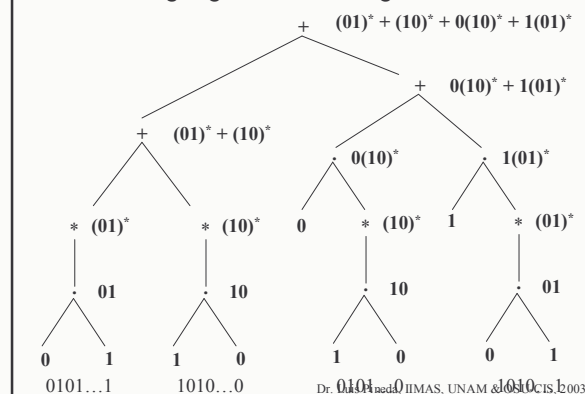
Syntax



Semantics

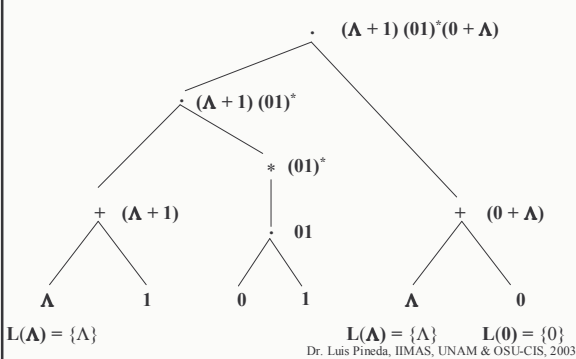
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### The language of alternating 0's and 1's



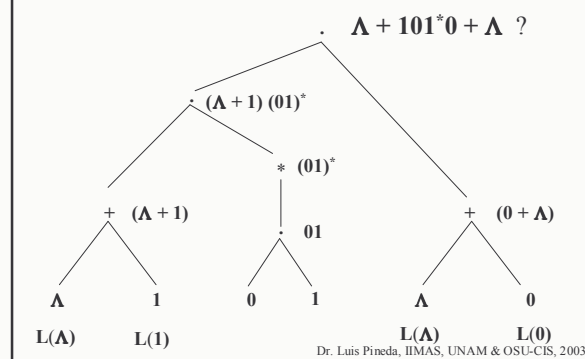
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### The expressive power of $\Lambda$



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

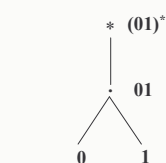
### Are RE ambiguous?



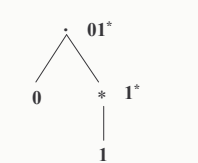
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Structure of RE

- Operators apply to the structure below:



$\{\Lambda, 01, 0101, \dots, 01\dots 01\}$

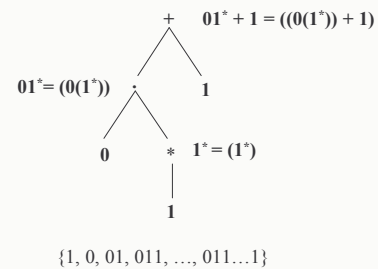


$\{0, 01, 011, \dots, 011\dots 1\}$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Structure and ambiguity

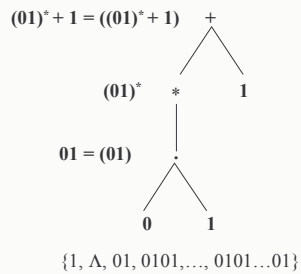
- "Ambiguity" of:  $01^* + 1$



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Structure and ambiguity

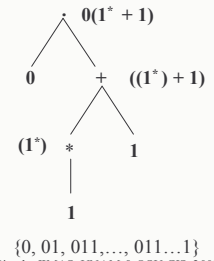
- “Ambiguity” of :  $01^* + 1$



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Structure and ambiguity

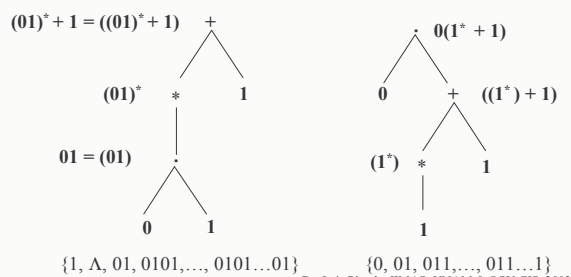
- “Ambiguity” of :  $01^* + 1$



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Structure and ambiguity

- “Ambiguity” of :  $01^* + 1$



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Precedence of operators

- Precedence order:
  - Highest: Star operator (\*)
    - Applies to the smallest sequence to its left
  - Next: Concatenation operator (dot)
    - juxtaposition of strings
    - Strings with no other operator in between are grouped together
    - Associative (conventionally we group by the left)
  - Lowest: Union operator (+)
    - Associative (conventionally we group by the left)

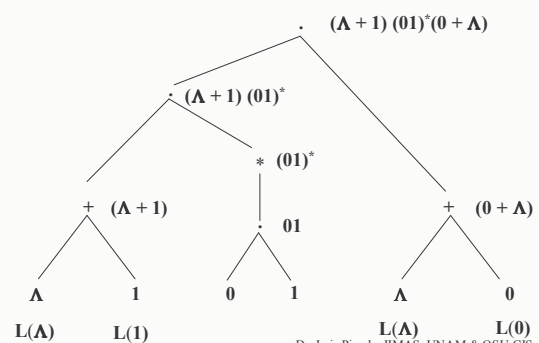
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Grouping “.” by the left

$$(\Lambda + 1)(01)^*(0 + \Lambda)$$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Grouping “.” by the left



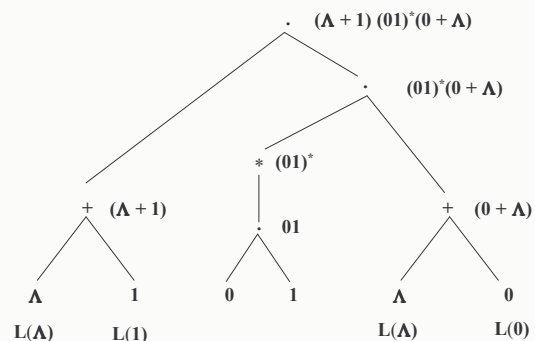
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Grouping “.” by the right

$$(\Lambda + 1) (01)^*(0 + \Lambda)$$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Grouping “.” by the right



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Precedence, parenthesis and ambiguity

- *RE* can apparently be ambiguous, but
  - Parenthesis and precedence order eliminate ambiguity
- Also:
  - *RE* have a structure
  - Trees show the structure of *RE* explicitly!
  - An ambiguous expression have several possible structures
  - There is only one structure for every *RE*
- Looking at precedence rules and parenthesis, or at the structure of expressions, there is no ambiguity
- *RE* are NOT ambiguous!

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Equality of Regular Expression

- Useful for simplifying expressions
- As we will see, useful for simplifying Automatas (with as few states as possible)
  - $1^*(1 + \Lambda) = 1^*$
  - $1^*1^* = 1^*$
  - $0^* + 1^* = 1^* + 0^*$
  - $(0^*1^*)^* = (0 + 1)^*$
  - $(0 + 1)^*01(0 + 1)^* + 1^*0^* = (0 + 1)^*$
- There is a general method (an algorithm) to decide whether two expressions define the same language

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Simplifying *RE*

- $(r + s + rs + sr)^*$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

### Simplifying *RE*

- $(r + s + rs + sr)^*$ 
  - $rs$  can be formed taking  $r$  and the  $s$ ; similarly for  $sr$ , then:
 
$$(r + s + rs + sr)^* = (r + s)^*$$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Simplifying RE

- $r(r^*r + r^*) + r^*$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Simplifying RE

- $r(r^*r + r^*) + r^*$ 
  - $r^*r = r^+$
  - $r(r^*r + r^*) + r^* = r(r^+ + r^*) + r^*$ 
    - $= r(r^*) + r^*$
    - $= rr^* + r^*$
    - $= r^+ + r^*$
    - $= r^*$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Interpreting RE

- Consider the two regular expressions:
  - $r = 0^* + 1^*$
  - $s = 01^* + 10^* + 1^*0 + (0^*1)^*$
- A string corresponding to  $r$  but not to  $s$ 
  - 00
- A string corresponding to  $s$  but not to  $r$ 
  - 01
- A string corresponding to both  $r$  and  $s$ 
  - Several obvious ones:  $\Lambda$ , 0, 1
- A string in  $\{0, 1\}^*$  corresponding to neither  $r$  or  $s$ 
  - Any string of the form:  $1^i0^i$  for  $i \geq 2$

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Finding RE

- Give a regular expression for the following language:
  - $\{s \in \{a, b\}^* : |s| \text{ is not divisible by } 2\}$
- If  $|s|$  is divisible by 2
  - its length is even (i.e. otherwise its length is odd)
- RE for strings of even length:
  - $(aa + ab + ba + bb)^*$
- Adding one symbol, either  $a$  or  $b$  (i.e. odd length strings):
  - $(aa + ab + ba + bb)^*(a + b)$
- Alternatively:
  - $(a + b)(aa + ab + ba + bb)^*$
- Introducing abstraction:
  - $(a + b)((a + b)(a + b))^*$

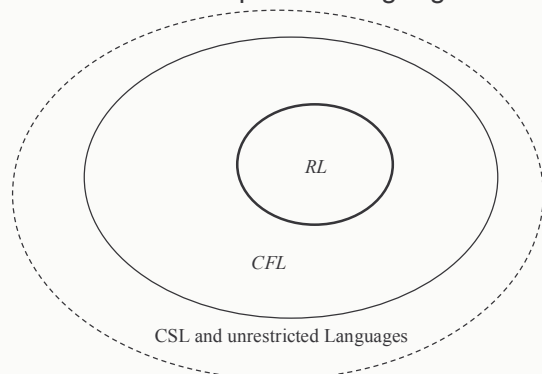
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## Regular sets

- Languages built out:
  - $\Phi$ ,  $\Lambda$  and all symbols in  $\Sigma$
- By means of:
  - Union
  - Concatenation
  - Closure (Kleen-star)
- Through a *finite* number of operations!
  - A regular expression is itself finite string!
  - Formally, we don't allow ellipsis (...)
  - The tree of the expression is finite too!
- The resulting set, a language, is subset of the power set of  $\Sigma^*$  ( $2^{\Sigma^*}$ ), which cannot even be counted!
- There are many, many, sets that are not regular!

Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

## RE in the space of languages



Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003