NON-DETERMINISTIC FINITE AUTOMATA (NFA)

Dr. A. Beulah

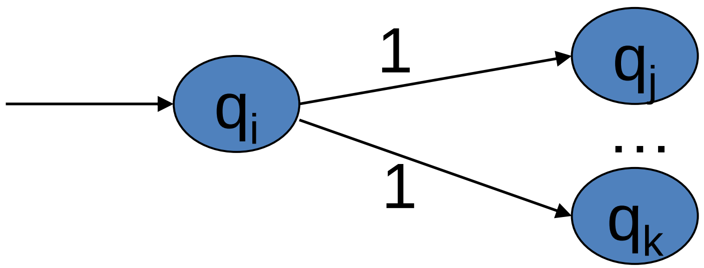
AP/CSE

LEARNING OBJECTIVE

- To construct finite automata for any given pattern and find its equivalent regular expressions
  - To learn the basic concept of NFA
  - Equivalence of DFA and NFA

NON-DETERMINISTIC FINITE AUTOMATA

- A Non-deterministic Finite Automaton (NFA)
  - is of course "non-deterministic"
    - Implying that the machine can exist in more than one state at the same time
    - Transitions could be non-deterministic



- Each transition function therefore maps to a set of states

DFA VS NFA

In a *DFA,*

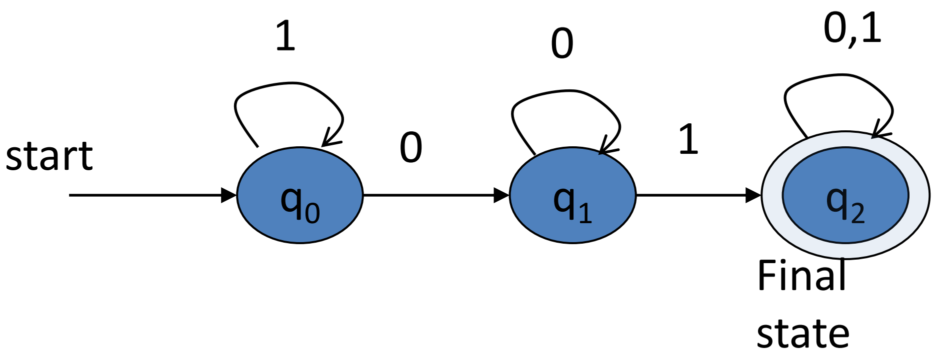
- Each symbol causes a move (eventhough the state of the machine remains unchanged after the move)
- The next state is completely determined by the current state and current symbol.

Where as in a *NFA*

- The machine can move without consuming any symbols and sometimes there is no possible moves and sometimes there are more than one possible moves.
- The state is only partially determined by the current state and input symbol.

EXAMPLE

DFA for strings containing 01



- $\mathcal{Q} = \{q_0,q_1,q_2\}$
- $\sum = \{0,1\}$
- S= q_0
- F = $\{q_2\}$
- Transition table

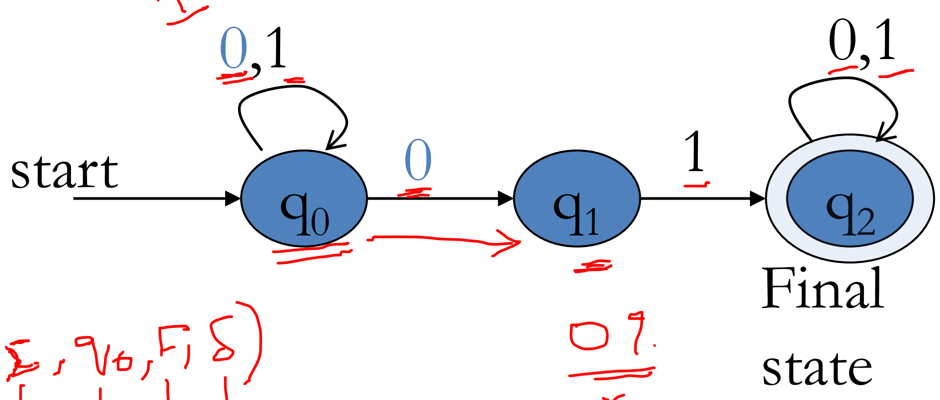
states

δ	0	1
→ q_0	q_1	q_0
q_1	q_1	q_2
*q_2	q_2	q_2

EXAMPLE

NFA for strings containing 01

FA



$DFA = (Q, \Sigma, q_0, F, \delta)$

$\delta \rightarrow Q \times \Sigma \rightarrow Q$

$\delta: Q \times \Sigma \rightarrow 2^Q$

TD

$Q = \{q_0, q_1, q_2\}$

$2^Q = \{\phi, \{q_0\}, \{q_1\}, \{q_2\}, \{q_0, q_1\}, \{q_1, q_2\}, \{q_0, q_2\}, \{q_0, q_1, q_2\}\}$

δ transition

$\delta(q_0, 0) = \{q_0, q_1\}$

$\delta(q_0, 1) = \{q_0\}$

$\delta(q_1, 0) = \phi$

$\delta(q_1, 1) = \{q_2\}$

$\delta(q_2, 0) = \{q_2\}$

$\delta(q_2, 1) = \{q_2\}$

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{0,1\}$
- $S = q_0$
- $F = \{q_2\}$
- Transition table

DFA
NFA

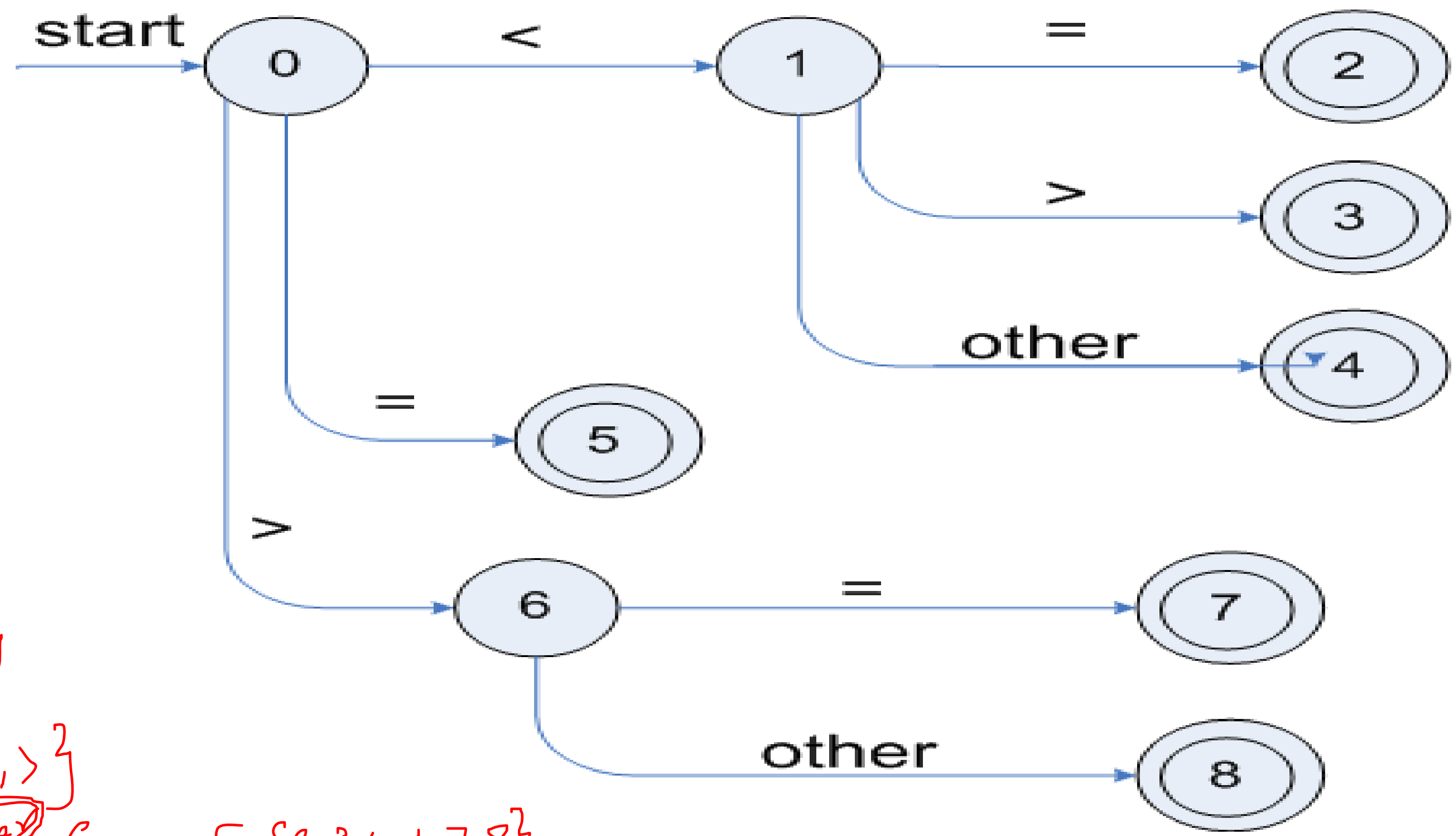
symbols

δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	Φ	$\{q_2\}$
$*q_2$	$\{q_2\}$	$\{q_2\}$

states

EXAMPLE

LEXICAL ANALYZER RECOGNIZING RELATIONAL OPERATORS



<
<=
=
>
>=
<>

$Q = \{0, \ldots 8\}$

$\Sigma = \{<, =, >, \text{other}\}$

$s : 0$ $\quad F = \{2, 3, 4, 5, 7, 8\}$

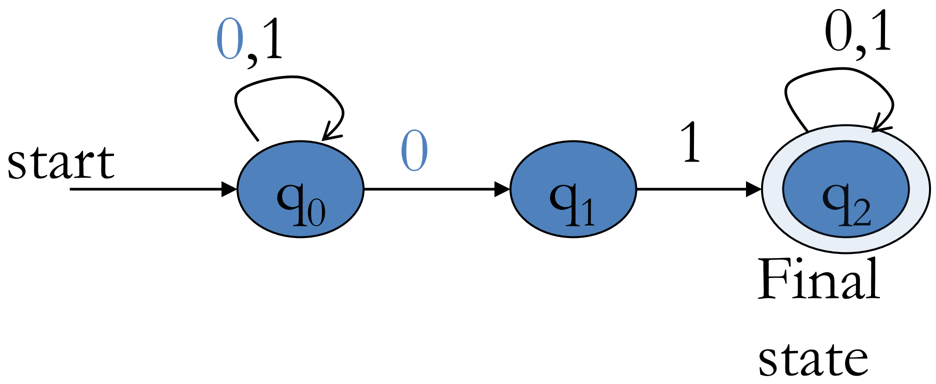
NFA SPECIFICATION

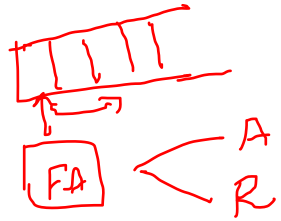
- A Non Deterministic finite automata (NFA) is a 5-tuple $(Q, \Sigma, S, F, \delta)$ where
  - Q is a finite set of states
  - Σ is a set of alphabets   or set of i/p symbols
  - S: $q_0 \in Q$ is the initial state
  - $F \subseteq Q$ is a set of final states
  - $\delta: Q \times \Sigma \rightarrow 2^Q$ is a transition function

    2^Q is power set of Q

$Q \times \Sigma \rightarrow Q$ $2^Q \rightarrow$ power set of Q

EXAMPLE





110

$\{q_0, q_1\} \notin F$

∴ Rejected

- Check whether 1011 is accepted by NFA or not.

$\delta(q_0, 1011) = \delta(q_0, 011)$

$= \delta(\{q_0, q_1\}, 11)$

$= \delta(\{q_0, q_2\}, 1)$

$= \{q_0, q_2\}$

$q_2 \in F$ ∴ Accepted

$\delta(q_0, 1) = \{q_0\}$

$\delta(q_0, 0) = \{q_0, q_1\}$

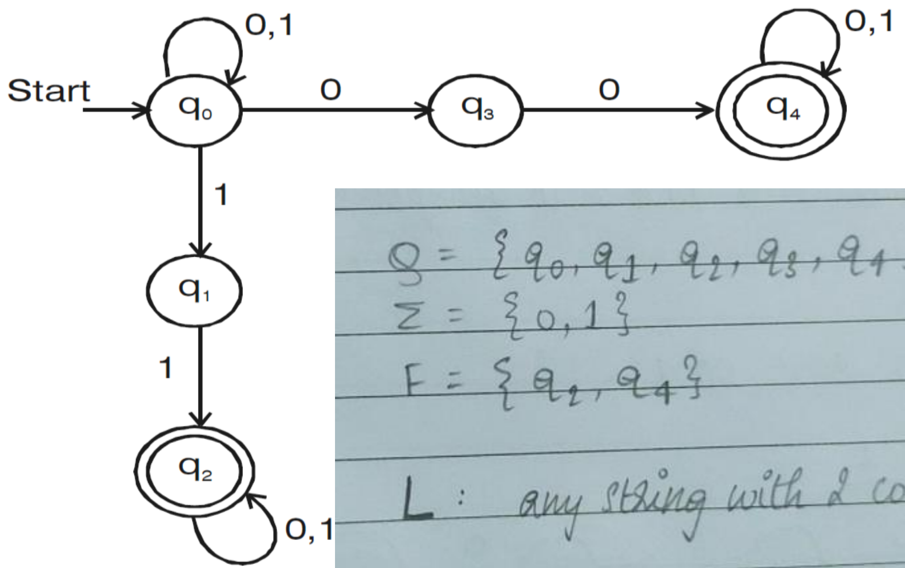
$\delta(\{q_0, q_1\}, 1) = \delta(q_0, 1) \cup \delta(q_1, 1)$

$= \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

$\delta(\{q_0, q_2\}, 1) = \delta(q_0, 1) \cup \delta(q_2, 1)$

$= \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

EXAMPLE



$Q = \{q_0, q_1, q_2, q_3, q_4\}$

$\Sigma = \{0, 1\}$

$F = \{q_2, q_4\}$

L : any string with 2 consecutive 0s/1s

Transition table :

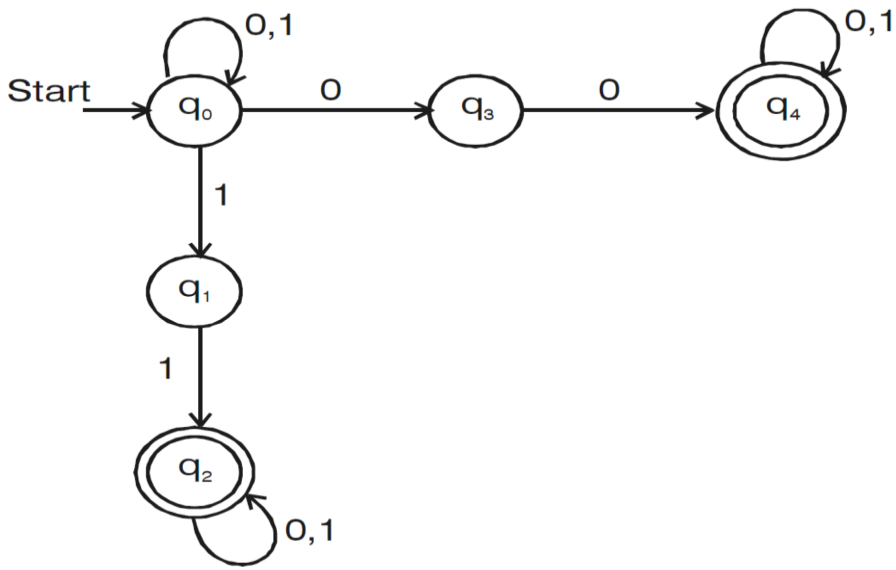
	0	1
q_0	$\{q_0, q_3\}$	$\{q_0, q_1\}$
q_1	ϕ	q_2
q_2	q_2	q_2
q_3	q_4	ϕ
q_4	q_4	q_4

* 1010

$$\begin{aligned}\delta(q_0, 1010) &= \delta(\{q_0, q_1\}, 010) \\ &= \delta(\{q_0, q_3\}, 10) \\ &= \delta(\{q_0, q_1\}, 0) \\ &= \{q_0, q_3\} \notin F\end{aligned}$$ $\therefore$ The string is rejected.

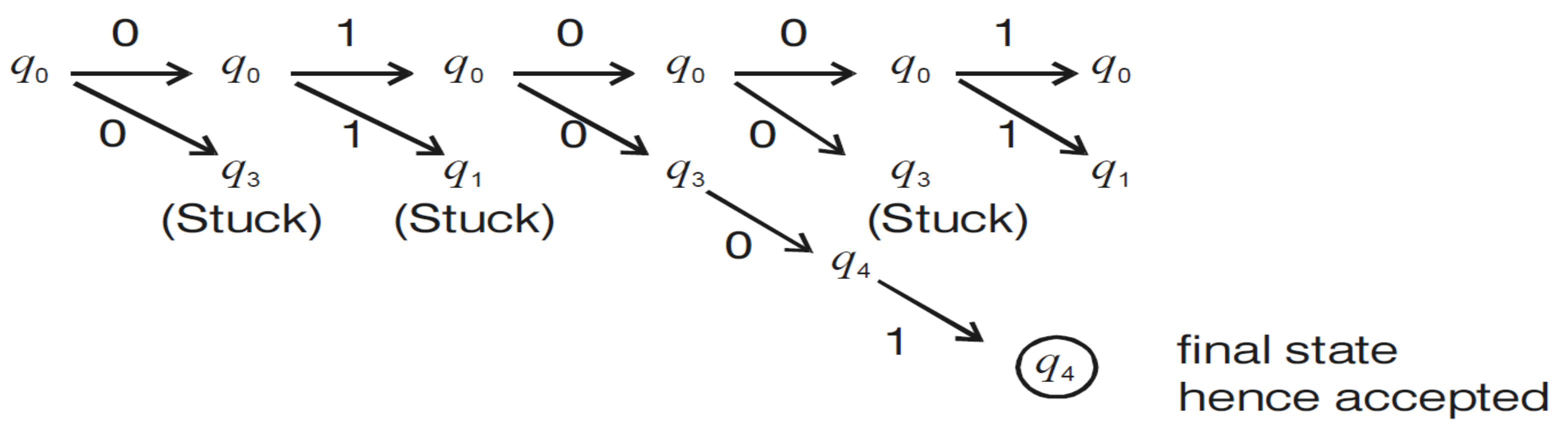
- Q, Σ, F =?
- L=?
- Transition table?
- Check for strings 1010, 1101,0100.

EXAMPLE



States	Inputs	
	0	1
q_0	$\{q_0, q_3\}$	$\{q_0, q_1\}$
q_1	ϕ	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$
q_3	$\{q_4\}$	ϕ
q_4	$\{q_4\}$	$\{q_4\}$

- 01001



EXTENDED TRANSITION FUNCTION ($\hat{\delta}$)

- Basis : $\delta(q, \varepsilon) = \{q\}$

- Induction : $\overline{\delta}(q, wa) = \bigcup_{P \in \overline{\delta}(q,w)} \delta(P,a)$

  for each $w \in \Sigma^*$, $a \in \Sigma$ and $P \in \delta(q, w)$

$\overline{\delta}(q, wa) = \delta(\overline{\delta}(q, w), a)$

LANGUAGE OF A NFA

- Language accepted by NFA is

$$L\ (A) = \{w : \overline{\delta}\ (q_0, w) \cap F \neq \varphi\}$$

$\{\ \} \cap F$

$\phi \rightarrow F$

CONSTRUCT NFA

- Construct an NFA to accept all strings terminating in 01.

L=?

Transition Diagram?

CONSTRUCT NFA

- Construct an NFA that accepts strings which has 3rd symbol b from right over $\Sigma=\{a,b\}$

L=?

Transition Diagram?

DIFFERENCES: DFA VS. NFA

- **DFA**

1. All transitions are deterministic
   - Each transition leads to exactly one state
2. For each state, transition on all possible symbols (alphabet) should be defined
3. Accepts input if the last state is in F
4. Sometimes harder to construct because of the number of states
5. Practical implementation is feasible

- **NFA**

1. Some transitions could be non-deterministic
   - A transition could lead to a subset of states
2. Not all symbol transitions need to be defined explicitly (if undefined will go to a dead state – this is just a design convenience, not to be confused with "non-determinism")
3. Accepts input if *one of* the last states is in F
4. Generally easier than a DFA to construct
5. Practical implementation has to be deterministic (convert to DFA) or in the form of parallelism

SUMMARY

- Definition of Non-deterministic Finite Automata
- Transition diagram, transition function and properties of transition function of NFA
- Equivalence of DFA and NFA

TEST YOUR KNOWLEDGE

- Design a NFA that accepts input string 0’s and 1’s that ends with 11
- Design a NFA over {0,1} to accept strings with 3 consecutive 0’s

LEARNING OUTCOME

On successful completion of this topic, the student will be able to:

- Understand the basic concept of NFA (K3)

REFERENCE

- Hopcroft J.E., Motwani R. and Ullman J.D, “Introduction to Automata Theory, Languages and Computations”, Second Edition, Pearson Education, 2008