Session 5

Finite Automata

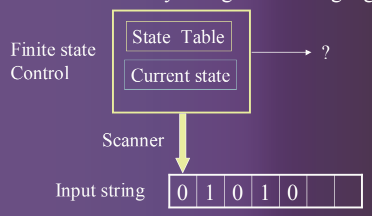
A Machine for interpreting *RE*

- $L = \{0, 1\}^*\{0\}$
  - The language of strings ending in 0 over $\Sigma = \{a, b\}$

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

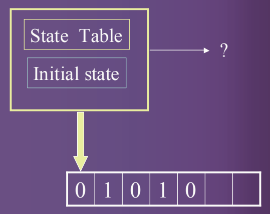
A Machine for interpreting *RE*

- $L = \{0, 1\}^*\{0\}$
  - The language of strings ending in 0 over $\Sigma = \{0, 1\}$
- A machine to identify strings in the language



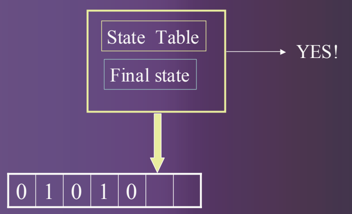
Dr. Luis Pineda, IIMAS, UNAM & OSU-CIS, 2003

A Finite Automata (FA)



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

A Finite Automata (FA)



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

How many states do we need?

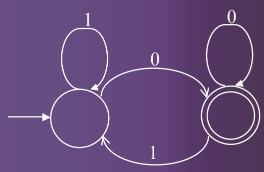
- We assume:
  - Only one pass through the input string (from left to right)
  - A tentative decision of whether the string is in the language is made after scanning each symbol!
- How much do we have to remember to make the right decision at the end of the string?
  - Remembering every thing?
  - Remembering nothing?
    - If the language is empty decide always NO!
    - If the language is Σ^* decide always Yes!
- But what if we have to distinguish between two strings, say x and y
  - We need to remember something!

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

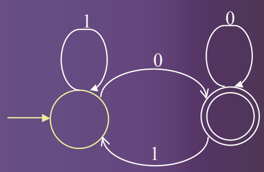
Example 1: Strings ending with 0

- $L = \{0, 1\}^*\{0\}$
  - $\Lambda \notin L$
  - Whether a string is in L depends only on the last symbol
  - Also, we can think in partitioning L^* in two sets of strings: those ending with “1” and those ending with “0”
  - For our purpose, any string in either subset is equivalent!
  - At every state we only need to distinguish the symbol that is currently being scanned!

A machine to do this job

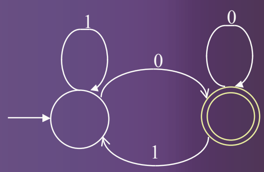


Initial state

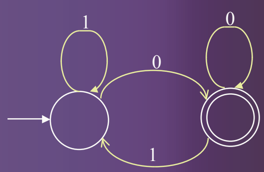


- Where one gets with the Λ string

Accepting state

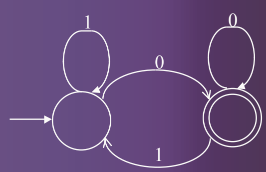


Transitions



- If the machine is an state, and the symbol labeling an arc is scanned on the string, the machine moves to the state at the end of the arc

A machine to do this job



- There are two states: one for each thing to remember
- The first remembers that the string is ending in 1
- The second, that it is ending in 0, and is accepting
- There is a state for each class of equivalent strings!

Example 2: Strings ending with 0X

- L is $\{0, 1\}^*$ and the next to last symbol is 0
  - $\Lambda \notin L$
  - First hypothesis: two classes
    - Strings ending in 00 and 01 (belong to the language?)
    - Strings ending in 10 and 11
  - But what happens when the next symbol is read?
    - …00 becomes 000 or 001
    - …01 becomes 010 or 011

      } Do not belong to the same class
    - …10 becomes 100 or 101
    - …11 becomes 110 or 111

      } Do not belong to the same class

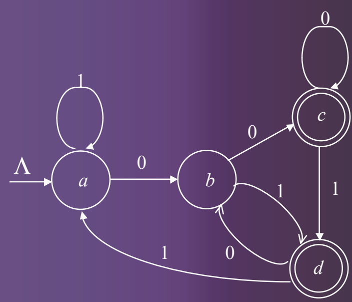
Example 2: Strings ending with 0X

- We do need four classes for strings of length 2
  - Strings ending in 00
  - Strings ending in 01 (accepted if it is the end of the string!)
  - Strings ending in 10
  - Strings ending in 11
- For strings of length less than two:
  - Λ and 1 can be grouped with 11, because at least two next symbols are required to make such a string into the language
  - 0 is in the same case as 10: neither is in the language, but it can be once the next symbol is read, unless it is the end of the string!

Example 2: Strings ending with 0X

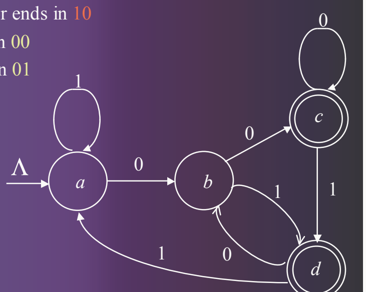
- So, we do need four classes:
  - Class *a*: The string is Λ and 1 or ends in 11
  - Class *b*: The string is 0 or ends in 10
  - Class c: Strings ending in 00
  - Class *d*: Strings ending in 01
- To identify a language, we need a FA with four states: one for each of these classes

A machine for strings ending with 0X



Example 2: Strings ending with 0X

- Class *a*: The string is Λ and 1 or ends in 11
- Class *b*: The string is 0 or ends in 10
- Class c: Strings ending in 00
- Class *d*: Strings ending in 01



Example 3: Strings ending with 11

- $L = \{0, 1\}^*\{11\}$
  - First hypothesis: four classes for strings of length 2
    - …00
    - …01
    - …10
    - …11
  - However 00 and 10 do not need to be distinguished!
    - …00 becomes 000 or 001
    - …10 becomes 100 or 101
    - Neither is in the language, and …00 and …10 belong to the same class!

Example 3: Strings ending with 11

- Three classes
  - …00, …10
  - …01
  - …11
- Also, the string 1 can be identified with 01
  - …01 becomes 010 or 011
  - 1 becomes 10 or 11

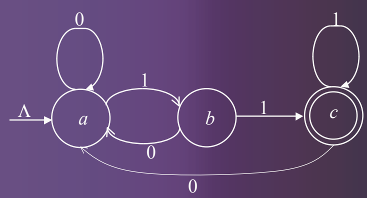
Example 3: Strings ending with 11

- Three classes
  - …00, …10
  - …01, 1
  - …11
- Also, Λ and 0 can be identified with all strings ending in 0: these require to read two additional symbols to be a part of the language

Example 3: Strings ending with 11

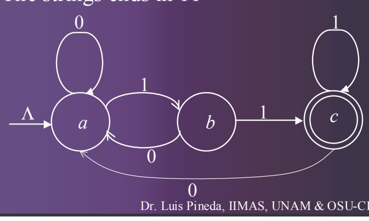
- Three classes
  - …00, …10, Λ and 0
  - …01, 1
  - …11
- Paraphrasing:
  - Class *a*: The string does not end in 1
  - Class *b*: The string is 1 or ends in 01
  - Class *c*: The strings ends in 11

A machine for strings ending with 11



A machine for strings ending with 11

- Paraphrasing:
  - Class *a*: The string does not end in 1
  - Class *b*: The string is 1 or ends in 01
  - Class *c*: The strings ends in 11



Formal definition of FA

- A *finite Automaton*, or *finite state machine* (FA) is a 5-tuple (Q, Σ, q_0, A, δ), where
  - Q is a finite set (of states)
  - Σ is a finite alphabet of *input symbols*
  - $q_0 \in Q$ (the initial state)
  - $A \subseteq Q$ (the set of accepting states)
  - δ is a function from Q x Σ into Q (the transition function)
- For any q of Q and $a \in \Sigma$, $\delta(q,a) = p$, where p is the state to which the FA moves if it is in q when receives the input a

Transition Function

z	p_j	…	p_l
…	…	…	…
a	p_i	…	p_k
Σ / Q	q_0	…	q_n

For any $q \in Q$ and $a \in \Sigma$, $\delta(q,a) = p$

Three notations for FA

- Abstract description
- Transition table
- Transition diagram

Abstract Description

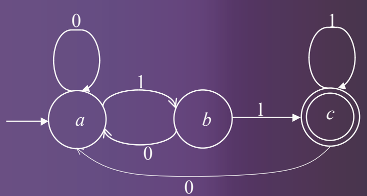
- Five-tuple:
  $M = (Q, \Sigma, q_0, A, \delta)$
- Example:
  $M = (\{a, b, c\}, \{0, 1\}, a, \{c\}, \delta)$
  – Where δ is as follows:
  $\delta(a, 0) = a \quad \delta(b, 0) = a \quad \delta(c, 0) = a$
  $\delta(a, 1) = b \quad \delta(b, 1) = c \quad \delta(c, 1) = c$

Transition Table

Q \ Σ	0	1
→*a*	*a*	*b*
b	*a*	*c*
**c*	*a*	*c*

- → : initial state
- * : member of the set of accepting states

Transition Diagram

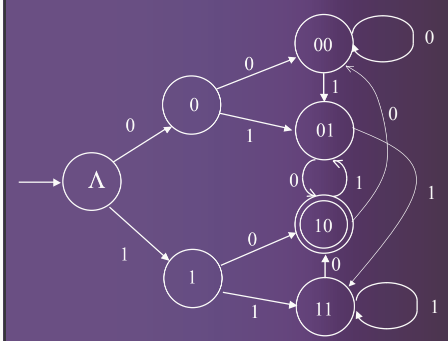


- → : initial state
- ◎ : member of the set of accepting states

Example 4: Strings ending with 10

- $L = \{0, 1\}^*\{10\}$
  – The worst case: seven classes for strings of length 2 or less
  - …00
  - …01
  - …10
  - …11
  - 1
  - 0
  - Λ

FA for recognizing $L = \{0, 1\}^*\{10\}$



$\delta(ab,c) = bc$

Transition table

1	1	01	11	01	11	01	11
0	0	00	(10)	00	(10)	00	10
Σ/Q	Λ	0	1	00	01	(10)	11

◯ The accepting state

Transition table

There are three equivalent states!

1	1	01	11	01	11	01	11
0	0	00	10	00	10	00	10
Σ/Q	Λ	0	1	00	01	10	11

The string 1, and the strings ending in 01 and 11 are in the same class!

Transition table

1	1	01	11	01	11	01	11
0	0	00	10	00	10	00	10
Σ/Q	Λ	0	1	00	01	10	11

- There are three equivalent states!

Transition table

1	1	01	11	01	11	01	11
0	0	00	10	00	10	00	10
Σ/Q	Λ	0	B	00	B	10	B

- Renaming the states 1, 01 and 11 as B

Transition table

1	B	B	B	B	B	B	B
0	0	00	10	00	10	00	10
Σ/Q	Λ	0	B	00	B	10	B

- Updating the new name for 1, 01 and 11 in the table

Transition table

1	B	B	B	B	B
0	0	00	10	00	00
Σ / Q	Λ	0	B	00	10

- Getting rid of the redundant columns

Transition table

1	B	B	B	B	B
0	0	00	10	00	00
Σ / Q	Λ	0	B	00	10

- Now, the states 0, 00 and 10 look the same

Transition table

1	B	B	B	B	B
0	0	00	(10)	00	00
Σ / Q	Λ	0	B	00	(10)

- 0 and 00 are in the same class
- But 10 is not, because it is the accepting state

Transition table

1	B	B	B	B	B
0	A	A	10	A	A
Σ / Q	Λ	A	B	A	(10)

- Renaming 0 and 00 as A

Transition table

1	B	B	B	B
0	A	A	10	A
Σ / Q	Λ	A	B	(10)

- Getting rid of the redundant column

Transition table

1	B	B	B	B
0	A	A	10	A
Σ / Q	Λ	A	B	(10)

- Now, the columns for Λ and A are the same!

Transition table

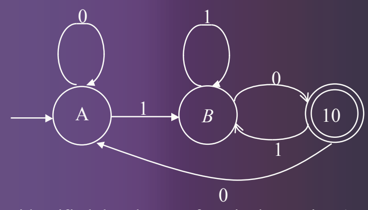
1	B	B	B
0	A	10	A
Σ / *Q*	A	B	(10)

- Getting rid of the column for Λ
- We have a minimal FA to recognize the language $L = \{0, 1\}^*\{10\}$

The transition function δ:

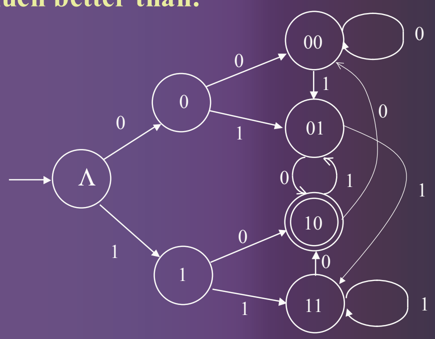
1	B	B	B
0	A	10	A
Σ / *Q*	A	B	(10)

The FA:



And we have also identified the classes of equivalent strings!

Much better than:



The moral!

- FA can be reduced
- All the classes of strings need to be identified
- The minimal FA will have one state for each different class!
- These is a very important property of FA, that will be used for more than one purpose!