27/12 1. FUNDAMENTALS.

Symbol :- A symbol is an abstract entity which indicates a letter or a digit.

String :- A string is a finite sequence of symbols.

→ For Ex:- 0 & 1 are symbols, 0110 is a string

→ The length of the string "|w|" is denoted by is the number of symbols composing the string.

For Ex:- w = 0110 then |w| = 4

→ The empty string is denoted by "ϵ" called null symbol.

→ A prefix of a string is any number of leading symbols of that string and a suffix is any number of trailing symbols of that string.

→ For Ex:- for the string 011 the prefixes are {ϵ, 0, 01, 011, 0110} and suffixes are {ϵ, 0, 10, 110, 0110}

Operation on Strings:-

Concatenation:- concatenation of two strings is writing the first one followed by the second string.

If $S_1 = 01$ $S_2 = 10$

$S_1S_2 = 0110$

→ The empty string is the identity for the concatenation operator.

i.e $\epsilon W = W\epsilon = W$

Kleen closure (*):- The kleen closure of a language is defined as

$$S^* = \{S^0 \cup S^1 \cup S^2 \cup \dots\}$$

for Ex:- $0^* = \{\epsilon, 0, 00, 000, \dots\}$

Positive closure:- The positive closure of a language is defined as

$$S^+ = \{S^1 \cup S^2 \cup S^3 \cup \dots\}$$

$$\boxed{SS^* = S^+}$$

considering S as 0

$$00^* = 0\{\epsilon, 0, 00, \dots\}$$

$$= 0, 00, 000, \dots = S^+$$

Alphabet:- An alphabet is finite set of symbols. It is denoted by "Σ"

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

Automata:- An Automation is defined as a system where information is transformed and transmitted for performing some function without direct participation of a Man.

→ Every Automation includes some essential features. It has a mechanism for reading input. The input is a string over a given alphabet written on an input file. The input file is divided into cells. Each of which can hold one symbol.

→ The input mechanism can read the input file from left to right one symbol at a time. The input mechanism can also detect the end of string.

→ It may have temporarily storage device consisting of an unlimited no. of cells each capable of holding a single symbol from an alphabet. The automation can read & change the contents of storage cell.

→ Finally automation has a control unit which can be in finite number of internal states and which can change state in some defined manner.

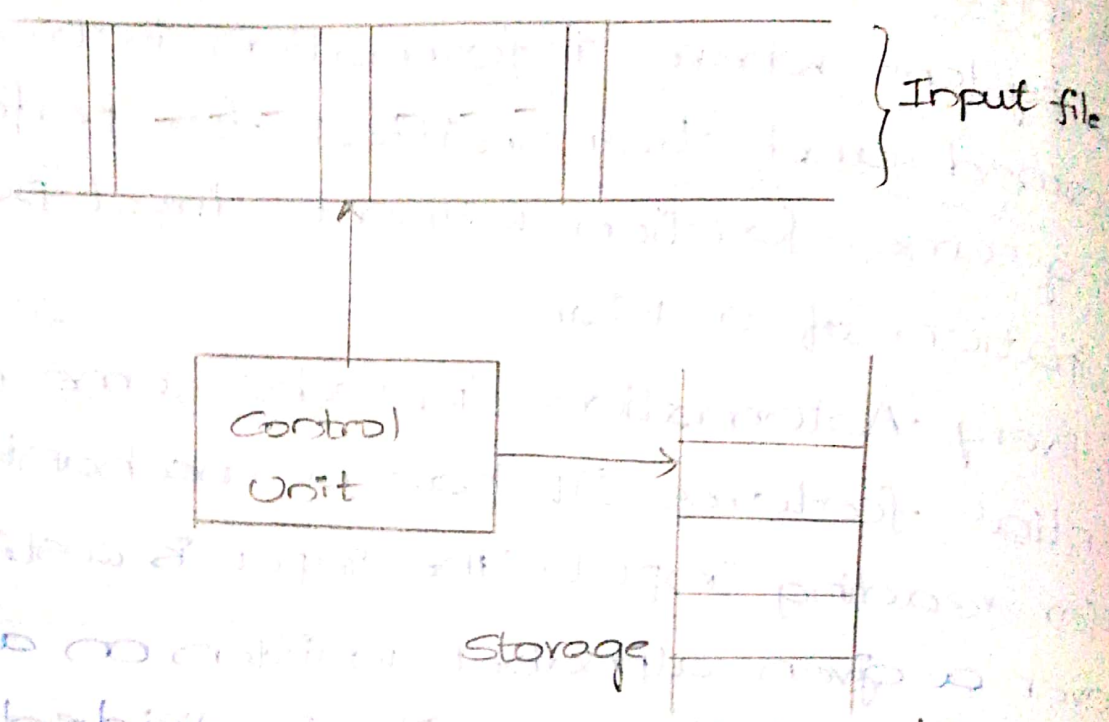


fig. Block Diagram of Automata

Applications:

→ Finite Automata are useful model for many important kinds of both hardware and software applications in computer science.

→ The following are some of the most important applications of finite automata

1. Software for designing and checking the behaviour of a digital circuit.
2. In the lexical phase of a typical compiler construction i.e to break up

given txt into logical units. Those log-ical units are also known as tokens.

28/12

3. Software for scanning large bodies of text such as collection of web pages to find occurences of words, phrase and other patterns

4. Software for verifying systems of all types that have finite number of distin-ct states such as communication proto-cols or protocol for secure exchange of information.

5. Formal languages and grammars are used widely to define programming language.

Finite Automata is of two types :-

1. Deterministic (DFA)
2. Non Deterministic (NFA)

DFA :-

A DFA is analytically defined by 5tuples

$M = (Q, \Sigma, \delta, q_0, F)$ where Q is finite set of internal states

Σ = finite set of symbols called input alphabet.

δ = transition f^n which maps $Q \times \Sigma \to Q$

$q_0 = q_0 \in Q$ is a initial state

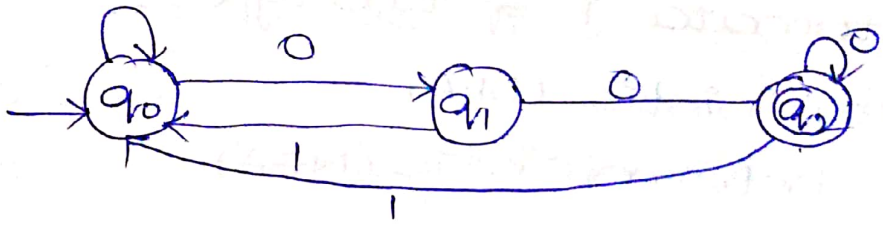
$F \subseteq Q$ is set of final states.

Note:- In DFA design all states must contain transitions for all input symbols and no state contain more than one part from the same input symbol.

→ Finite automata is represented by transition diagrams in which vertices represent the states and edges represent transitions.

→ In transition diagram starting state is indicated by "→" symbol and final state is indicated by "◎".

For Ex:



Here q_0 is initial state

q_2 is final state

→ The above transition diagram represents DFA which accepts all the string ending with "00"

$M = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$

δ is defined as $\delta(q_0, 0) = q_1$

$\delta(q_0, 1) = q_0$

$\delta(q_1, 0) = q_2$

$\delta(q_1, 1) = q_0$

$\delta(q_2, 0) = q_2$

$\delta(q_2, 1) = q_0$

(or)

δ	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_2	q_0
(q_2)	q_2	q_0

→ The acceptability of the string is processed as follows.

Consider the string 10100

$\delta(q_0, 10100) \vdash \delta(q_0, 0100)$

$\vdash \delta(q_1, 100)$

$\vdash \delta(\delta_0, 00)$

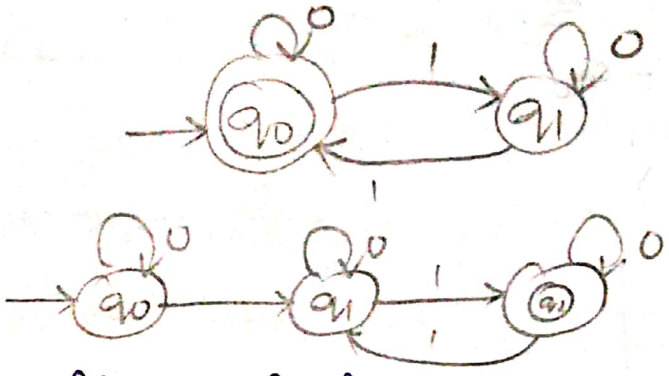
$\vdash \delta(q_1, 0)$

$\vdash \delta(q_0, \epsilon)$

Since, q_2 is the final states. It accepts the string 10100.

→ The DFA is halting in the final state after processing the given i/p string.

→ Design DFA which accepts strings with 0's and 1's such that the number of 1's should be even.



$M = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_0\})$

→ Design DFA of 0110. (checking)

$$\delta(q_0, 0110) \vdash \delta(q_0, 110)$$
$$\vdash \delta(q_1, 10)$$
$$\vdash \delta(q_0, 0)$$
$$\vdash \delta(q_0, \epsilon)$$

q_0 is a final state

As the given string is containing even no. of 1's the DFA is halting in the final state q_0. so the given string is accepted.

consider 01101

$$\delta(q_0, 01101) \vdash \delta(q_0, 1101)$$
$$\vdash \delta(q_1, 101)$$
$$\vdash \delta(q_0, 01)$$
$$\vdash \delta(q_0, 1)$$

$\vdash \delta(q_1, \epsilon)$

Rejected.

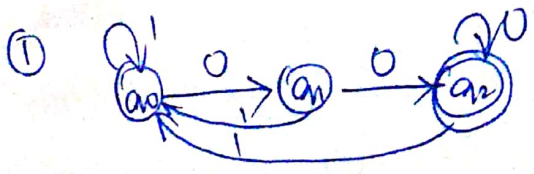
As the given string contains odd no. of 1's the DFA is halting in the non final state q_1 so it is Rejected.

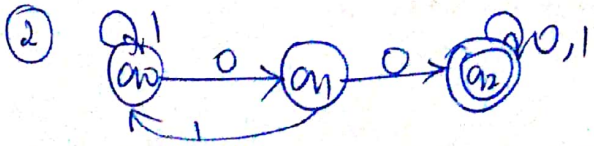
→ Design DFA which accepts the strings with 0's & 1's such that the string should end with 00.

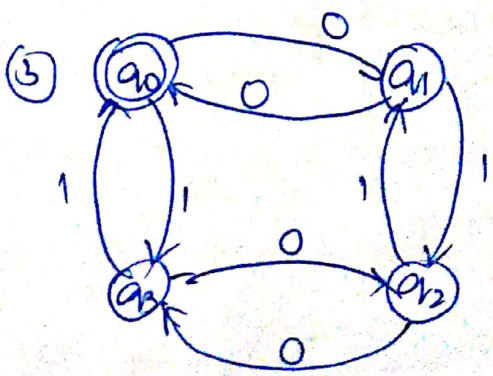
→ Design DFA for the strings with 0's & 1's such that the string should contain atlea-st two consecutive zeros.

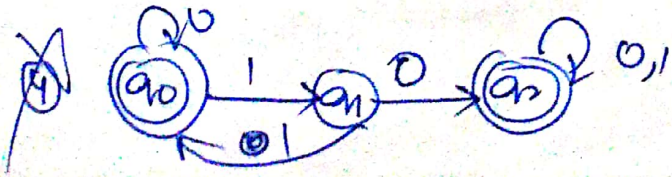
→ Design DFA with the strings of 0's & 1's such that string should contain even number of zero's and even number of 1's.

→ Design DFA which accepts the strings 0's and 1's such that strings should not con-tain atleast 2 consecutive zero's.









2/1/18 NON-DETERMINISTIC FINITE AUTOMETA.

In DFA all states must contain transi-stions for all the input symbols & also every state contain only a single or unique transition from the same input symbol. where as in NFA a state may contain more than one transition for same input symbol and also a state may not have transition for some of the input symbols

→ NFA is analytically defined a 5 tuples

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

Q = finite set of internal states

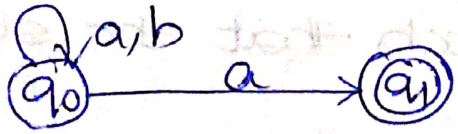
Σ = finite set of symbols called i/p alphabet

δ = transition f^n which maps $Q \times \Sigma \Rightarrow 2^Q$

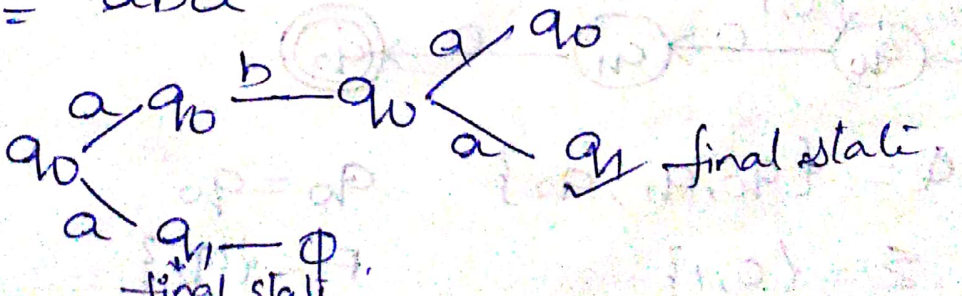
q_0 = initial state $\in Q$

F = set of final states $\subseteq Q$

Design NFA which accepts all the strings with a's & b's such that the string should end with a.



Eg: aba



$Q = \{q_0, q_1\}$

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

δ	a	b
$\rightarrow q_0$	$q_0 q_1$	q_0
(q_1)	ϕ	ϕ

$q_0 = q_0$

$F = \{q_1\}$

String aba Accepted because it is ending in the final state in one of the possible paths.

for abb

$q_0 \xrightarrow{a} q_0 \xrightarrow{b} q_0 \xrightarrow{b} q_0$ ✗

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} \phi$ ✗

The string abb is rejected.

Note:- In NFA for checking of acceptability of the string we must check all the possible paths if atleast one path reaching the final state after processing the given string it is accepted. Otherwise rejected.

3)1

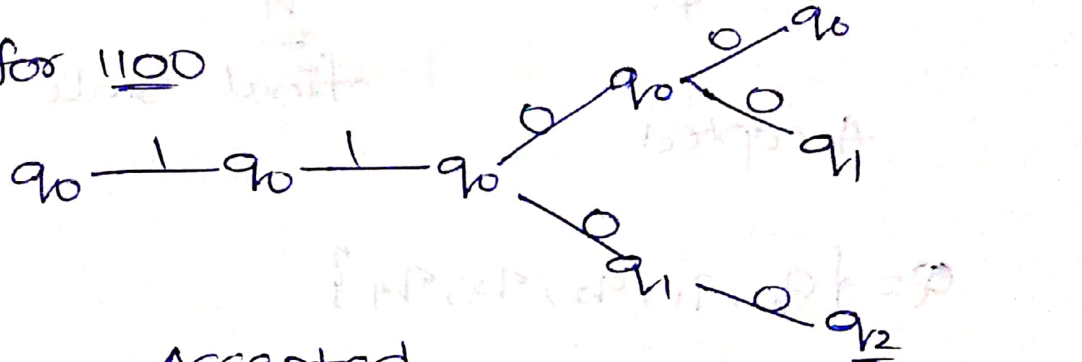
→ Design NFA which accepts all the strings with 0's and 1's such that the string end with 00

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

$\Sigma = \{0, 1\}$ $F = \{q_2\}$

δ	0	1
→ q_0	q_0, q_1	q_0
q_1	q_2	ϕ
(q_2)	ϕ	ϕ

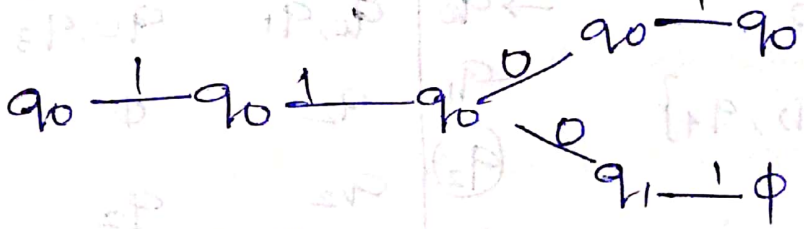
for 1100



Accepted.

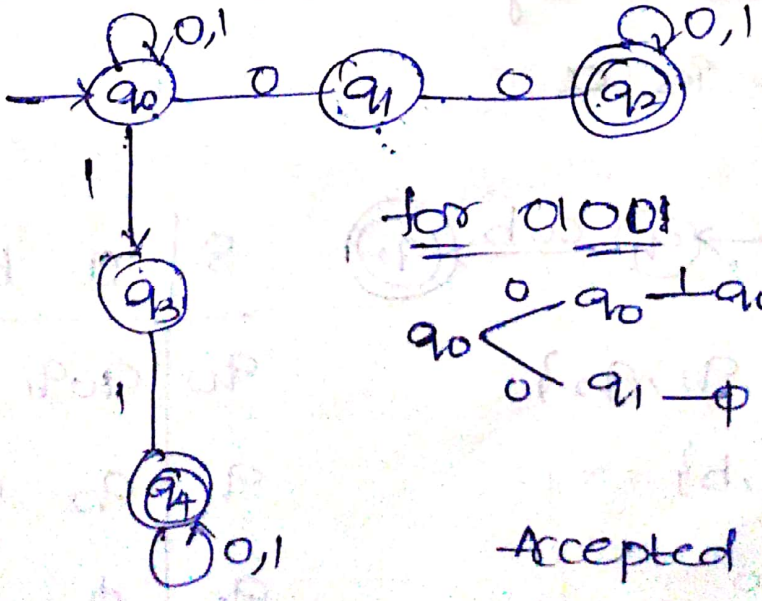
final state

for 1101



Rejected

→ Design NFA which accepts all the strings with 0's & 1's such that string should contain 2 consecutive zero's or two consecutive 1's.

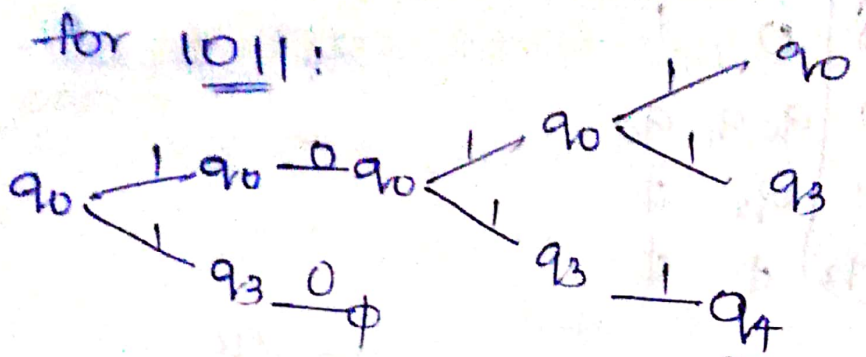


for 01001

final

Accepted

for 1011:



= final state

Accepted

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

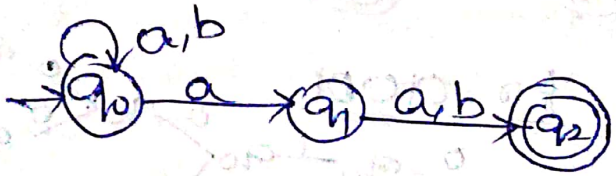
$\Sigma = \{0, 1\}$

$q_0 = q_0$

$F = \{q_2, q_4\}$

δ	0	1
$\rightarrow q_0$	q_0, q_1	q_0, q_3
q_1	q_2	ϕ
(q_2)	q_2	q_2
q_3	ϕ	q_4
(q_4)	q_4	q_4

→ Design NFA which accept all the strings with a's & b's such that the strings second symbol from the right hand side is a



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

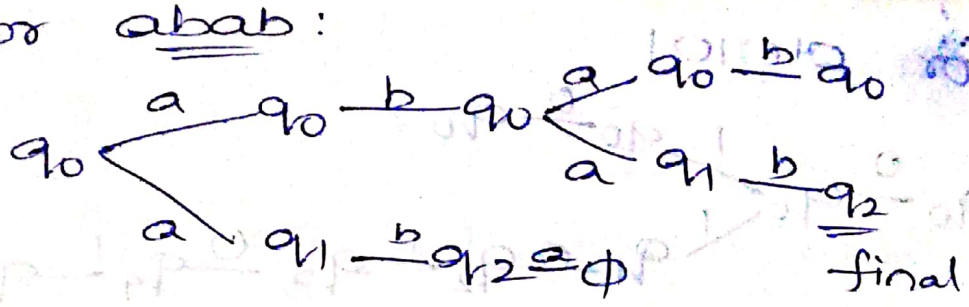
$\Sigma = \{a, b\}$

$q_0 = q_0$

$F = \{q_2\}$

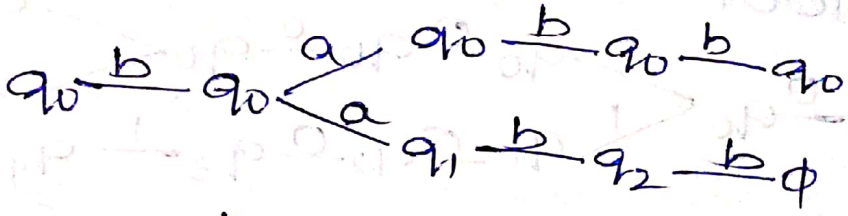
δ	a	b
q_0	$q_0 q_1$	q_0
q_1	q_2	q_2
q_2	ϕ	ϕ

for abab:



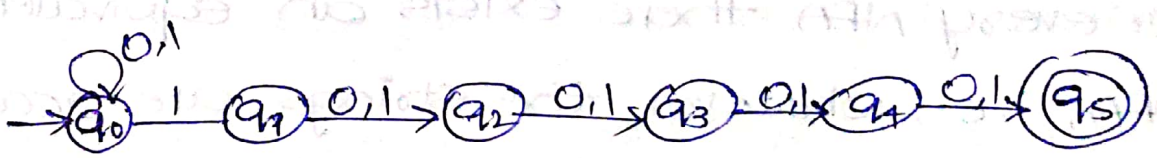
Accepted.

for babb



Rejected

→ Design NFA which accept all the strings with 0's and 1's such that the string should contain 5th symbol from the RHS is 1.



$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

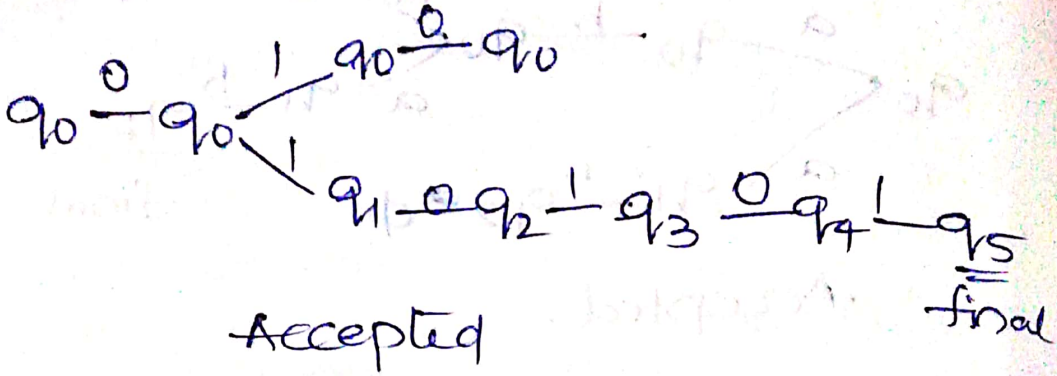
$\Sigma = \{0, 1\}$

$q_0 = q_0$

$F = \{q_5\}$

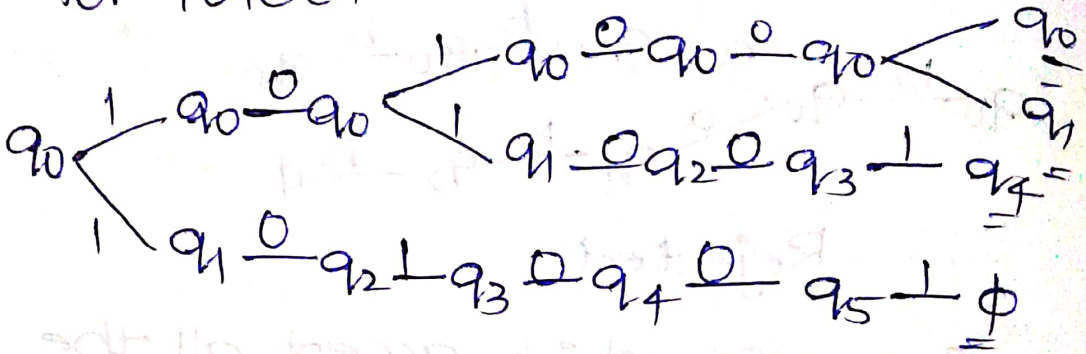
δ	0	1
q_0	q_0, q_1	q_0
q_1	q_2	q_2
q_2	q_3	q_3
q_3	q_4	q_4
q_4	q_5	q_5
q_5	ϕ	ϕ

for 01010|



Accepted

for 101001



Rejected

EQUIVALENCE of NFA TO DFA

For every NFA there exists an equivalent DFA, i.e., whatever the strings are accepted by NFA same set of strings should be accepted by its equivalent DFA

Same for rejecting:

Procedure to convert NFA to its equivalent DFA:

- If NFA represents $m = (Q, \Sigma, \delta, q_0, F)$ then its equivalent DFA can be constructed as follows $m' = (Q', \Sigma, \delta', q_0, F')$

(i) Q' can be computed as 2^Q.

Ex:- $Q = \{q_0, q_1, q_2\}$ then

$$Q' = \{\phi, q_0, q_1, q_2, q_1q_2, q_0q_1, q_0q_2, q_0q_1q_2\}$$

(ii) Σ is same as in given NFA

(iii) δ' can be computed as $\delta'(q_0, a) = \delta(q_0, a)$ for any state q_0 but $\delta(q_0q_1, a)$

$$\delta'(q_0q_1, a) = \delta(q_0, a) \cup \delta(q_1, a)$$

(iv) Initial state is same as in given NFA.

(v) F' can be computed as:
in Q' what are the states containing final states in F

Convert the following NFA to DFA:-

NFA

$Q = \{q_0, q_1\}$

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

δ	a	b
q_0	q_0, q_1	q_0
q_1	ϕ	ϕ

$q_0 = q_0$

$F = \{q_1\}$

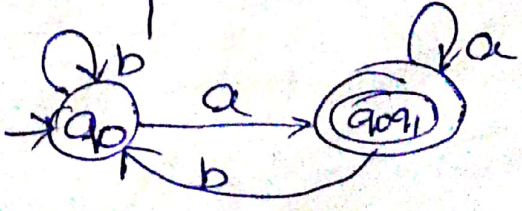
DFA

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

$Q' = \{\phi, q_0, q_1, q_0q_1\}$

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

δ'	a	b
$\rightarrow q_0$	q_0q_1	q_0
q_0q_1	q_0q_1	q_0



$q_0 = q_0$

$F = \{q_0q_1, q_1\}$

→ If $m = (\{p, q, r, s\}, \{0, 1\}, \delta, p, \{s\})$ represents an NFA and δ is given as

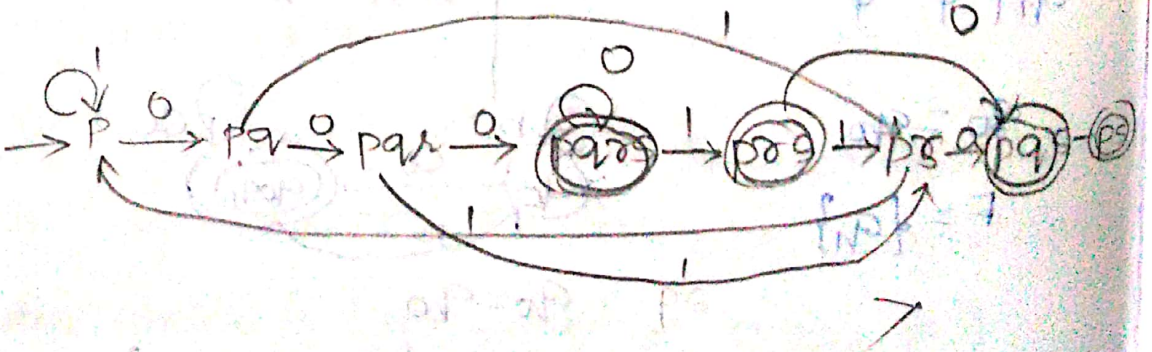
δ	0	1
→ p	p, q	p
q	r	r
R	s	ϕ
(s)	s	s

→ Its equivalent DFA $m' = (Q', \Sigma, \delta', q_0, F')$ can be constructed as follows

$Q' = \{\phi, p, q, r, s, pq, pr, ps, qr, qs, rs, pqr, qrs, rsp, pqs, pqrs\}$

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

δ'	0	1
→[p]	[pq]	[p]
[pq]	[pqr]	[ps]
[pqr]	[pqrs]	[ps]
[ps]	[pqs]	[p]
[pqs]	[pqrs]	[prs]
[prs]	[pqs]	[ps]
[ps]	[pqs]	[ps]
[pqrs]	[pqrs]	[prs]



5|| NFA WITH NULL TRANSITIONS

(OR)

NFA WITH ϵ-TRANSITIONS

- This is same as NFA except we are using a special input symbol called "ϵ" (epsilon) using this symbol machiene can move from one state to the other state with out reading any i/p symbol.

→ This is also analytically defined as

$$M = (Q, \Sigma, \hat{\delta}, q_0, F')$$

where Q = set of ~~finite~~ interal states

Σ = input symbols

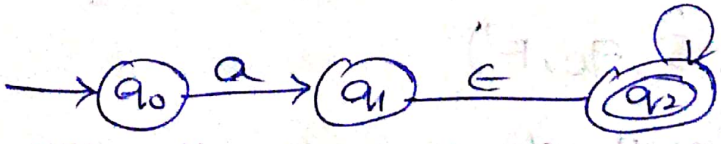
$\hat{\delta}$ = transition function.

$$\hat{\delta} : \Sigma \cup \{\epsilon\} \rightarrow 2^Q$$

q_0 - Initial states

F' - Final state

Ex:- The following NFA with null transition accept i/p strings with a's and b's such that string contains single a or a follo -wed by any no. of b's.



→ If a language L is accepted by NFA with null transitions then L is also accepted by NFA with out NULL transitions.

Null (ε) -closure :-

ε-closure of a state is defined as (ε-closure(q)). A set of all vertices p such that there is a path from q to p with label "ε" (exclude by itself).

For the above example

ϵ-closure $(q_0) = \{q_0\}$

ϵ-closure $(q_1) = \{q_1, q_2\}$

ϵ-closure $(q_2) = \{q_2\}$.

Procedure to convert NFA with ε-transitions to NFA without ε-transitions

→ Let $M = (Q, \Sigma, \delta, q_0, F)$ represents NFA with null transitions, there exists an equivalent NFA without null transitions $M' = (Q, \Sigma, \hat{\delta}, q_0, F')$

step(i): Find null-closure of all states in the given design.

(ii) Calculate $\hat{\delta}$ (extended transition fn)

Using the following conversion formulae

(i) $\hat{\delta}(q_0, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), a))$

(ii) $\hat{\delta}(q, \epsilon) = \epsilon\text{-closure}(q)$

where $q \in Q$

$a \in \Sigma$

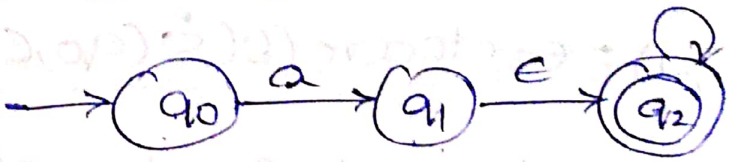
(iii) F' is a set of all states whose null closure contians a final state in F.

→ convert the following NFA with NULL transition to without ϵ-transitions

(or)

Elemenate ϵ-transitions from the following NFA.

Given that



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

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

q_0 is state → its of transition

δ	a	b	ϵ
→ q_0	q_1	q	ϕ
q_1	ϕ	ϕ	q_2
(q_2)	ϕ	q_2	ϕ

$q_0 = q_0$

$F = \{q_2\}$

→ We can construct NFA without null transitions as follows.

(i) ϵ-closure $(q_0) = q_0$

ϵ-closure $(q_1) = q_1, q_2$

ϵ-closure $(q_2) = q_2$

(ii) Calculate $\hat{\delta}$.

$$\hat{\delta}(q,a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q,\epsilon), a))$$

$$\hat{\delta}(q,\epsilon) = \epsilon\text{-closure}(q)$$

where $q \in Q$

$a \in \Sigma$.

$$\hat{\delta}(q_0,a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0,\epsilon), a))$$

$$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_0), a))$$

$$= \epsilon\text{-closure}(\delta(q_0,a))$$

$$= \epsilon\text{-closure}(q_1)$$

$$= q_1, q_2$$

→ $\hat{\delta}(q_0,b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0,\epsilon), b))$

$$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_0), b))$$

$$= \epsilon\text{-closure}(\delta(q_0,b))$$

$= \epsilon\text{-closure}(\phi)$

$= \phi$

$\rightarrow \hat{\delta}(q_1, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), a)$

$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_1), a)$

$= \epsilon\text{-closure}(\delta(q_1, q_2), a)$

$= \epsilon\text{-closure}(\delta(q_1, a) \cup \delta(q_2, a))$

$= \epsilon\text{-closure}(\phi)$

$= \phi$

$\rightarrow \hat{\delta}(q_1, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), b)$

$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_1), b)$

$= \epsilon\text{-closure}(\delta(q_1, b), q_2, b))$

$= \epsilon\text{-closure}(q_2)$

$= q_2$

$\rightarrow \hat{\delta}(q_2, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2, \epsilon), a))$

$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_2), a))$

$= \epsilon\text{-closure}(\delta(q_2, a))$

$= \epsilon\text{-closure}(\phi)$

$= \phi$

$\rightarrow \hat{\delta}(q_2, b) = \epsilon\text{-closure}(\delta(q_2, b))$

$= \epsilon\text{-closure}(q_2)$

$= q_2$

(iii) Calculate F'.

$F' = \{q_1, q_2\}$ are final states

Since q_2 is the final state in e-transition NFA so

e-closure of $q_2 = q_2$

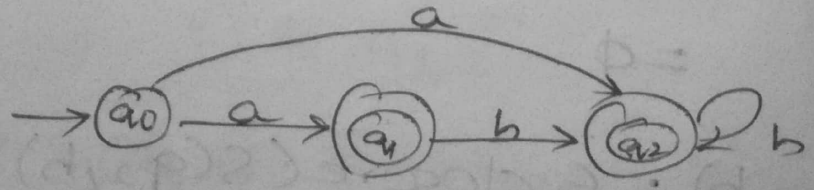
" " $q_1 = q_1 q_2$

q_2 is present in both the null closures of q_1 & q_2 so q_1 & q_2 are the final states in NFA.

$\hat{\delta}(q_0, a)$

$=$

	a	b
→ q_0	q_1, q_2	ϕ
(*) q_1	ϕ	q_2
(*) q_2	ϕ	q_2



→ Convert the above NFA to DFA.

Its equivalent DFA

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

$Q' = \{\phi, q_0, q_1, q_2, q_0q_1, q_1q_2, q_0q_2, q_0q_1q_2\}$

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

δ'	a	b
→ q_0		

δ'	a	b
→ q_0	q_1q_2	ϕ
(q_1q_2)	ϕ	q_2
(q_2)	ϕ	q_2

$F = \{q_1q_2, q_2\}$ $q_0 = q_0$.