Session 12

Minimal Automata

What property makes a language regular?

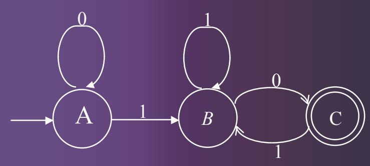
- Kleen's theorem: if we have a RE describing L or a FA accepting L, then L is regular
- But, what if we have a language described by some other means:
  - $L = \{0^n1^n \in \Sigma^* \mid n > 0\}$
  - Is this language regular?
- We know that if there are n classes of strings in L, a FA for recognizing L needs to have at least n states
  - All FA have a finite set of states
  - If the number of classes of strings of L is finite, there is a FA for recognizing L, and also a *RE* for describing L

Using the classes of strings

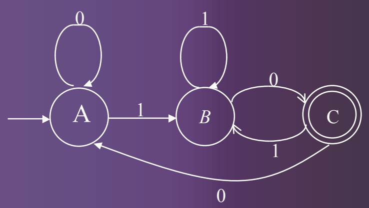
- We can tell what is the FA from the set of classes!
- A machine for the language of strings ending with 10:
  - Class A: The string needs 10 to be in the language
  - Class B: The string needs a 0 to be in the language
  - Class C: The string is in the language!

From the finite set of classes to the FA…

- A machine for strings ending with 10:
  - Class A: The string needs 10 to be in the language
  - Class B: The string needs a 0 to be in the language
  - Class C: The string is in the language!



Very sorry… it can get tricky!



- Fortunately, there is an algorithm for producing a FA from a finite set of classes of strings!

The classes of strings!

- We study the strings in the language directly: we don't have a FA or a *RE*!
- First, we need to identify the classes of strings:
  - Class A: The string needs 10 to be in the language: Λ, 0 or ends in 00
  - Class B: The string needs a 0 to be in the language: 1, 11 or ends in 01
  - Class C: The string ends with 10 and it is in the language!
- More generally, there is a distinguishing string z to get strings of each class into the language:

$$L/x = \{z \in \Sigma^* \mid xz \in L\}$$

Distinguishing strings

- Let L be a language in Σ^*, and $x \in \Sigma^*$
- Let L/x a set of strings such that:
  - $L/x = \{z \in \Sigma^* \mid xz \in L\}$
- Two strings x and y are distinguishable with respect to L if
  - $L/x \neq L/y$
- Any string z such that $xz \in L$ and $yz \notin L$ or vice versa, is said to distinguish x and y with respect to L
- If $L/x = L/y$, x and y are indistinguishable (or equivalent) with respect to L

The classes of strings!

- Class A:
  - Λ
  - 0
  - Σ^*00
- Class B:
  - 1
  - 11
  - Σ^*01
- Class C:
  - Σ^*10

Distinguishing the classes of strings!

- $z = 10$ and $L/x = \{z \in \Sigma^* \mid xz \in L\}$
  - $\Lambda 10$
  - 010
  - Σ^*0010
- $z = 0$ and $L/y = \{z \in \Sigma^* \mid yz \in L\}$
  - 10
  - 110
  - Σ^*010
- $z = \Lambda$ and $L/u = \{z \in \Sigma^* \mid uz \in L\}$
  - $\Sigma^*10\Lambda$
- We have one z for each class of strings!

distinguishing *L/x* from *L/y*

- Let $z = 0$
  - $L/x = \{z \in \Sigma^* \mid xz \notin L\}$
    - $\Lambda 0$
    - 00
    - Σ^*000
  - $L/y = \{z \in \Sigma^* \mid yz \in L\}$
    - 10
    - 110
    - Σ^*010
- $L/x \neq L/y$

distinguishing *L/x* from *L/u*

- Let $z = \Lambda$
  - $L/x = \{z \in \Sigma^* \mid xz \notin L\}$
    - $\Lambda\Lambda$
    - 0Λ
    - $\Sigma^*00\Lambda$
  - $L/u = \{z \in \Sigma^* \mid uz \in L\}$
    - $\Sigma^*10\Lambda$
- $L/x \neq L/u$

distinguishing *L/y* from *L/u*

- Let $z = 0$
  - $L/y = \{z \in \Sigma^* \mid yz \in L\}$
    - 10
    - 110
    - Σ^*010
  - $L/u = \{z \in \Sigma^* \mid uz \notin L\}$
    - Σ^*100
- $L/y \neq L/u$

The indistinguishability relation I_L

- The indistinguishability relation I_L on Σ^* is defined by

  $x\ I_L\ y$ if and only if $L/x = L/y$
- I_L is an equivalent relation on Σ^*
  - Reflexive
  - Transitive
  - Symmetric
- Notation:
  - $[x]$ denotes the equivalent class containing x
- Two strings are distinguishable with respect to L if they are in different classes of I_L

Properties of I_L

- Any string within the same partition of I_L can be considered the same for operations on strings involving I_L
- I_L is right invariant with respect to concatenation:
  - For $x, y \in \Sigma^*$ and any $a \in \Sigma$,

    if $x\ I_L\ y$ then $xa\ I_L\ ya$

    if $[x] = [y]$ then $[xa] = [ya]$

The algorithm

- Let $L \subseteq \Sigma^*$ and Q_L the set of equivalent classes of I_L on Σ^*. If Q_L is finite then $M_L = (Q_L, \Sigma, q_0, A_L, \delta)$ is FA accepting L, where
  - $q_0 = [\Lambda]$
  - $A_L = \{q \in Q_L \mid q \cap L \neq \Phi\}$
  - δ: Q_L x $\Sigma \rightarrow Q_L$ is

    $\delta([x], a) = [xa]$
- M_L is the FA with the minimum number of states that accepts L
- Extended transition function for M_L:

  $\delta^*([x], y) = [xy]$

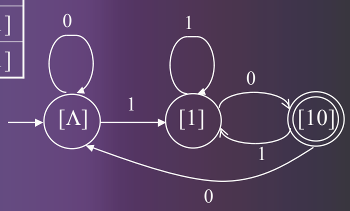
Now we can make sure!

- $M_L = (Q_L, \{0, 1\}, [\Lambda], \{[10]\}, \delta)$ where
  - $Q_L = \{[\Lambda], [1], [10]\}$
    - $[\Lambda] = \{\Lambda, 0, \Sigma^*00\}$
    - $[1] = \{1, 11, \Sigma^*01\}$
    - $[10] = \{\Sigma^*10\}$
- Definition of δ: $\delta([x], a) = [xa]$
  - $\delta([\Lambda], 0) = [\Lambda 0] = [0] = [\Lambda]$ and $\delta([\Lambda], 1) = [\Lambda 1] = [1]$
  - $\delta([1], 0) = [10]$ and $\delta([1], 1) = [11] = [1]$
  - $\delta([10], 0) = [100] = [\Lambda]$ and $\delta([10], 1) = [101] = [1]$
- Notice that you can choose any string in the class as the name of the class: they all are equivalent!

Now we can make sure!

- $M_L = (Q_L, \{0, 1\}, [\Lambda], \{[10]\}, \delta)$ where
  - $Q_L = \{[\Lambda], [1], [10]\}$
  - δ:

Q_L	0	1
[Λ]	[Λ]	[1]
[1]	[10]	[1]
[10]	[Λ]	[1]



Myhill and Nerode Theorem

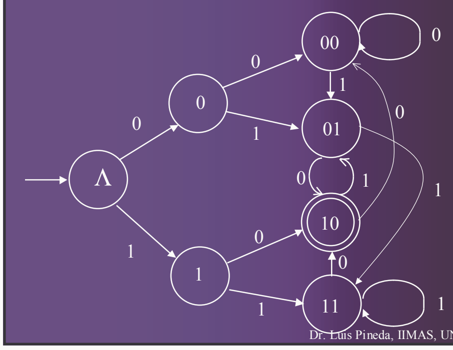
- L is a regular language if and only if the set of equivalence classes is finite
- Regularity is a property of a language, not a property of the languages we use to study languages (*RE*) or the FA that accepts the language!
- We are interested in properties of the objects of study… more than in the properties of the physical or abstract devices we use to study these objects!

A related problem

- Finding I_L can be very difficult!
- However, if we have a FA already, can we find its equivalent minimal automata?
- This is more practical for real applications!

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

Is there an algorithm for finding the minimal FA?



$\delta(ab,c) = bc$

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

A second way to partition the set of strings!

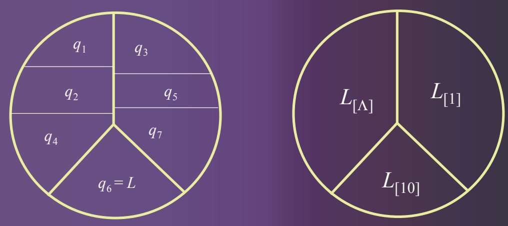
- The language consisting of the sets of strings that reach a state from the initial state:

$$L_q = \{x \in \Sigma^* \mid \delta^*(q_0, x) = q\}$$

- The set of all strings is the union of these sets for all $q \in Q$!
- The states of a FA offer a second partition on the set of strings

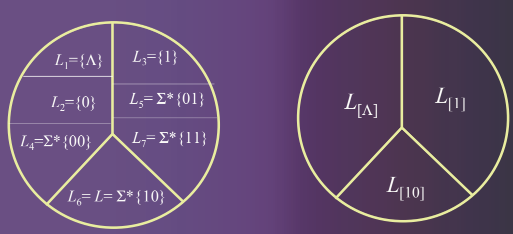
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

$L_q = \{x \in \Sigma^* \mid \delta^*(q_0, x) = q\}$



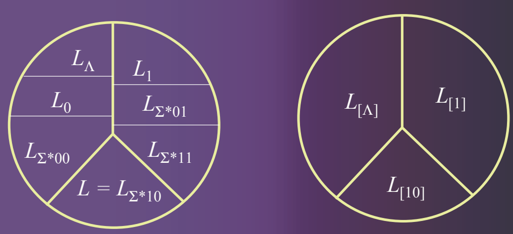
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

$L_q = \{x \in \Sigma^* \mid \delta^*(q_0, x) = q\}$



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

$L_q = \{x \in \Sigma^* \mid \delta^*(q_0, x) = q\}$



A number of classes determined by the states of a FA will correspond to a single partition in I_L

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

The reduction

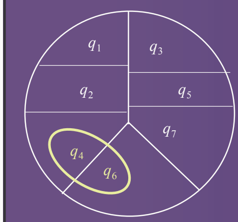
- Let (p, q) be pairs of states in the same equivalent class
- The languages L_p and L_q are subsets of the same equivalent class
  - $p \equiv q$
- On the other hand, if p and q do not belong to the same partition, the corresponding languages are subsets of different equivalent classes
  - $p \not\equiv q$
- The minimal FA can be found identifying al (p, q) such that $p \not\equiv q$

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

The property we need…

S is the set of pairs (p, q) such that $p \not\equiv q$

Accepting states and all non-accepting are in different partitions:

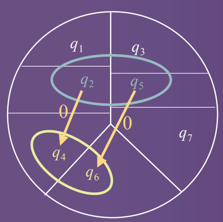


$(q_4, q_6) \in S$

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

The property we need…

S is the set of pairs (p, q) such that $p \not\equiv q$



$(q_4, q_6) \in S$

and

(q_2, q_5) is such that

$\delta(q_2, 0) = q_4$

$\delta(q_5, 0) = q_6$

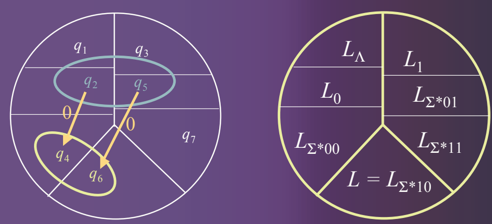
then $(q_2, q_5) \in S$.

0 is a distinguishing string for states q_2 and q_5

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

The property we need…

- 0 is a distinguishing string taken strings in $L_{\Sigma*01}$ into L but not strings in L_0 into L
- Subsequently, q_2 and q_5 belong to different partitions of I_L



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

The property we need…

- If (p, q) is a pair of states such that $p \not\equiv q$ and $\delta(r, a) = p$ and $\delta(s, a) = q$, then (r, s) is also a pair of states such that $r \not\equiv s$
- If $p \not\equiv q$ then there is a z such that
  - $\delta^*(p, z) \in A$ or $\delta^*(q, z) \in A$, but not both
- So, only one of
  - $\delta^*(r, az) = \delta^*(\delta(r, a), z) = \delta^*(p, z)$
  - $\delta^*(s, az) = \delta^*(\delta(s, a), z) = \delta^*(q, z)$

  will be in A!
- Paths of non-equivalent states are formed by non-equivalent states!

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

The algorithm…

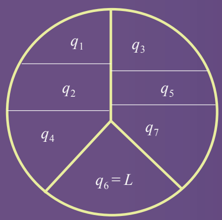
- Let S be the set of pairs (p, q) such that $p \not\equiv q$
  - For any p and q such that only one of them is in A, include (p, q) in S
  - For any pair $(p, q) \in S$, if (r, s) is a pair for which $\delta(r, a) = p$ and $\delta(s, a) = q$ then $(r, s) \in S$
  - No other pairs are in S

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

Compute *S*

S is the set of pairs (p, q) such that $p \not\equiv q$

1. For any p and q such that only one of them is in A, include (p, q) in S
2. So far, we only know that the accepting state is not in the same class of non-accepting states



$S = \{(q_1, q_6), (q_2, q_6), (q_3, q_6), (q_4, q_6), (q_5, q_6), (q_7, q_6)\}$

Compute *S*

S is the set of pairs (p, q) such that $p \not\equiv q$

2. For any pair $(p, q) \in S$, if (r, s) is such that
   $\delta(r, a) = p$
   $\delta(s, a) = q$
   then $(r, s) \in S$.

- Note that *S* is also a relation (the one of states in different partitions)
- The complement of this relation, will be also a relation, but the one of the states in the same partition!

Computation of *S*

q_7						1	
q_6						0	
q_5						1	
q_4						1	
q_3						1	
q_2						1	
q_1						1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

$S = \{(q_6, q_1), (q_6, q_2), (q_6, q_3), (q_6, q_4), (q_6, q_5), (q_6, q_7)\}$

Computation of *S*

q_7						1	
q_6	1	1	1	1	1	0	1
q_5						1	
q_4						1	
q_3						1	
q_2						1	
q_1						1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

S is symmetrical!

Computation of *S*

q_7						1	0
q_6	1	1	1	1	1	0	1
q_5					0	1	
q_4				0		1	
q_3			0			1	
q_2		0				1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

We also know that no state is related to itself in *S*

Computation of *S*

q_7						1	0
q_6	1	1	1	1	1	0	1
q_5					0	1	
q_4				0		1	
q_3			0			1	
q_2		0				1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

Finding the pairs (r, s)

If (r, s) is such that $\delta(r, a) = p$ and $\delta(s, a) = q$ then $(r, s) \in S$.

Computation of S

q_7						1	0
q_6	1	1	1	1	1	0	1
q_5					0	1	
q_4				0		1	
q_3			0			1	
q_2		0				1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

Q	0	1
q_1	q_2	q_1
q_2	q_4	q_5
q_3	q_6	q_7
q_4	q_4	q_5
q_5	q_6	q_7
q_6	q_4	q_5
q_7	q_6	q_7

To compute (r, s) we need to look at δ going backwards

Computation of S: going backwards

q_7						1	0
q_6	1	1	1	1	1	0	1
q_5					0	1	
q_4				0		1	
q_3			0			1	
q_2		0				1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

Q	0	1
q_1	q_2	q_1
q_2	q_4	q_5
q_3	q_6	q_7
q_4	q_4	q_5
q_5	q_6	q_7
q_6	q_4	q_5
q_7	q_6	q_7

If (r, s) is such that $\delta(r, a) = q_4$ and $\delta(s, a) = q_6$

then $(r, s) \in S$.

Computation of S: going backwards

q_7						1	0
q_6	1	1	1	1	1	0	1
q_5		1			0	1	
q_4				0		1	
q_3			0			1	
q_2		0				1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

Q	0	1
q_1	q_2	q_1
q_2	q_4	q_5
q_3	q_6	q_7
q_4	q_4	q_5
q_5	q_6	q_7
q_6	q_4	q_5
q_7	q_6	q_7

(q_2, q_5) is one such pair:

$\delta(q_2, 0) = q_4$ and $\delta(q_5, 0) = q_6$ then $(q_2, q_5) \in S$.

Computation of S: going backwards

q_7						1	0
q_6	1	1	1	1	1	0	1
q_5		1			0	1	
q_4				0		1	
q_3			0			1	
q_2		0			1	1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

S is symmetrical: $(q_5, q_2) \in S$.

Computation of S: going backwards

q_7						1	0
q_6	1	1	1	1	1	0	1
q_5		1			0	1	
q_4				0		1	
q_3			0			1	
q_2		0				1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

S is symmetrical: We only need to find the values for the relation in one way

Computation of S

q_7	1	1		1		1	0
q_6	1	1	1	1	1	0	1
q_5	1	1		1	0	1	
q_4			1	0		1	
q_3	1	1	0			1	
q_2		0				1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

Q	0	1
q_1	q_2	q_1
q_2	q_4	q_5
q_3	q_6	q_7
q_4	q_4	q_5
q_5	q_6	q_7
q_6	q_4	q_5
q_7	q_6	q_7

The pairs found by searching backwards!

Computation of S

q_7	1	1		1		1	0
q_6	1	1	1	1	1	0	1
q_5	1	1		1	0	1	
q_4			1	0	1	1	1
q_3	1	1	0	1		1	
q_2		0	1		1	1	1
q_1	0		1		1	1	1
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

S is symmetrical!

Computation of S

q_7	1	1	0	1	0	1	0
q_6	1	1	1	1	1	0	1
q_5	1	1	0	1	0	1	0
q_4	0	0	1	0	1	1	1
q_3	1	1	0	1	0	1	0
q_2	0	0	1	0	1	1	1
q_1	0	0	1	0	1	1	1
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

The values for all the matrix of states in different classes

The relation of related states:

q_7			1		1		1
q_6						1	
q_5			1		1		1
q_4	1	1		1			
q_3			1		1		1
q_2	1	1		1			
q_1	1	1		1			
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

The complement: the relation between the pairs in the **same** partition**!**

Computation of Q_L

q_7			1		1		1
q_6						1	
q_5			1		1		1
q_4	1	1		1			
q_3			1		1		1
q_2	1	1		1			
q_1	1	1		1			
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

The relation is symmetrical

Computation of Q_L

q_7			1		1		1
q_6						1	
q_5			1		1		1
q_4	1	1		1			
q_3			1		1		1
q_2	1	1		1			
q_1	1	1		1			
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

The relation is reflexive

Computation of Q_L

q_7			1		1		1
q_6						1	
q_5			1		1		1
q_4	1	1		1			
q_3			1		1		1
q_2	1	1		1			
q_1	1	1		1			
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

The relation is transitive: States q_1, q_2 and q_4 are mutually related, but not with any one else

Computation of Q_L

q_7			1		1		1
q_6						1	
q_5			1		1		1
q_4	1	1		1			
q_3			1		1		1
q_2	1	1		1			
q_1	1	1		1			
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

The relation is transitive: States q_3, q_5 and q_7 are mutually related, but not with any one else

Computation of Q_L

q_7			1		1		1
q_6						1	
q_5			1		1		1
q_4	1	1		1			
q_3			1		1		1
q_2	1	1		1			
q_1	1	1		1			
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

The relation is transitive: State q_6 is related with itself but not with any one else!

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

Computation of Q_L

q_7					1	1	1
q_5					1	1	1
q_3					1	1	1
q_6				1			
q_4	1	1	1				
q_2	1	1	1				
q_1	1	1	1				
	q_1	q_2	q_4	q_6	q_3	q_5	q_7

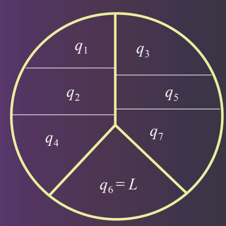
Reordering the set of states!

The relation is transitive: the classes form clusters where the properties of the reflexive relation can be seen directly!

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

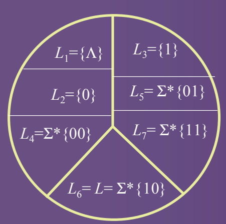
The set of equivalent states!

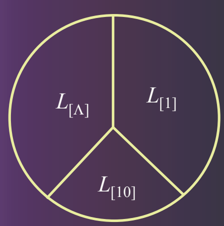
q_7					1	1	1
q_5					1	1	1
q_3					1	1	1
q_6				1			
q_4	1	1	1				
q_2	1	1	1				
q_1	1	1	1				
	q_1	q_2	q_4	q_6	q_3	q_5	q_7



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

The minimal set of states: Q_L





The reduction is complete!

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

Computation of S

q_7	1	1		1		1	0
q_6	1	1	1	1	1	0	1
q_5	1	1		1	0	1	
q_4			1	0		1	
q_3	1	1	0			1	
q_2		0				1	
q_1	0					1	
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

Mark the pairs found by searching backwards in one or more passes

The pairs found by searching backwards in the upper left part of the matrix!

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

The algorithm

q_7	1	1		1		1	
q_6	1	1	1	1	1		
q_5	1	1		1			
q_4			1				
q_3	1	1					
q_2							
q_1							
	q_1	q_2	q_3	q_4	q_5	q_6	q_7

- The pairs that are not marked belong to equivalent classes
- Add the pairs relating the accepting states with themselves

[Λ] = {(1, 2), (1,4), (2,4)}
[1] = {(3, 5), (3, 7), (5, 7)}
[10] = {(6, 6)} (The reference state)

The algorithm

1. Include in S all pairs containing only one accepting state
2. Select region of matrix: if $(x, y) \in S$ and $y < x$ add (y, x) in S (as S is symmetrical)
3. Define Q as the set of pairs of the upper left part of the matrix that are not in S
4. For each $(x, y) \in Q$, check whether it can reach a pair in S in one transition
   – If it does, remove (x, y) from Q and include it in S
5. Repeat 4 until no more pairs can be added to S
6. Add to Q all pairs (x, x), where x is an accepting state
7. Q is the equivalence relation sought

Example

- Find the minimal automata M_L for the FA M:

M

Q	a	b
→1	2	1
*2	4	1
3	2	5
4	6	3
*5	4	5
6	5	2

S

6		1			1	0
5	1	0	1	1	0	1
4		1		0	1	
3		1	0		1	
2	1	0	1	1	0	1
1	0	1			1	
Q	1	2	3	4	5	6

- S is the relation of between states of different partitions
- Initial assumption for S: (p, q) such that *exactly* one element is in A

Example … cont

(r, s)	$\delta(r, a)$	$\delta(s, a)$	(p, q)	Pass
(4, 6)	6	5	(6, 5)	1
(3, 6)	2	5	(2, 5)	
(3, 4)	2	6	(2, 6)	1
(1, 6)	2	5	(2, 5)	
(1, 4)	2	6	(2, 6)	1
(1, 3)	2	2	(2, 2)	

S

6	0	1	0	1	1	0
5	1	0	1	1	0	1
4	1	1	1	0	1	1
3	0	1	0	1	1	0
2	1	0	1	1	0	1
1	0	1	0	1	1	0
Q	1	2	3	4	5	6

- The related states are: [1] = {1, 3, 6}, [4] and [2] = {2, 5}

Example … cont

S

6	0	1	0	1	1	0
5	1	0	1	1	0	1
4	1	1	1	0	1	1
3	0	1	0	1	1	0
2	1	0	1	1	0	1
1	0	1	0	1	1	0
Q	1	2	3	4	5	6

E

6	1		1			1
5		1			1	
4				1		
3	1		1			1
2		1			1	
1	1		1			1
Q	1	2	3	4	5	6

E

5					1	1
2					1	1
4				1		
6	1	1	1			
3	1	1	1			
1	1	1	1			
Q	1	3	6	4	2	5

- The equivalent relation: [1] = {1, 3, 6}, [2] = {2, 5} and [4]

Example … cont

- $M_L = (Q_L, \{0, 1\}, [136], \{[25]\}, \delta)$ where
- Q = {[136], [25], [4]},
- δ:
  – $\delta([136], a) = [25]$ and $\delta([136], b) = [15]$?
  – $\delta([25], a) = [4]$ and $\delta([25], b) = [1] = [15]$?
  – $\delta([4], a) = [6]$ and $\delta([4], b) = [3] = [136]$
- The string a is not distinguishing if the final states are equivalent!

Q	a	b
→1	2	1
*2	4	1
3	2	5
4	6	3
*5	4	5
6	5	2

Example … cont

- Assume accepting states are not related!

M

Q	a	b
→1	2	1
*2	4	1
3	2	5
4	6	3
*5	4	5
6	5	2

S

6		1			1	0
5	1	1	1	1	0	1
4		1		0	1	
3		1	0		1	
2	1	0	1	1	1	1
1	0	1			1	
Q	1	2	3	4	5	6

- *S* is the relation of between states of different partitions

Example … cont

(r, s)	$\delta(r, a)$	$\delta(s, a)$	(p, q)	Pass
(4, 6)	6	5	(6, 5)	1
(3, 6)	2	5	(2, 5)	
(3, 4)	2	6	(2, 6)	1
(1, 6)	2	5	(2, 5)	
(1, 4)	2	6	(2, 6)	1
(1, 3)	2	2	(2, 2)	

S

6	1	1	1	1	1	0
5	1	1	1	1	0	1
4	1	1	1	0	1	1
3	0	1	0	1	1	0
2	1	0	1	1	1	1
1	0	1	0	1	1	0
Q	1	2	3	4	5	6

- The related states are: [1] = {1, 3}, [2], [4], [5] and [6]

Example … cont

S

6	1	1	1	1	1	0
5	1	1	1	1	0	1
4	1	1	1	0	1	1
3	0	1	0	1	1	1
2	1	0	1	1	1	1
1	0	1	0	1	1	1
Q	1	2	3	4	5	6

E

6						1
5					1	
4				1		
3	1		1			
2		1				
1	1		1			
Q	1	2	3	4	5	6

E

6						1
5					1	
4				1		
2			1			
3	1	1				
1	1	1				
Q	1	3	2	4	5	6

- The equivalent relation: [1] = {1, 3}, [2], [4], [5] and [6]

Example … cont

- $M_L = (Q_L, \{0, 1\}, [136], \{[25]\}, \delta)$ where
- Q = {[13], [2], [4], [5], [6]}
- δ:
  - δ([13], *a*) = [2] and δ([13], *b*) = [15]?
  - δ([25], *a*) = [4] and δ([25], *b*) = [1] = [15]?
  - δ([4], *a*) = [6] and δ([4], *b*) = [3] = [136]
- 1 and 5 are in different partitions: *a* is not distinguishing!

Q	a	b
→1	2	1
*2	4	1
3	2	5
4	6	3
*5	4	5
6	5	2

Example … cont

- We take *b* to be the distinguishing string, and the accepting states to be related

(r, s)	$\delta(r, b)$	$\delta(s, b)$	(p, q)	Pass
(4, 6)	3	2	(3, 2)	1
(3, 6)	5	2	(5, 2)	
(3, 4)	5	3	(5, 3)	1
(1, 6)	1	2	(1, 2)	1
(1, 4)	1	3	(1, 3)	2
(1, 3)	1	5	(1, 5)	1

S

6	1	1	0	1	1	0
5	1	0	1	1	0	1
4	1	1	1	0	1	1
3	1	1	0	1	1	0
2	1	0	1	1	0	1
1	0	1	0	1	1	1
Q	1	2	3	4	5	6

- The related states are: [1], [3] = {3, 6}, [4] and [2] = {2, 5}

Example … cont

- $M_L = (Q_L, \{0, 1\}, [1], \{[25]\}, \delta)$ where
- Q = {[1], [25], [36], [4]},
- δ:
  - δ([1], *a*) = [2] = [25] and δ([1], *b*) = [1]
  - δ([25], *a*) = [4] and δ([25], *b*) = [1] = [15] ?
  - δ([4], *a*) = [6] and δ([4], *b*) = [3] = [136]
- 1 and 5 belong to different partitions, so the accepting states cannot be related

Q	a	b
→1	2	1
*2	4	1
3	2	5
4	6	3
*5	4	5
6	5	2

Example … cont

- We take b to be the distinguishing string, and the accepting states NOT to be related

(r, s)	$\delta(r, b)$	$\delta(s, b)$	(p, q)	Pass
(4, 6)	3	2	(3, 2)	1
(3, 6)	5	2	(5, 2)	1
(3, 4)	5	3	(5, 3)	1
(1, 6)	1	2	(1, 2)	1
(1, 4)	1	3	(1, 3)	2
(1, 3)	1	5	(1, 5)	1

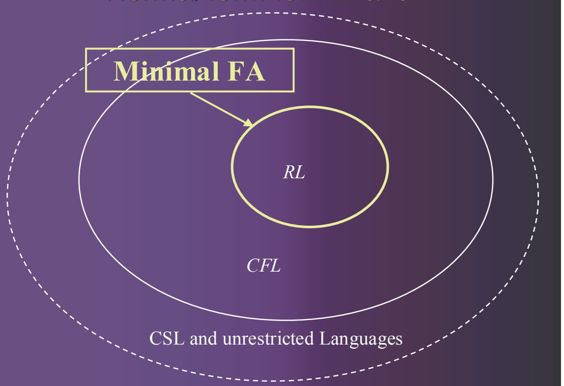
S						
6	1	1	1	1	1	0
5	1	1	1	1	0	1
4	1	1	1	0	1	
3	1	1	0		1	
2	1	0	1	1	1	1
1	0	1			1	
Q	1	2	3	4	5	6

- There are no equivalent states and the FA is already minimal

Minimal Automata

- For every FA there is always a minimal FA with n states
- This is useful for simplification of FA
- Minimal FA also provide a "normal form" for FA: all FA that reduce to the minimal one are equivalent
- This also provides a test for equivalence between FA!
- Using Kleen's Theorem we can find the "normal form" for a *RE*

Normal form for *RE* and FA



The power of *RE* and FA

- A *problem* is the question of deciding whether a string is a member of some particular language
- FA solve simple decision problems (i.e. whether x is an even number, but not whether x is a palindrome).
- Generic decision problem for a FA: given a strings x and a FA M decide whether x is accepted by M:

Decision problems for *RL*

- Given a *RE* r and a string x, $x \subseteq L(r)$?
- Given a FA M, is $L(M) = \Phi$?
- Given a FA M, is $L(M)$ finite?
- Given M_1 and M_2, is $L(M_1) \cap L(M_2) \neq \Phi$?
- Given M_1 and M_2, is $L(M_1) = L(M_2)$?
- Given M_1 and M_2, is $L(M_1) \subseteq L(M_2)$?
- Given two *RE*s r_1 and r_2, is $L(r_1) = L(r_2)$?
- Given a FA M is it minimal?

RL is a very solid system for simple problems!