CSE-233 : Section A

Summer 2020

Non-Determinism

Reference:

Book2 Chapter 1.2

Md. Saidul Hoque Anik

anik@cse.uiu.ac.bd

Non-Deterministic Finite Automata

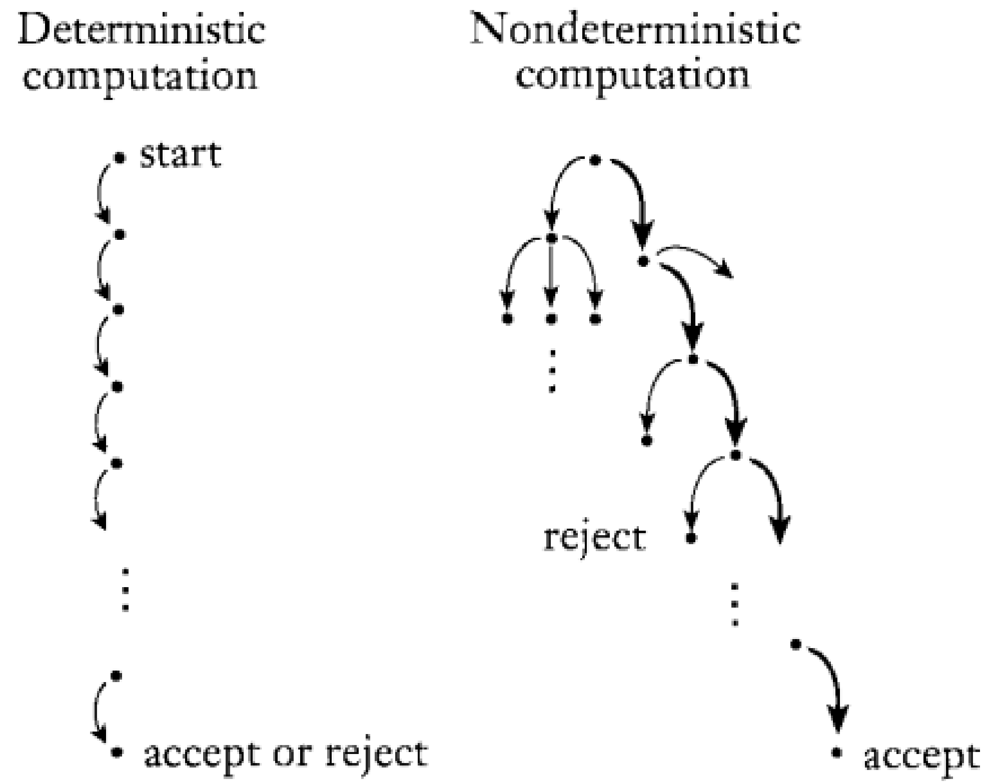


FIGURE 1.28
Deterministic and nondeterministic computations with an accepting branch

NFA Example

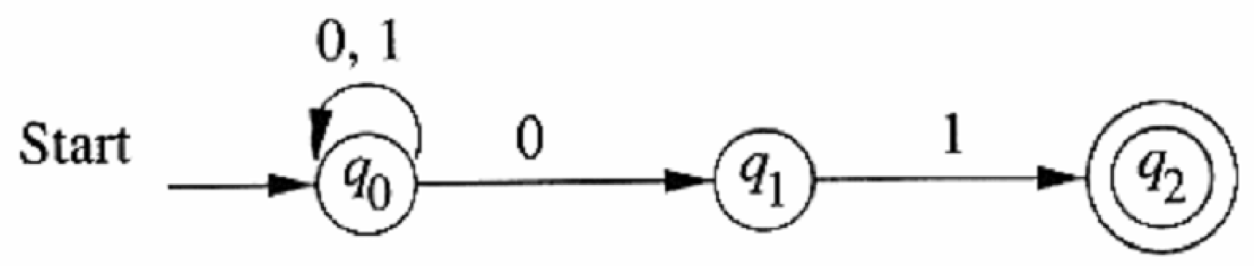


Figure 2.9: An NFA accepting all strings that end in 01

- Each state can have zero, one, or more transitions out labeled by the same symbol
  - Eg, for a single input 1, we can guess that the next state can either be q_0 or q_1
  - If no valid input is given, the thread "dies"

- **What will be the tree for input: 00101?**

NFA Example

For input: 00101

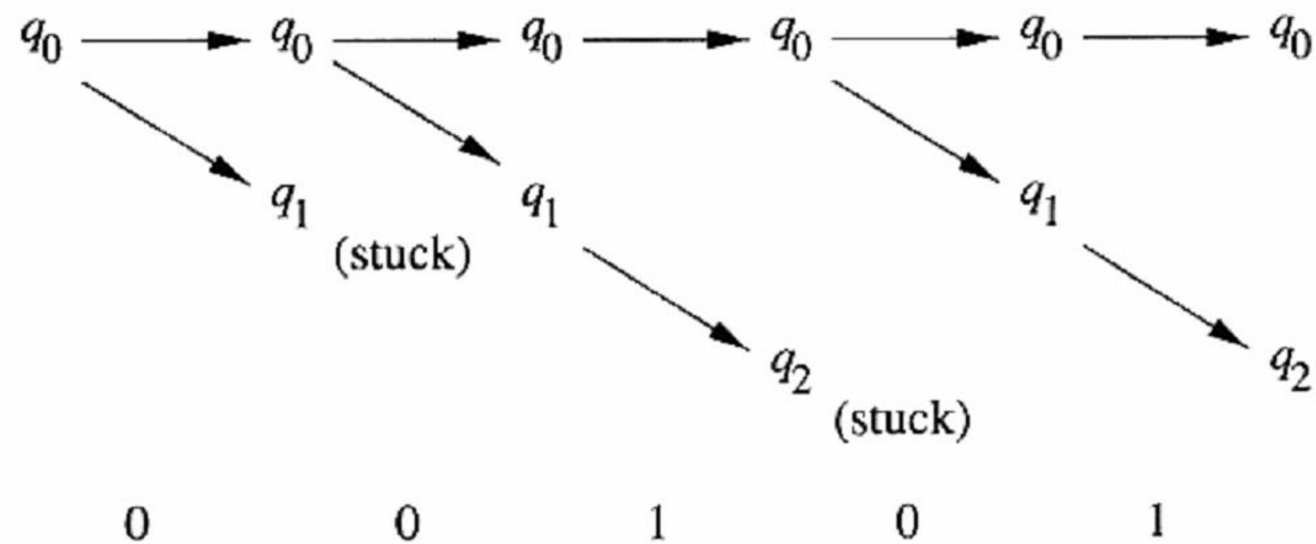


Figure 2.10: The states an NFA is in during the processing of input sequence 00101

NFA Example

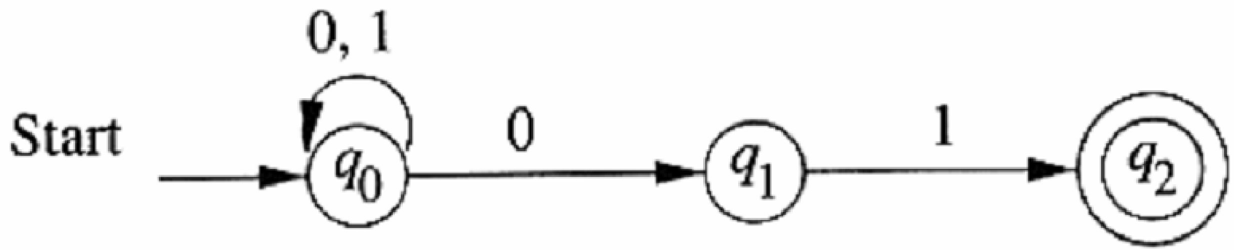


Figure 2.9: An NFA accepting all strings that end in 01

Can you write the transition function of this NFA?

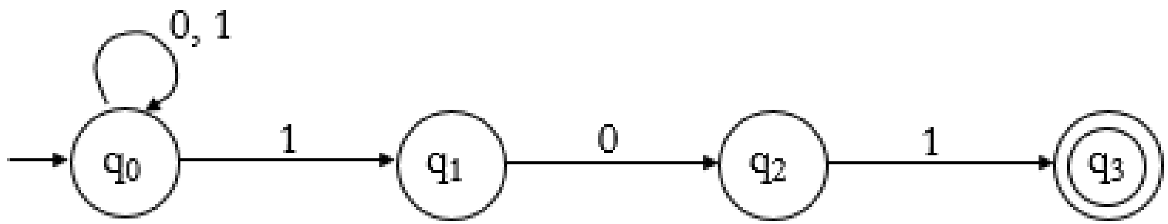
Formal Definition of NFA

A ***nondeterministic finite automaton*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set of states,
2. Σ is a finite alphabet,
3. $\delta\colon Q \times \Sigma_\varepsilon \longrightarrow \mathcal{P}(Q)$ is the transition function,
4. $q_0 \in Q$ is the start state, and
5. $F \subseteq Q$ is the set of accept states.

Example

What does this NFA do?



Alphabet = {0, 1}

start state $Q = \{q_0, q_1, q_2, q_3\}$

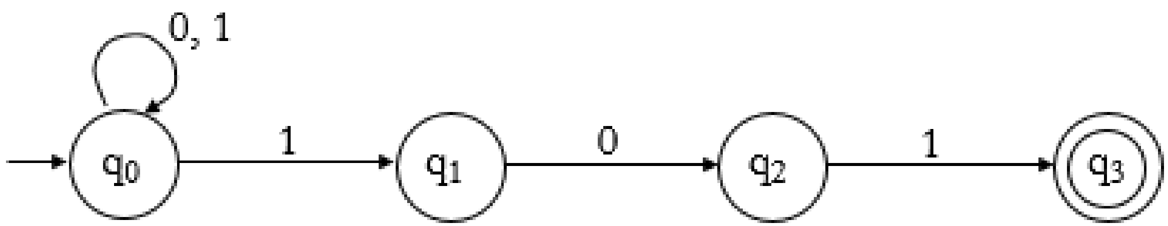
initial state q_0

accepting states $F = \{q_3\}$

Transition Function = ?

Example

What does this NFA do?



Alphabet = {0, 1}

start state *Q* = {q_0, q_1, q_2, q_3}

initial state q_0

accepting states *F* = {q_3}

Transition Function:

states	inputs: 0	inputs: 1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_2\}$	$\varnothing$
q_2	$\varnothing$	$\{q_3\}$
q_3	$\varnothing$	$\varnothing$

NFA vs. DFA

Design a Machine that detected all strings over {0, 1} containing 1 in the third position from the end

NFA vs. DFA

Design a Machine that detected all strings over {0, 1} containing 1 in the third position from the end

Let A be the language consisting of all strings over $\{0,1\}$ containing a 1 in the third position from the end (e.g., 000100 is in A but 0011 is not). The following four-state NFA N_2 recognizes A.

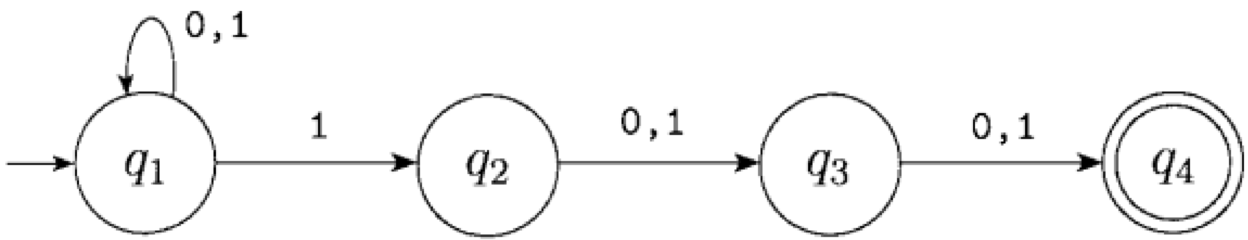
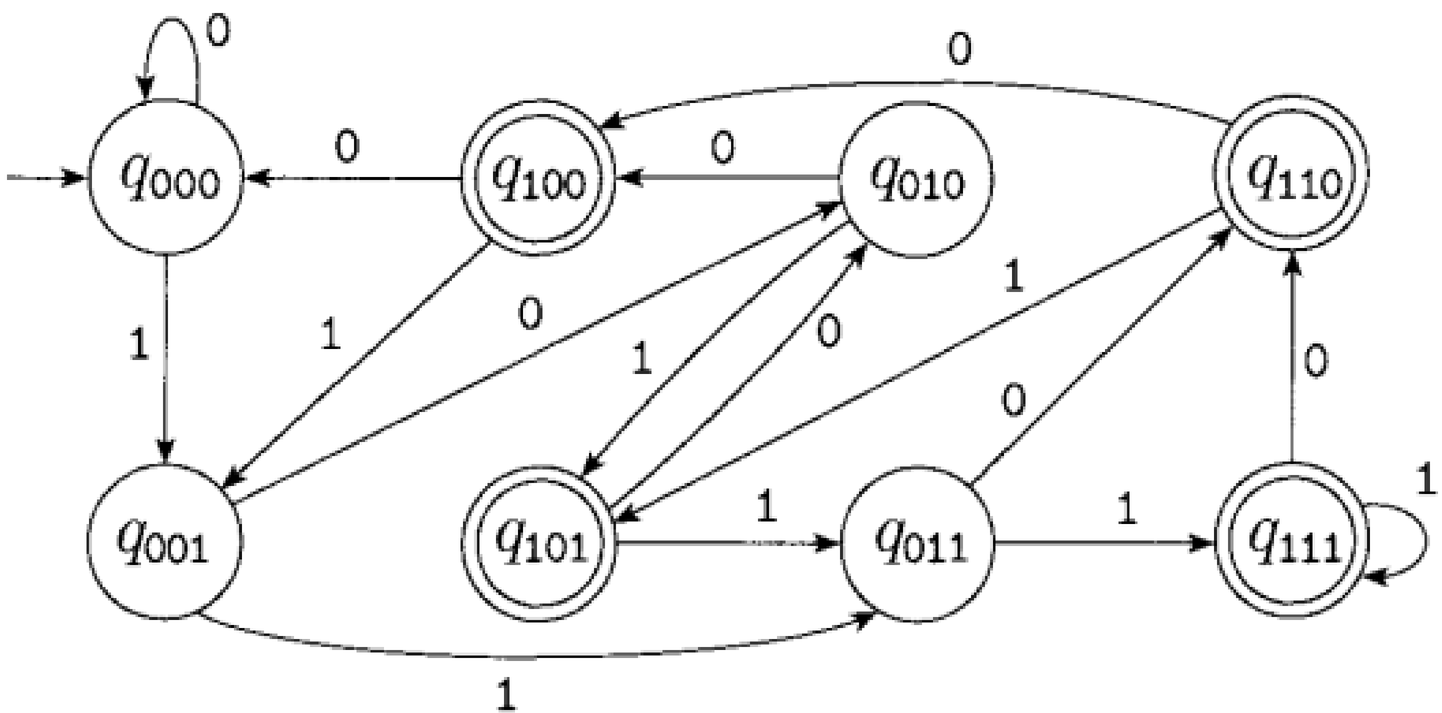


FIGURE 1.31
The NFA N_2 recognizing A

NFA vs. DFA

Design a Machine that detected all strings over {0, 1} containing 1 in the third position from the end



NFA vs. DFA

DFA:

- Faster: follows only one path
- Complex Design: more number of states and transitions

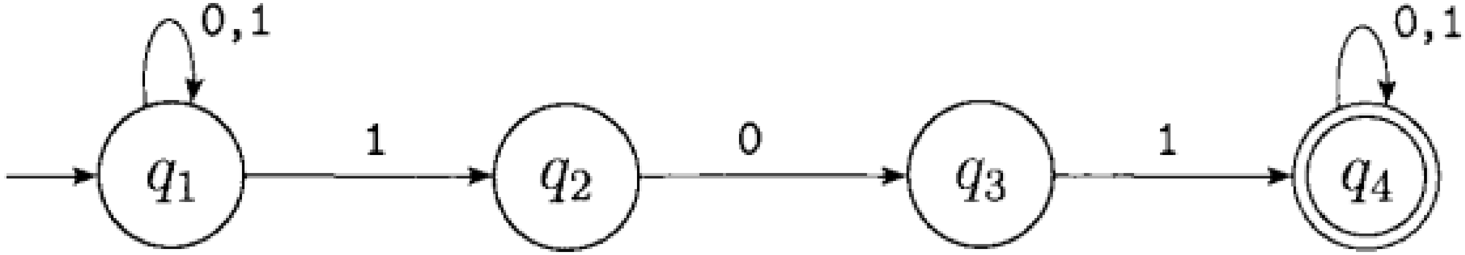
NFA:

- Slower: Simultaneous paths
- Simple Design: Easy to express and joins multiple machines

How does an NFA compute?

- After reading a symbol, the machine can split into multiple copies and follow all possibilities
- Each copy receives same input symbol and proceeds as before
  - If there's any choice, the machine splits again
- If the next symbol doesn't appear in the arrow, the copy of that machine dies
- If after the input of the complete string, one of the states is in final state, then NFA accepts the string

Does this NFA accept 010110?



Practice

Draw the state diagram of the NFA that can recognize the following languages:

- L(M) = { w | w begins with 101 } over Alphabet Σ = {0, 1}
- L(M) = { w | w begins with abb } over Alphabet Σ = {a, b}

- L(M) = { w | w ends with 101 } over Alphabet Σ = {0, 1}
- L(M) = { w | w ends with aa } over Alphabet Σ = {a, b}

- L(M) = { w | w contains with 110 } over Alphabet Σ = {0, 1}
- L(M) = { w | w contains with abb } over Alphabet Σ = {a, b}

- L(M) = { w | w is exactly 101 } over Alphabet Σ = {0, 1}