THEORY OF COMPUTATION

BOOK

- Theory of Automation by
  K.L.P Mishra

What is Theory of Automation

It is a branch of Computer Science and Mathematics which deals with how efficiently problems can be solved on a Computation model using an algorithm.

Definition of Automation:

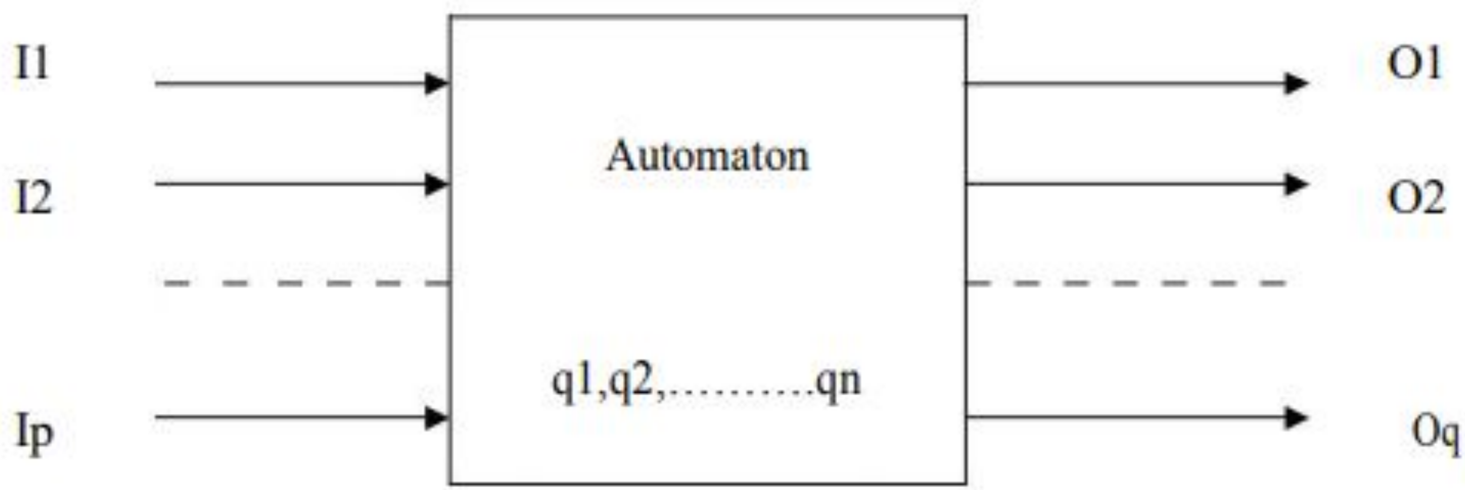
An automation is defined as a system where information are transformed for performing some functions without human intervention.

Definition of Automation

It is a model / Construct that accepts input and produces output, may have some temporary storage and can take decision.

Description of Automaton

An automaton can be defined in an abstract way by the following figure.



Model of a discrete automaton

Example: Compilers

- How Compilers Works?

int a,b,c;   There is no error

Int a,b,c*   There is an error

Compiler scans line by line instructions.

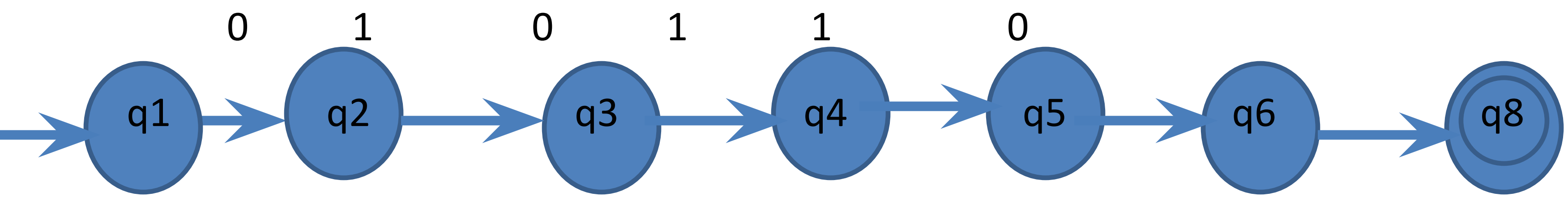
How it Checks?

The concept is based on Theory of Computation

Int a,b,c; $\longrightarrow$ Converted into binary language

- For example 010110 (Valid string)

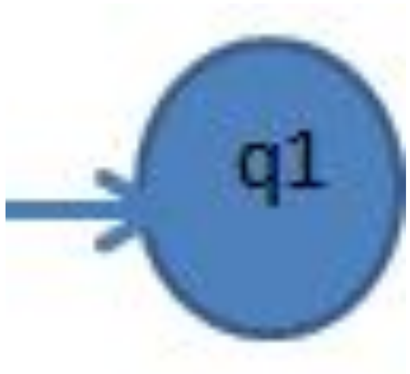
Now how compiler works



For example 010111 (Invalid string)

Compilers Generates Error Message, so the string 01011 is not acceptable.

Initial State



Arrow Indicates initial state

Final State

Sl.No	0	1
Q0	Q6	Q7
Q1	Q9	Q8

Alphabet

It is defined as a finite non empty set ∑ of symbols.

∑= { a,b, …..,Z}

Binary Alphabet ∑= {0,1}

Small letters are used for alphabet.

String

- It is a finite sequence of symbols from the alphabet
- "01110" is a string over the alphabet $\sum=\{0,1\}$
- "abcaa" is a string over the alphabet $\sum=\{a,b,c\}$

So w= 01110   w=abcaa

Lowercase letters u,v,w,x,y,z are used to represent a string.

Empty String

A string with no symbol. Denoted as ϵ / **λ**

Length of the string : Number of symbols present in the string.

W= 01010

|w| =5

Substring

- X is a substring of w if x appears consecutively with w. The substring x can be at the end or the beginning or in the middle of the string w.

  x=111

  w = 1110000   w=000111000 w=000111

Concatenation

x= 111 y =000

xy = 111000

OR

0 + 1

Closure : Repetition / Iteration

Ex $\sum$ = {1}

L = { null, 1, 11, 111, 1111,.........}

1*

$\sum^*$ = set of all string formed from the inputs, including null values.

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

$\sum^* = \{\boldsymbol{\lambda}, 0,00,000....,1,11,111,....., 011010, 1110110, 11100,.......\}$

$\sum^+$ = set of all string excluding null value

$\sum^+ = \{0,00,000....,1,11,111,....., 011010, 1110110, 11100,.......\}$

$\sum^* = \sum^+ - \{\boldsymbol{\lambda}\}$

i) Input: - At each of the discrete instants of time t1,t2,.....input values I1,I2……… each of which can take a finite number of fixed values from the input alphabet ∑, are applied to the input side of the model.

ii) Output : - O1,O2….are the outputs of the model, each of which can take finite numbers of fixed values from an output O.

iii) States : - At any instant of time the automaton can be in one of the states q1,q2…..qn

iv) State relation : - The next state of an automaton at any instant of time is determined by the present state and the present input. ie, by the transition function.

v) Output relation : - Output is related to either state only or both the input and the state. It should be noted that at any instant of time the automaton is in some state. On 'reading' an input symbol, the automaton moves to a next state which is given by the state relation.

NOTATION

An **alphabet** Σ is a finite set (e.g., $\Sigma = \{0,1\}$)

A **string** over Σ is a finite-length sequence of elements of Σ

Σ^* denotes the set of finite length sequences of elements of Σ

For x a string, $|x|$ is the **length** of x

The unique string of **length 0** will be denoted by ε and will be called the **empty** or **null string**

A **language over Σ** is a set of strings over Σ, ie, a subset of Σ^*

A deterministic finite automaton (DFA) is represented by a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$:

Q is the set of states (finite)

Σ is the alphabet (finite)

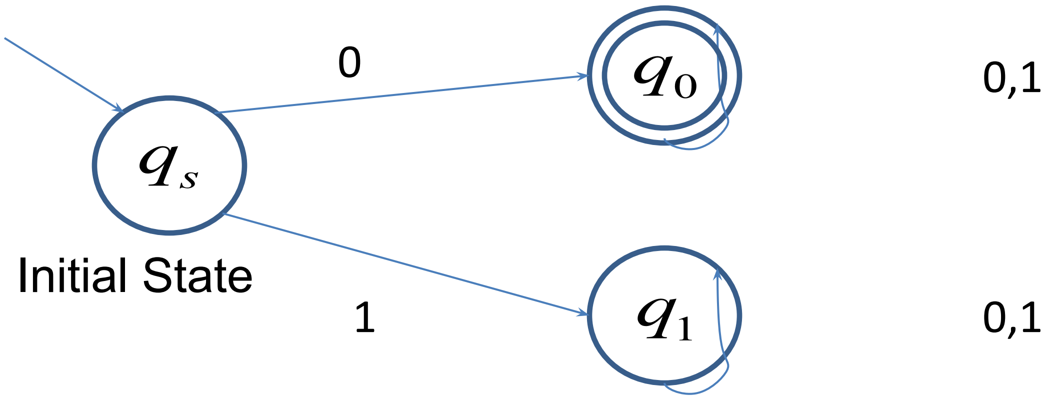
$\delta : Q \times \Sigma \to Q$ is the transition function

$q_0 \in Q$ is the start state

$F \subseteq Q$ is the set of accept states

$L(M)$ = the language of machine M
= set of all strings machine M accepts

Finite Automaton - An Example

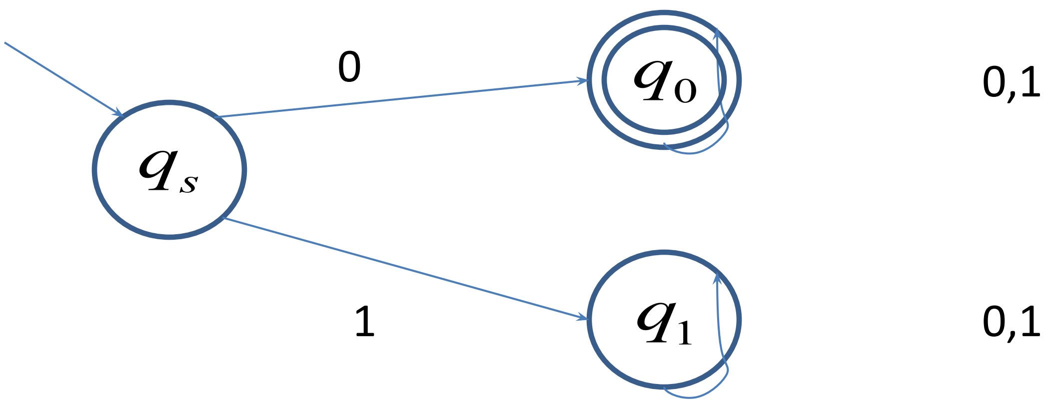


States: $Q = \{q_s, q_0, q_1\}$

Initial State: q_s

Final State: q_0

Finite Automaton – An Example



Transition Function: $\delta(q_s,0)=q_0 \quad \delta(q_s,1)=q_1$

$\delta(q_0,0)=\delta(q_0,1)=q_0 \quad \delta(q_1,0)=\delta(q_1,1)=q_1$

Alphabet: $\{0,1\}$. **Note:** Each state has **all** transitions .

Accepted words: $0,00,01,000,001,\ldots$

Finite Automaton – Formal Definition

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

1. Q is a finite set called the ***states***.
2. Σ is a finite set called the ***alphabet***.
3. $\delta : Q \times \Sigma \to Q$ is the ***transition function***.
4. $q_0 \in Q$ is the ***start state***, and
5. $F \subseteq Q$ is the set of ***accept states***.

Finite Automaton

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

where

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

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

$$F = \{q_0\}$$

δ	0	1
q_0	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Finite Automaton

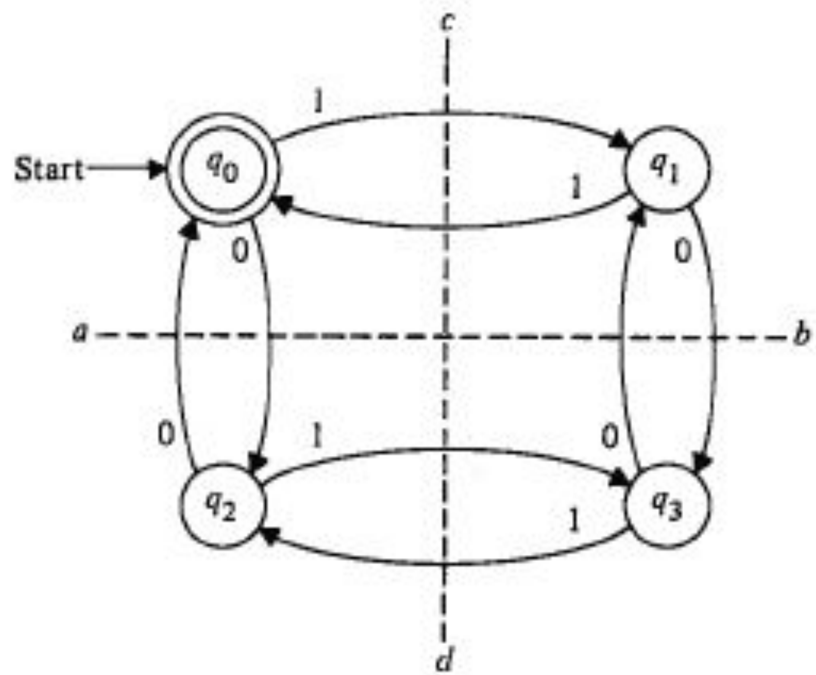


Fig. 2.2 The transition diagram of a finite automaton.

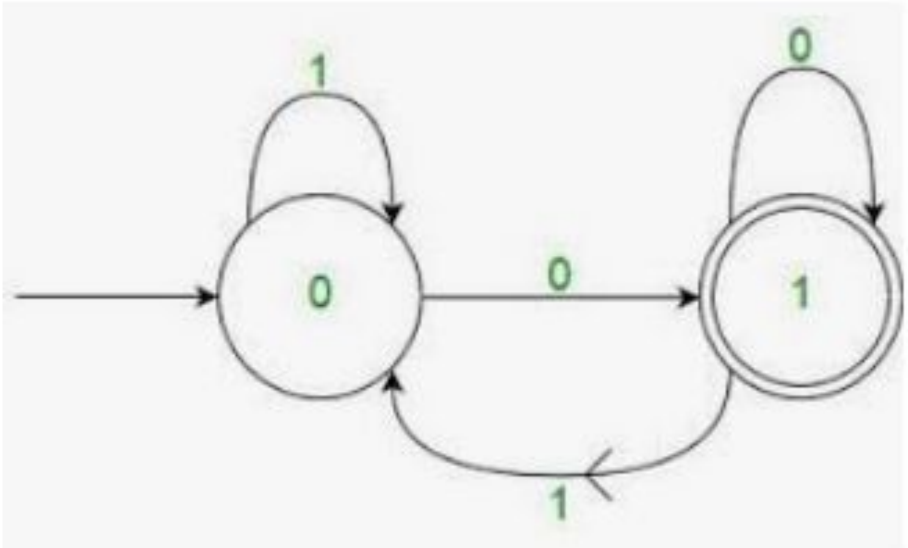
Example of DFA

Q={q0,q1}

∑={0,1}

Initial State={q0}

Final State={q1}



δ: (q0,1)=q0 (q0,0)=q1 (Q1,0)=q1   (Q1,1)=q0

Each state has **a single transition** for each symbol in the alphabet.

Example of DFA

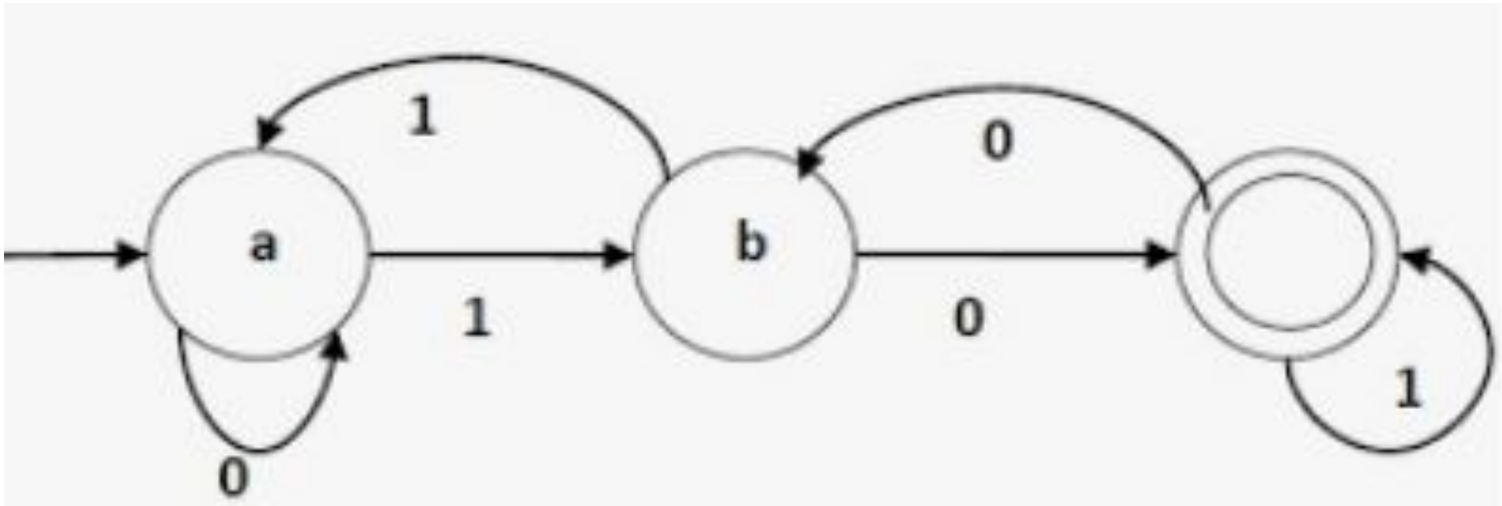
Q={qa,qb,qc}

∑={0,1}

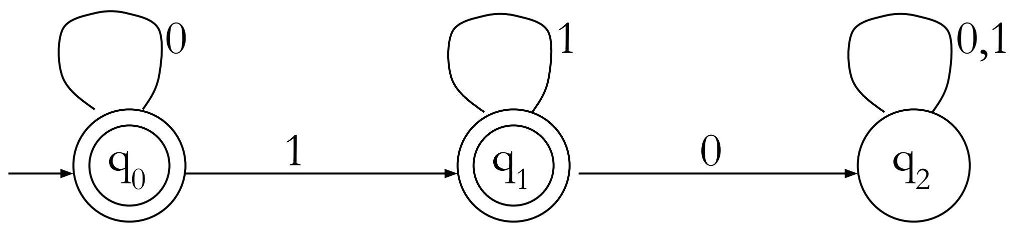
Initial State={qa}

Final State={qc}

δ: (Qa,1)=Qb (Qa,0)=Qa (Qb,0)=Qc (Qb,1)=Qa (Qc,0)=Qb (Qc,1)=Qc



Example



alphabet $\Sigma = \{0, 1\}$
start state $\mathcal{Q} = \{q_0, q_1, q_2\}$
initial state q_0
accepting states $F = \{q_0, q_1\}$

transition function δ:

states	inputs: 0	inputs: 1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_2	q_2

Whether the string w=0010010 is accepted by the automaton or not

$\delta(S_o,0010010)$

$=\delta(S_o,010010)$

$=\delta(S_o,10010)$

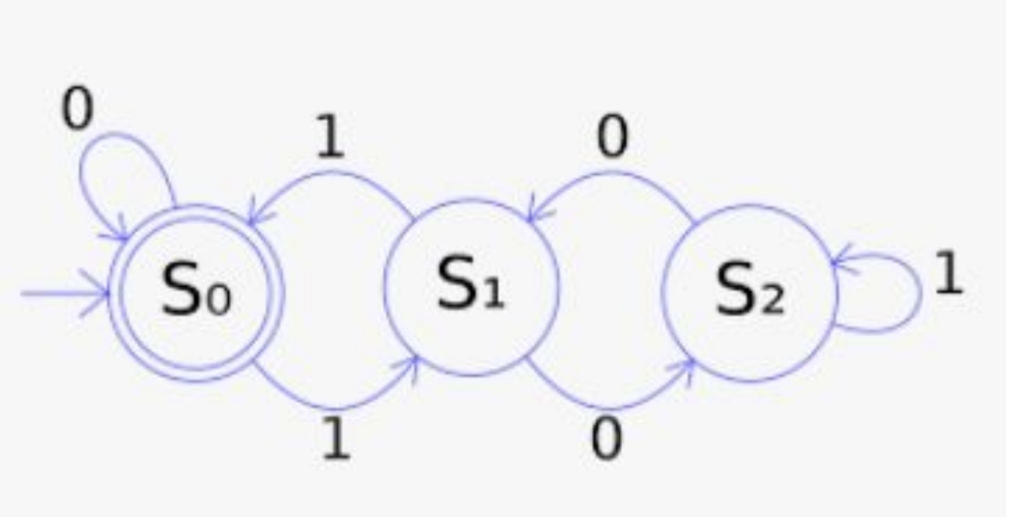
$=\delta(S_1,0010)$

$=\delta(S_2,010)$

$=\delta(S_1,10)$

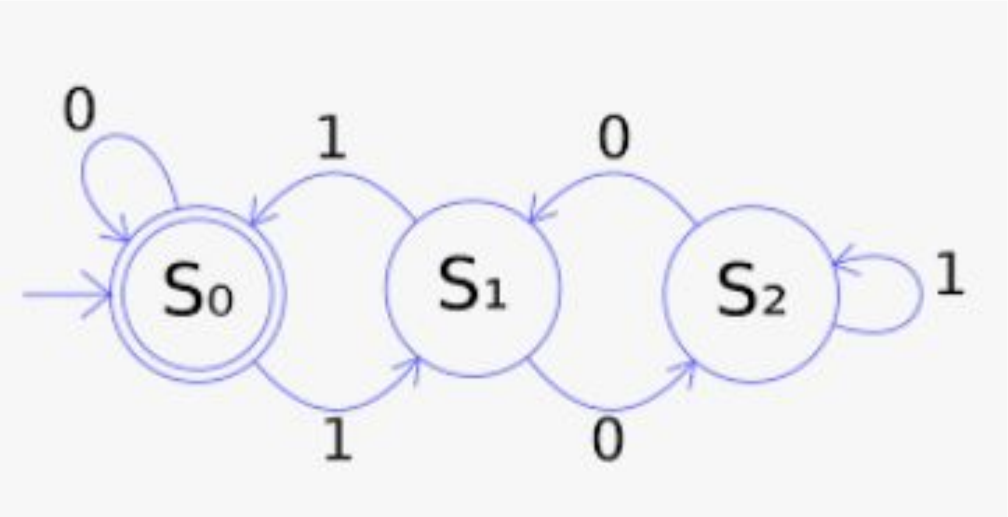
$=\delta(S_o,0)$

$=\delta(S_o,\boldsymbol{\lambda})$



As there are no more symbol and So is a final state(Double circle, The string is accepted by the automaton)

Transition Diagram



Transition Table

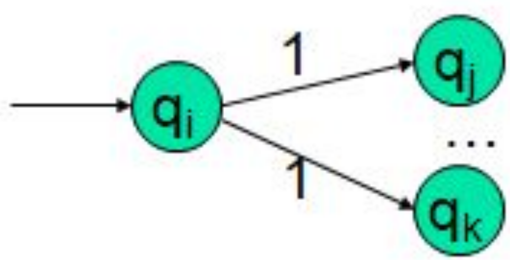
State	0	1
*So	So	S1
S1	S2	So
S2	S1	S2

Non-deterministic Finite Automata (NFA)

- A Non-deterministic Finite Automaton (NFA) consists of:
  - Q ==> a finite set of states
  - ∑ ==> a finite set of input symbols (alphabet)
  - q_0 ==> a start state
  - F ==> set of accepting states
  - δ ==> a transition function, which is a mapping between Q x ∑ ==> subset of Q
- An NFA is also defined by the 5-tuple:
  - {Q, ∑ , q_0,F, δ }

Non-deterministic Finite Automata (NFA)

- 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

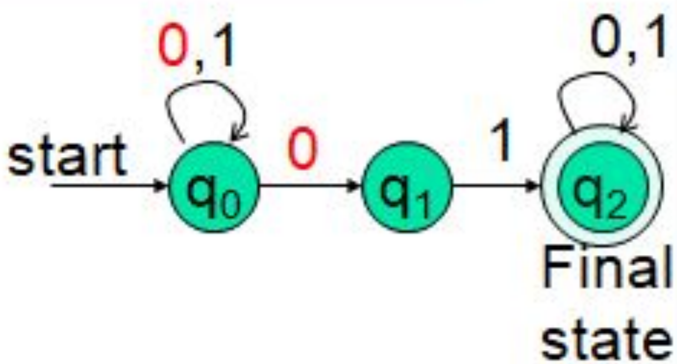


• Each transition function therefore maps to a <u>set</u> of states

Regular expression: $(0+1)^*01(0+1)^*$

NFA for strings containing 01

Why is this non-deterministic?



What will happen if at state q_1 an input of 0 is received?

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

symbols

δ (states)	0	1
→ q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	Φ	$\{q_2\}$
$^*q_2$	$\{q_2\}$	$\{q_2\}$

Nondeterministic Finite Automata

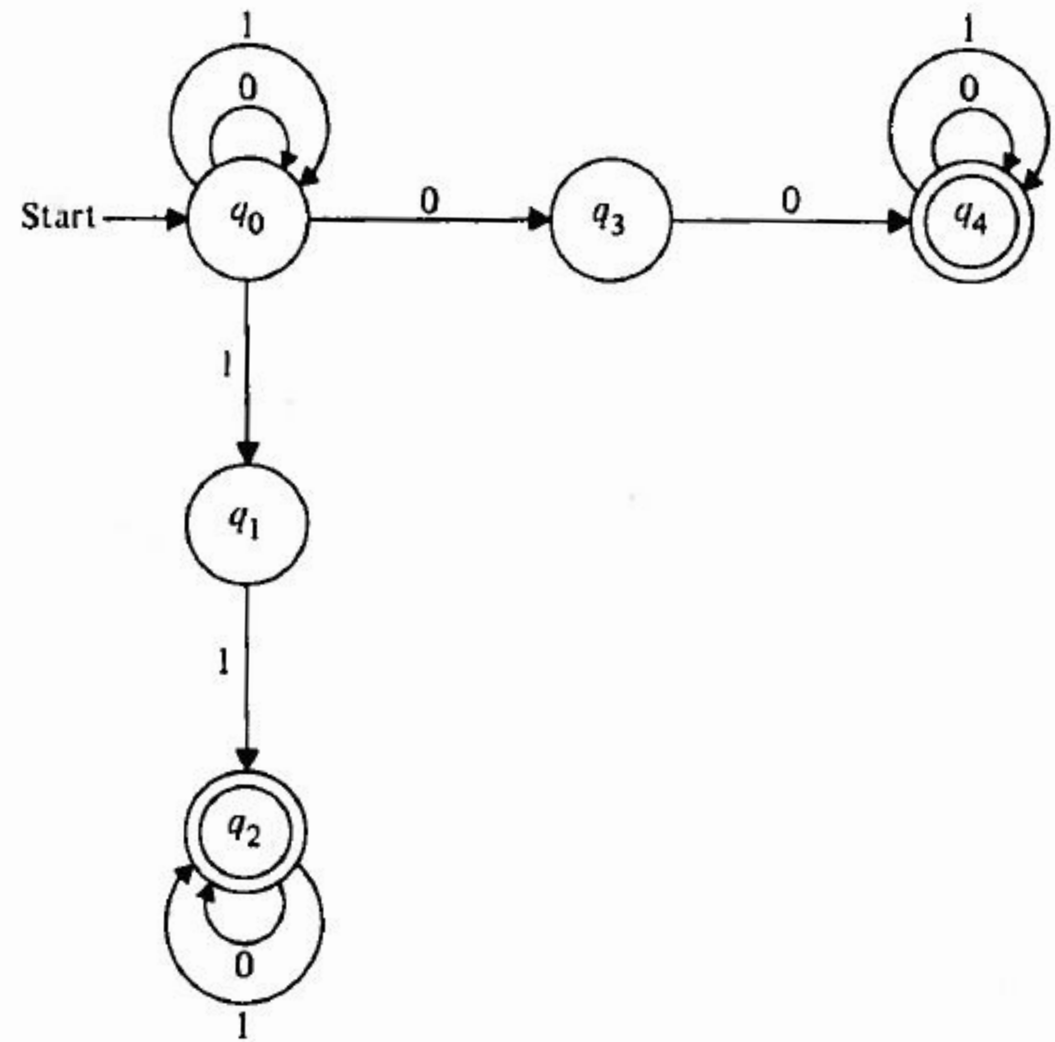


Fig. 2.5 The transition diagram for a nondeterministic finite automaton.

Nondeterministic Finite Automaton

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

where

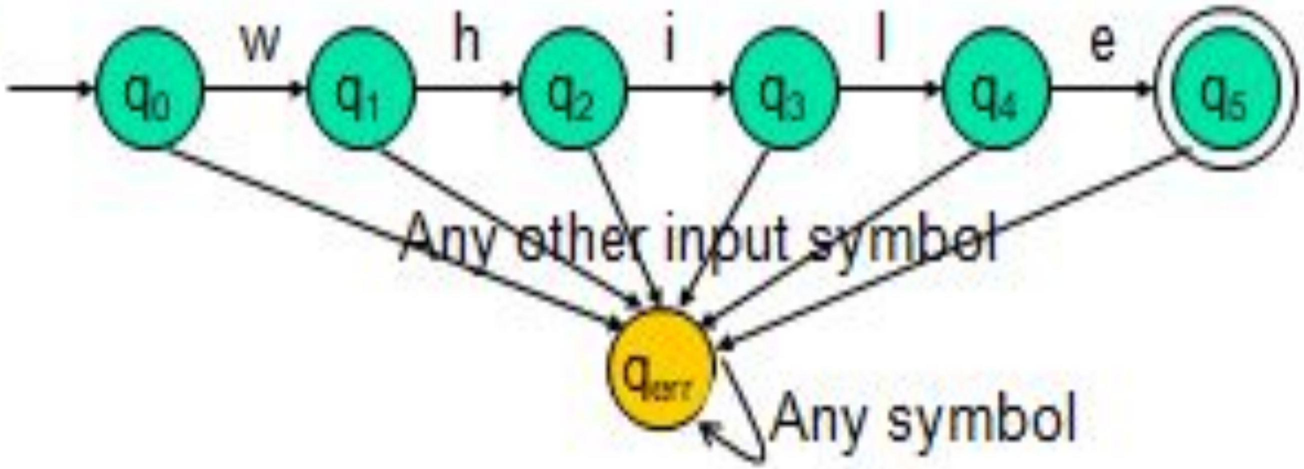
$$Q = \{q_0, q_1, q_2, q_3, q_4\}$$

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

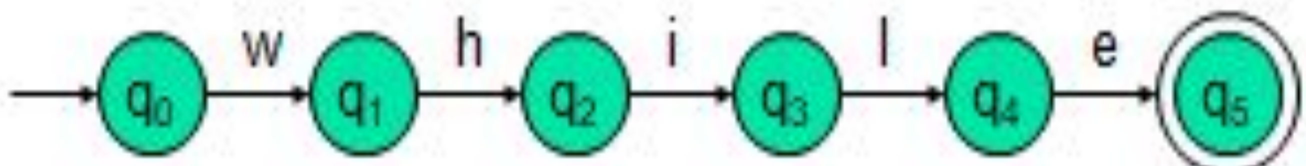
$$F = \{q_2, q_4\}$$

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

- A DFA for recognizing the key word "*while*"



- An NFA for the same purpose:

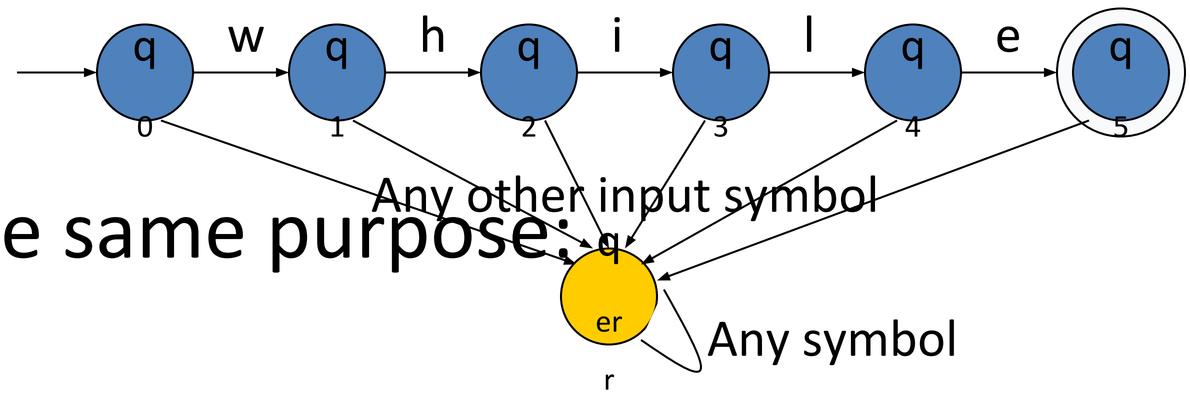


Transitions into a dead state are implicit

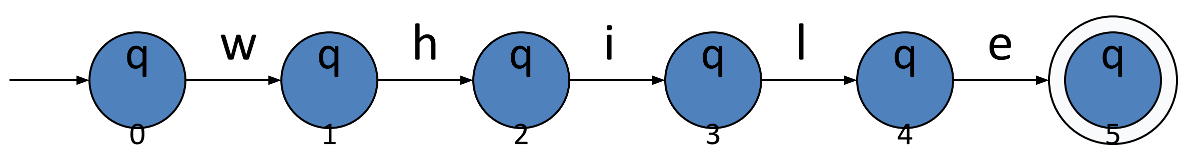
Note: Omitting to explicitly show error states is just a matter of design convenience
(one that is generally followed for NFAs), and
i.e., this feature should not be confused with the notion of non-determinism.

What is an “error state”?

- A DFA for recognizing the key word “*while*”



- An NFA for the same purpose:



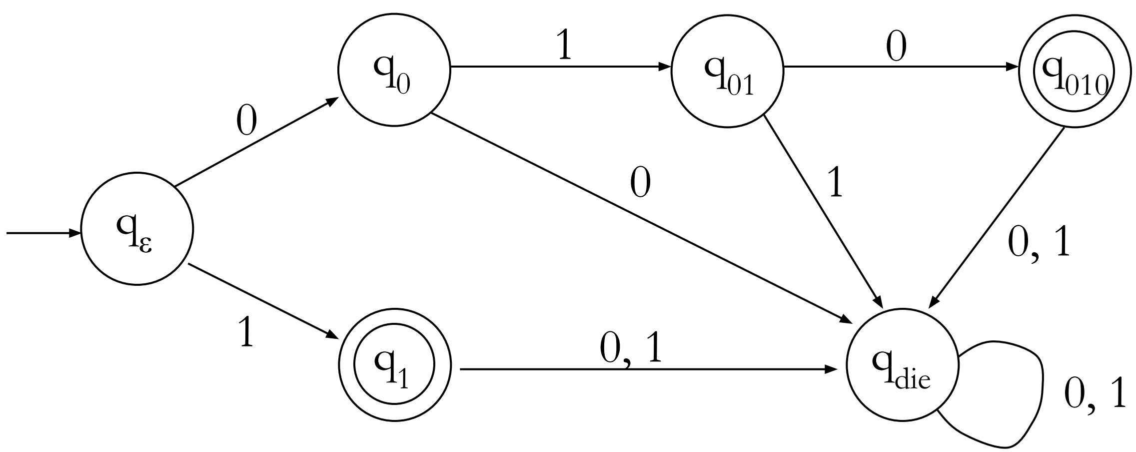
Transitions into a dead state are implicit

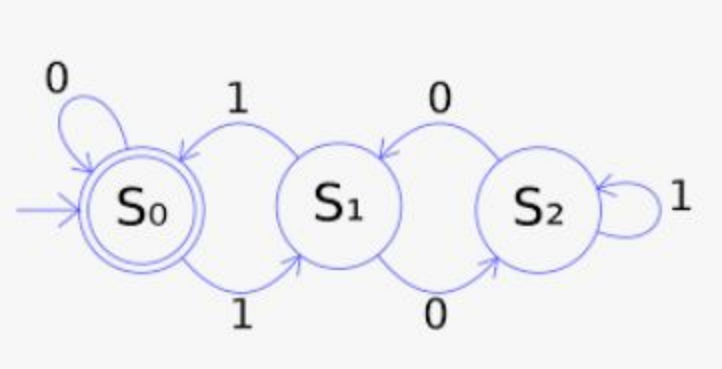
Examples

- Construct a DFA that accepts the language

$$L = \{010, 1\} \qquad\qquad (\,\Sigma = \{0, 1\}\,)$$

- Answer




0
1
0
S0
S1
S2
1
1
0