Theory of Automata and computation

Input → | Automatic Conversion | → Output

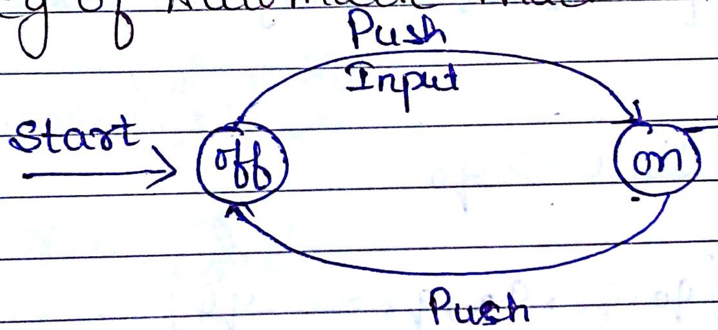
Language

a, b

Grammar Rules

1. b after a
2. Starting Symbol a

Eg of Automatic machine:-



→ state

valid state → on or off

Input → Push

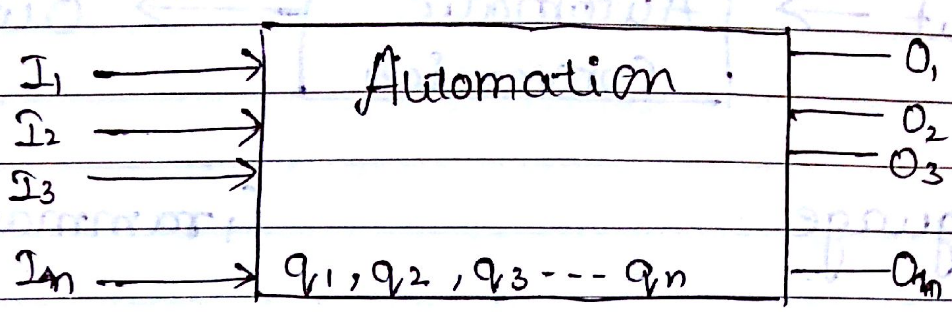
Output :→ change of state

Defination of Automation:-

An Automation is defined as a System where energy, materials and information are transffered transformed and used for performing some functions without direct participation of man eg are Automatic machine Tools, Automatic washing machine

1) All finite states of an Automation are represented by a circle
2) Input of the system are represented by arc
It shows the external influence on the system
3) One of the state is designated the start state

The State in which the system is placed initally It is indicate by the word Start and an arrow leading to that State



Model of Discrete Automation

alphabet

Denote by Σ ① Input $\rightarrow \{I_1, I_2, I_3, I_4, --- I_n\} \rightarrow$ finite

Q ② Valid State $\rightarrow \{q_1, q_2, q_3 --- q_n\}$

Denoted by F ③ Output $\rightarrow \{O_1, O_2, O_3 --- O_n\} \rightarrow$ limitted.

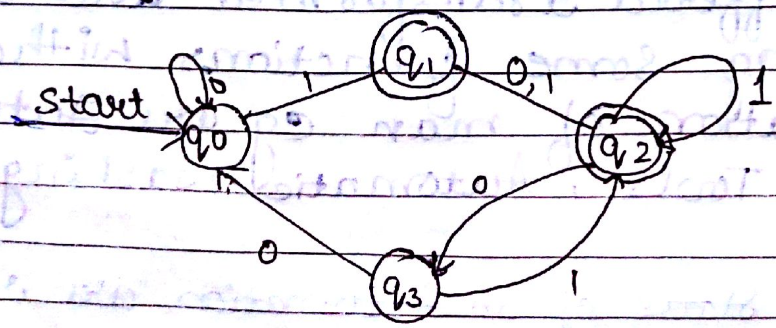
Final state

④ Initial state $\rightarrow q_0$

State $\rightarrow \{q_0, q_1, q_2, q_3 --- q_4\}$

⑤ $\delta \rightarrow$ transiting function

$M = \{Q, \Sigma, q_0, F, \delta\}$



01011

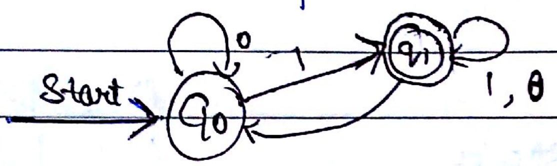
$Q = \{q_0, q_1, q_2, q_3\}$

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

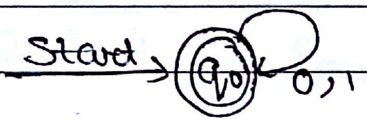
Initial State $\rightarrow q_0$
Final State $\rightarrow \{q_1, q_2\}$
$\delta \rightarrow Q \times \Sigma \longrightarrow Q$

$q_0 \times 0 \longrightarrow q_0$
$q_0 \times 1 \rightarrow q_1$
$q_1 \times 0 \longrightarrow q_2$
$q_2 \times 1 \rightarrow q_2$ → we check this state if it is final state then it accept the input

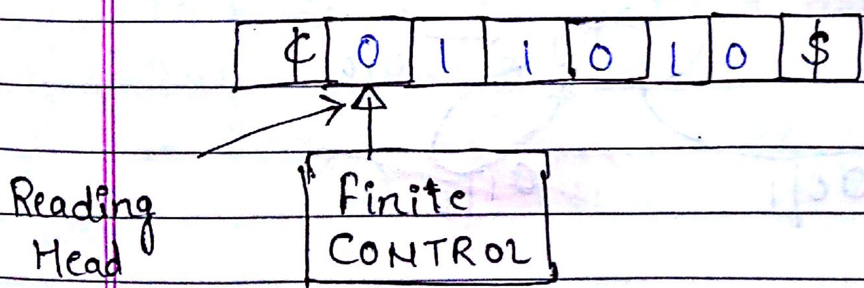
Regular Expression → eg 0001111111 then it
Regular Expression is $0^* 1^*$
Machine for $0^* 1^*$



OR



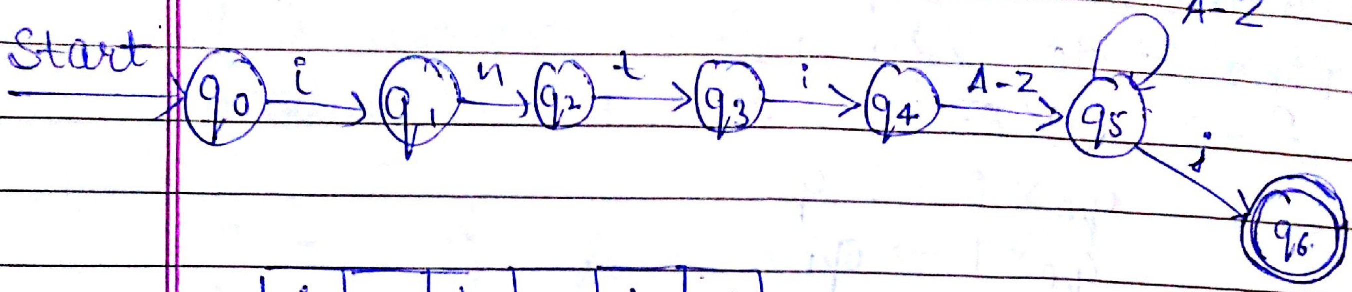
→ Finite automation, DFA automation



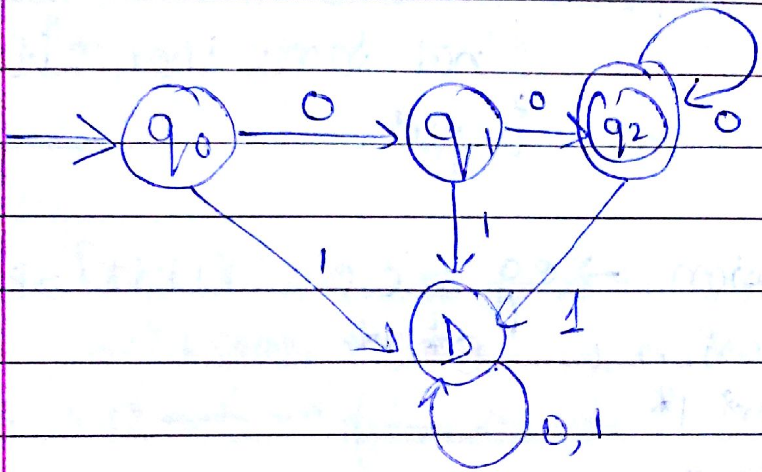
Block Diagram of a finite Automation.

States = $\{q_0, q_1\}$
$\Sigma = \{0, 1\}$

int i;=0



i	n	t		i	;

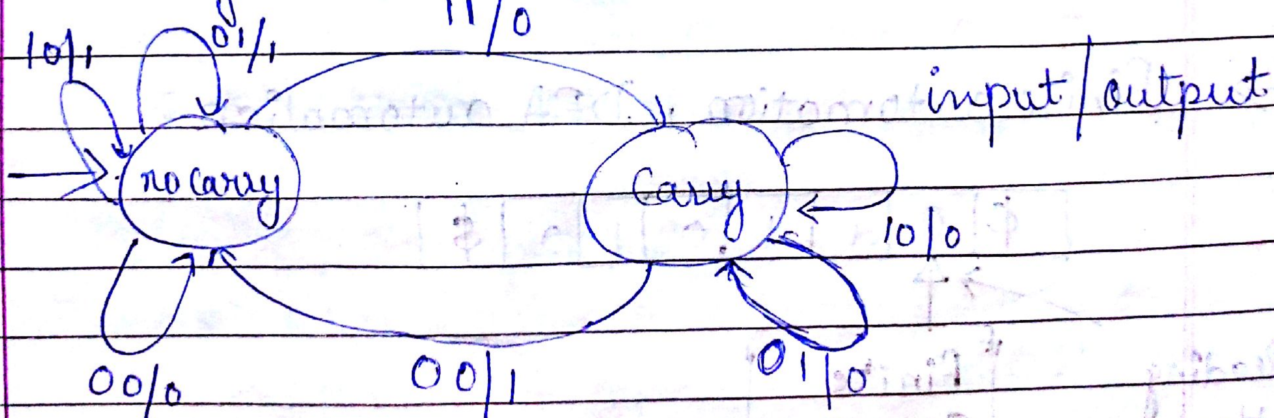


$$\begin{array}{r} 01110 \\ 00101 \\ \hline 00011 \end{array}$$

System Check two state

1) Carry state

2) No carry state



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

$L(M) = \{00, 10, 01, 11\}$

output $= \{0, 1\}$

Central Concepts of Automata Theory:

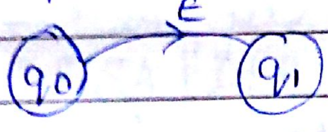
1) Alphabets (set of symbols) $\Sigma = \{0, 1\}$
2) Strings (list of symbols from an alphabet)
3) Language (strings from same alphabet)

This concepts include alphabets, strings & language

1) **Alphabet:** It is a finite nonempty set of symbol denoted by Σ eg $\Sigma = \{0, 1\}$ it is eg of Binary set
eg $\Sigma = \{a, b, c, z\}$ it is a set of lower case letter

2) **String:** A string is a finite sequence of symbols choosen from some alphabet (Σ) sigma
eg 01101 is a strings from the Binary alphabet $\Sigma = \{0, \}$

- **Empty String:** It is defined as a string with zero occurances of symbols. This string denoted as ϵ is a string that may be choosen from any alphabet



- **Length of a String:** String are classified by their length. Length means no of position for symbol in a string. The standard notation for the length of string is donated as $|w|$
~~Let~~ $w = \{1100011\}$

$|w| = 7$ $\qquad$ $|\epsilon| = 0$

The Length of String $|w|$ is 7.

- **Powers of an alphabet (Σ):** If Σ is a alphabet we can express the set of all string of a certain length from that alphabet by using an exponential notation
Σ^k where K is an integer where K = 3
eg:- $\Sigma^3 = \{000, 001, 101, 010, 110, --- 111\}$

$\Sigma^0 = \{\epsilon\}$, $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 --- \Sigma^n$

$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3$

- **Contactination of String:-**
$x = 01101$, $y = 0011 \longrightarrow xy = 011010011$

3) **Languages:-** A set of strings all of which are choosen from some Σ^*, where Σ is a particular alphabet is called language
If L is a language and Σ is alphabets and language is a subset of alphabets
$L \subseteq \Sigma^*$ over Σ
$L = \{\Sigma^1$
Any language over Σ is also the language of any alphabets i.e supper set of Σ

- **Why we are using term language:-**

Because every language can be viewed as set of string.

eg:- $\Sigma = \{a, b, c, ---- z\}$

L is English $L \subseteq \Sigma^*$

English is collection of legal english word is a set of string over the alphabets that consists of all the 26 letter

C-language + legal program are subset of the possible string that can be formed from the alphabets of the language
This alphabet is subset of ASCII

$\phi \rightarrow$ Empty language is a language of over any alphabet
$\{\epsilon\} \rightarrow$ A language consisting of only the empty String

$$\phi \neq \{\epsilon\}$$

ϕ has no string $\{\epsilon\}$ has 1 String

NOTE:- For any language alphabet are finite. A language can have an infinite no of string, are restricted to consist of string from one fixed finite alphabet

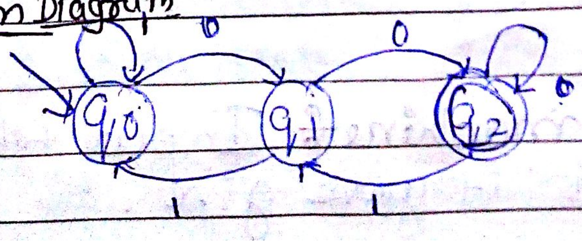
Assignment:- Transition System

FSA | DFSA | DFA :-

$$M = \{Q, \Sigma, \delta, q_0, F\}$$

$$M = \{H, \Sigma, \delta, H_0, F\}$$

Transition Diagram



$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{0, 1\}$
$\delta = Q \times \Sigma \rightarrow Q$
Intinal State q_0
$F = \{q_2\}$

Transition Table:

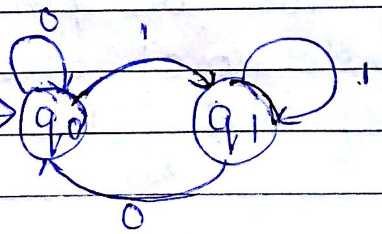
State	Input 0	Input 1
$\rightarrow q_0$	q_1	q_0
q_1	q_2	q_0
q_2* (q2)	q_2	q_1

Final State

eg:

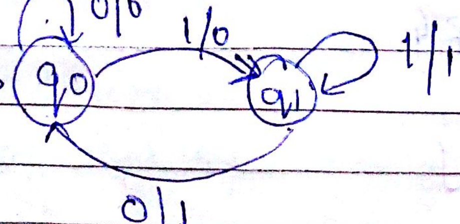
State	Σ 0	Σ 1
$\rightarrow q_0$	q_0	q_1
q_1	q_0	q_1

Diagram:

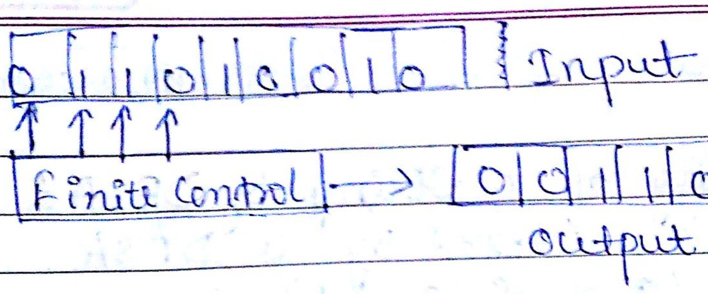


eg $\rightarrow$ Input $\rightarrow$ 01101001
output $\rightarrow$ 00110100

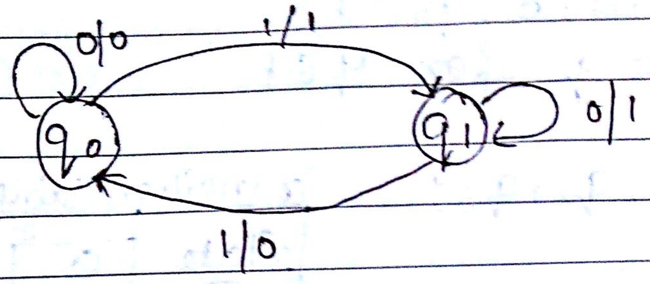
Given:



One moment delay machine: Input which it take the same output it gave give.
It start from zero



eg :-



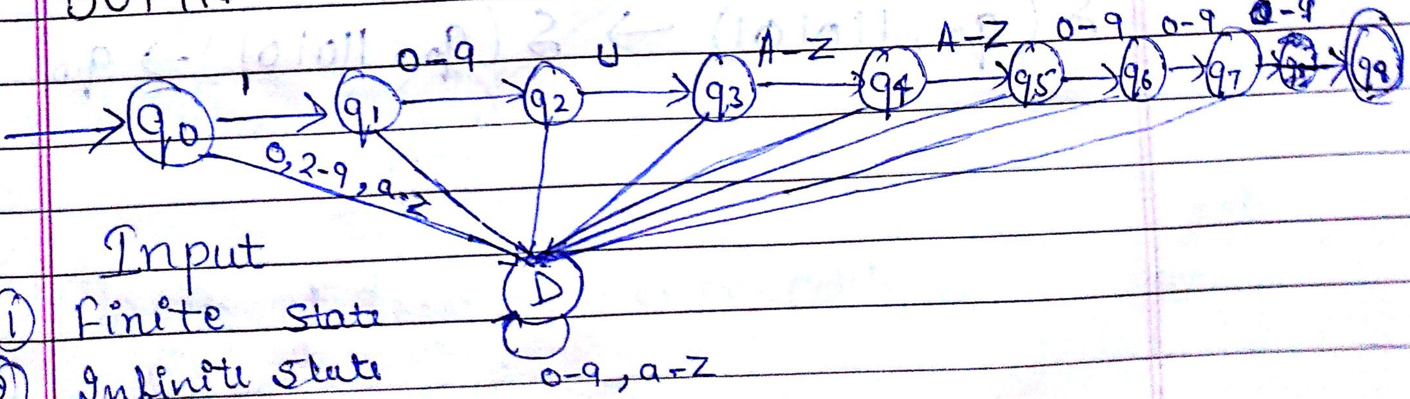
State	0		1	
		output		output
q_0	q_0	0	q_1	1
q_1	q_1	1	q_0	0

Input → 1 1 0 0 1 1 0 1
Output → 1 0 0 0 1 0 0 1

If odd no of ones in Input then output of Last is one i.e is called parity checker and viceversa for even no of ones in Input then output of last is Zero

-11UCS001

BUPIN



Input
1. Finite State
2. Infinite State

$\Sigma = \{0-9, a-z\}$

It is a String acceptor or BUPIN acceptor

Acceptability of a String by DFSA : A String w is accepted by a DFSA
$M = (Q, \Sigma, \delta, q_0, F)$
if $\delta(q_0, w) = q$ for $q \in F$

$Q = \{q_0, q, q_2, q_3\}$
$\Sigma = \{0, 1\}$
$F = \{q_0\}$

Transition table :

State	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

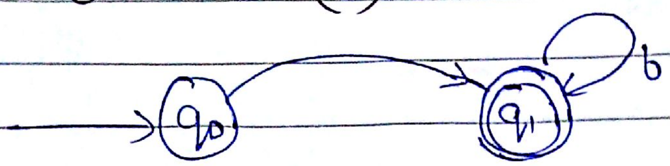
Input - 110101

$\delta(q_0, 110101) \rightarrow \delta(q_1, 10101) \rightarrow$
$\delta(q_0, 0101) \rightarrow \delta(q_2, 101) \rightarrow \delta(q_2, 01)$
$\delta(q_1, 1) = q_0 \in F$

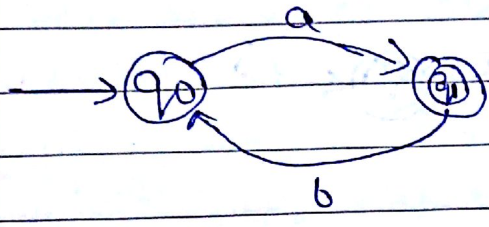
Hence this machine accept the String

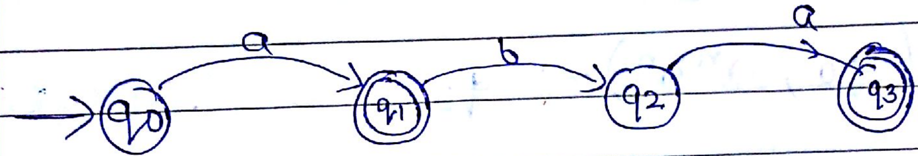
$\delta(q_0, 110101) \rightarrow \hat{\delta}(q_0, 110101) \rightarrow q_0$

Eg: $ab^n \to a(b)^*$

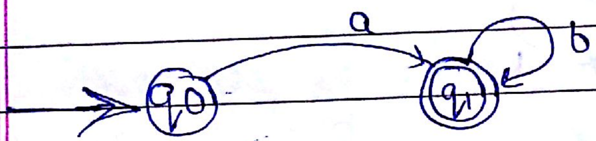


→ aba

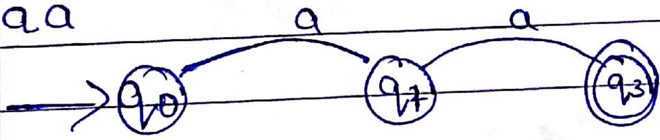


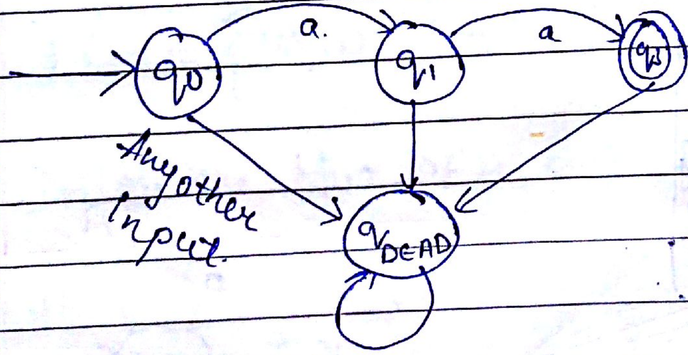


→ $ab^na \to a(b)^*a$.

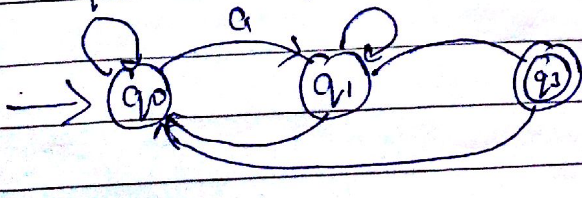


→ aa

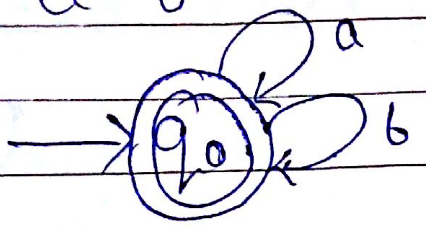




To prevent q_{DEAD} we use try catch

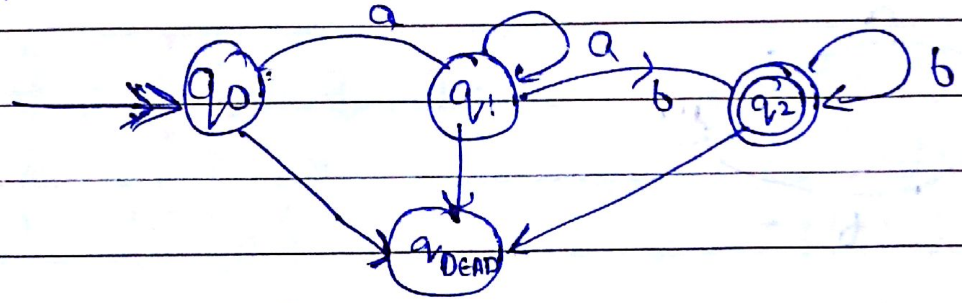


→ $a^* b^* \longrightarrow 0^* 1^*$



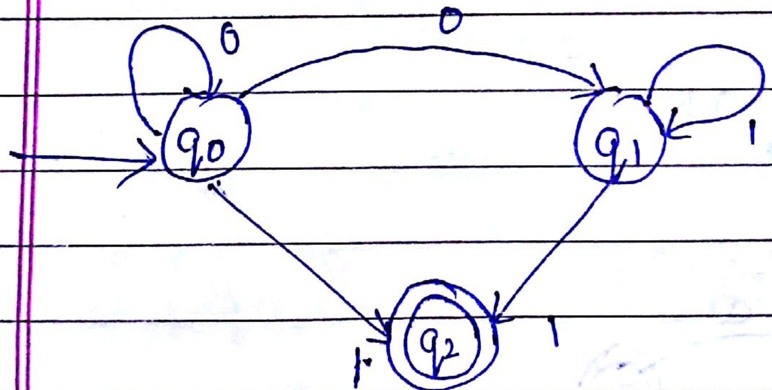
→ $a^n b^n, \ n > 0 \quad \Sigma^* = \{a, b\} \quad \Sigma^+, \Sigma^* \rightarrow$ 0 occur includ

Σ^+



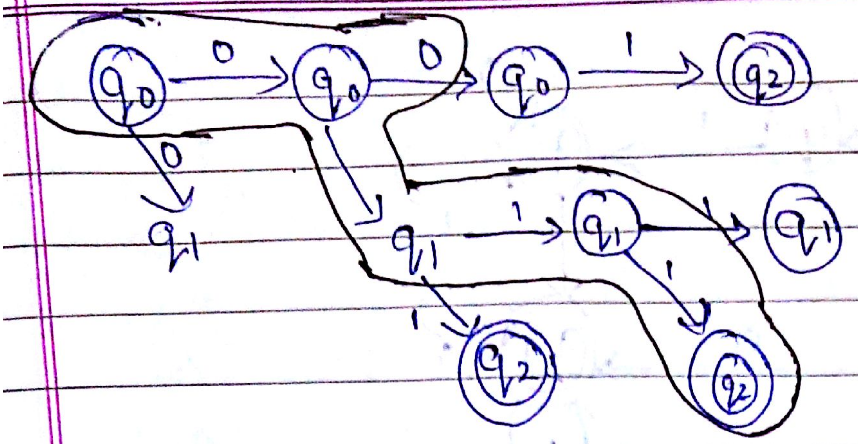
$\delta(q_0, a^n b^n) \rightarrow q_2$

NFSM (Non Deterministic finite State Machine):



Transition Table:

State	Σ: 0	Σ: 1
→ q_0	$\{q_0, q_1\}$	q_2 ~~$\{q_1, q_2\}$~~
q_1	ϕ	$\{q_1, q_2\}$
q_2	ϕ	ϕ



0011

Some moves of the machine cannot be determine uniquely by the input symbol and the present state. Such machine are called Non Deterministic Finite automation.

$$M_N = \{Q, \Sigma, \delta, q_0, F\}$$

Non Deterministic machine

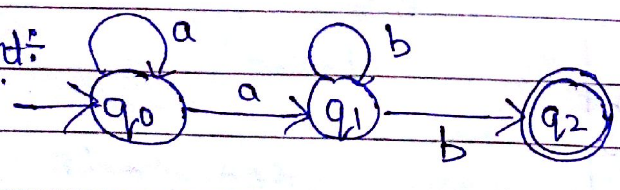
$Q \rightarrow$ set of state
$\Sigma \rightarrow$ input alphabet
$\delta \rightarrow Q \times \Sigma \rightarrow 2^Q \quad p(Q)$

$$L(M) = \{w \mid w \in \Sigma^*\}$$

$$L(M) = \{w \mid w \in \Sigma^*, \delta(q_0, w) \rightarrow Q \cap F \neq \phi\}$$

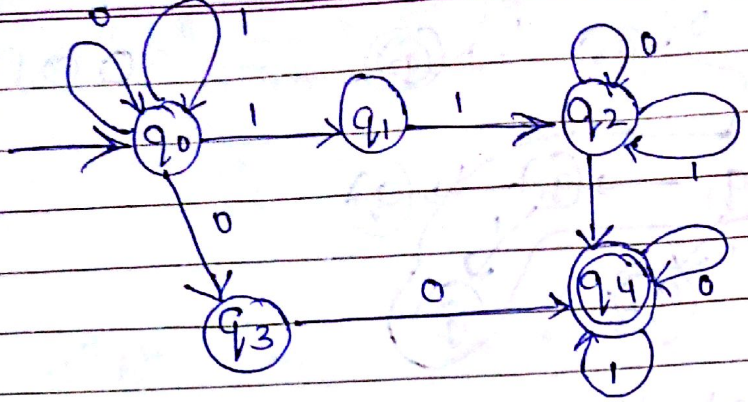
$\longrightarrow$ Difference b/w DFA & NDFA

Assignment:



Input $\rightarrow$ aaaabb.

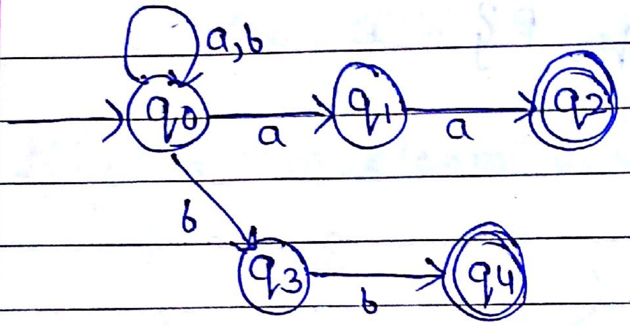
Que →



Input - 0100

$q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_2 \rightarrow q_1$

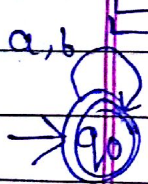
Que →

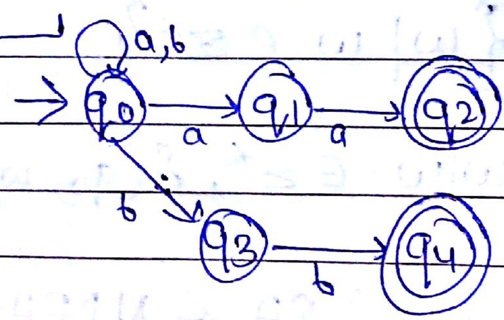


→ NDFA

$a^* bb$, $b^* aa$

$(a+b)^* \ (bb+aa)$



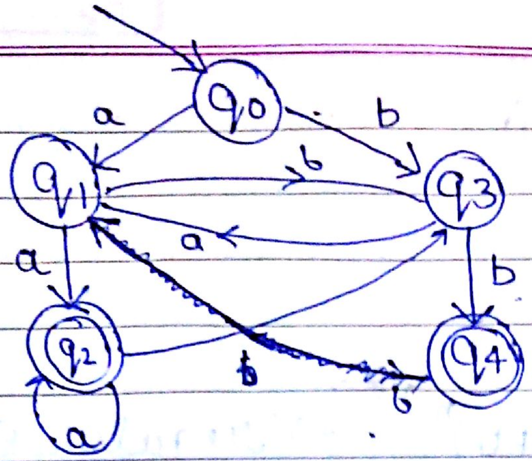


Transition table :-

State	a	b
→ q_0	$\{q_0, q_1\}$	$\{q_0, q_3\}$
q_1	q_2	ϕ
q_2	ϕ	ϕ
q_3	ϕ	q_4
q_4	ϕ	ϕ

Date ______
Page ______

(3)



$(a+b)^* \ (aa+bb)$

aaabb
bbbaa

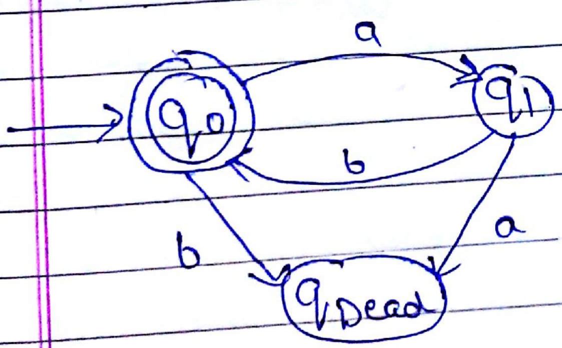
→ To Convert NDFA and DFA.

NDFA

State	0	1
q_0	$\{q_0, q_1\}$	ϕ
q_1	ϕ	$\{q_1, q_2\}$
q_2	ϕ	ϕ

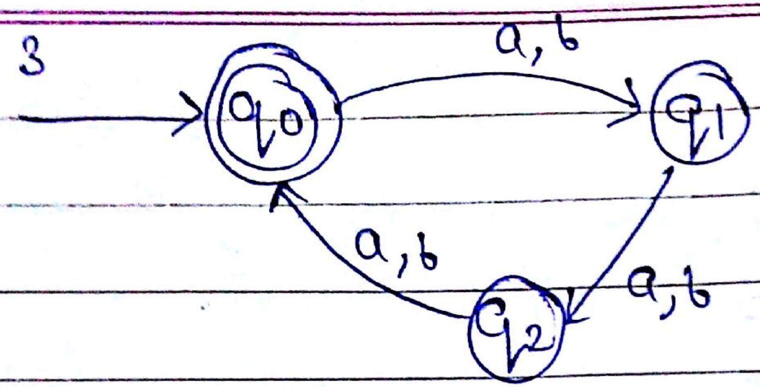
DFA

q_0	$[q_0 \ q_1]$	ϕ
$[q_0 \ q_1]$	$[q_0 \ q_1]$	$[q_1, q_2]$
$[q_1 \ q_2]$	ϕ	$[q_1 \ q_2]$
ϕ	ϕ	ϕ



$(ab)^*$ abababab.

3



$\Sigma = \{a, b\}$

$\Sigma^3 =$ aaa , $(a+b)^n$ n*3. → regular Expression
bbb
abb
baa

NFA ↔ DFA

Transition table ↔ Regular Expression

CSE4projects.wordpress.com
↳ Theory of Automata Notes