Unit-I

Finite Automata

Basics of Formal languages

Alphabet: Alphabet is a finite and non-empty set of input symbols. It can be denoted by Σ.

String: A String can be defined as finite sequence of symbols that are choosen from alphabet Σ. It can be denoted by ω

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

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

$\omega = \{0, 1, 00, 01, 10, \ldots \epsilon\}$

where ϵ is a empty string or null string. The length of ϵ is 0.

Operations on Strings:-

1) Length of a string
2) Prefix of a string
3) Suffix of a string
4) Proper Prefix or Proper suffix of a string
5) Substring of a string
6) Reverse of a string

7) Palindrome of a string
8) Union of two strings
9) Concatenation of two strings

Length of a string:-

It is defined as the no. of symbols in the given string. It can be denoted by $|w|$

Eg: $w = ab$ $|w| = 2$

$w = abcd$ $|w| = 4$

$w = \epsilon$ $|w| = 0$

Prefix of a string:

It is any number of leading symbols in the given string.

eg: $w = abcd$

Prefix $= \{\epsilon, a, ab, abc, abcd\}$

Suffix of a string:-

It is any number of trailing symbols in the given string.

eg: $w = abcd$

Suffix $= \{\epsilon, d, cd, bcd, abcd\}$

Proper Prefixes and Proper suffixes:-

A Prefix or suffix of a string other than the given string is known as Proper Prefixes and Proper suffixes.

Eg: $w = abcd$

Proper Prefix = $\{\epsilon, a, ab, abc\}$

Proper Suffix = $\{\epsilon, d, cd, bcd\}$

Note: If the string length is 'n', how many no. of Prefixes, suffixes and either Proper Prefixes or Proper suffixes

Prefixes = $n+1$

suffixes = $n+1$

Proper Prefix or Proper suffix = n

Substring:

Every Prefix and suffix of a string 'w' is a substring of the string.

Eg:- $w = ab$

substring = ϵ, a, b, ab

$w = abcd$

substring = $\{\epsilon, a, b, c, d, ab, bc, cd, abc, bcd, abcd\}$

If the string length is n, how no. of substrings are possible?

i.e $\frac{n(n+1)}{2} + 1$

Reverse of a String:-

It is obtained by writing the symbols of the given string in reverse order. It can be denoted by 'w^R'.

Eg: $w = abc$ $w^R = cba$

Palindrome of a string:-

A string is said to be Palindrome of a string if & only if $w = w^R$.

Eg: $w = aba$ $w^R = aba$

$\therefore$ aba is a Palindrome

Union of two strings:-

It is denoted by $w_1 \cup w_2$ (or) $w_1 + w_2$ (or) $w_1 | w_2$.

Eg: $w_1 = ab$ $w_2 = cd$

$w_1 \cup w_2 = \{ab, cd\}$

Concatenation of two strings:-

It is denoted by $w_1 \cdot w_2$ i.e string1 followed by string 2.

Eg:- $w_1 = ab$ $w_2 = cd$

$w_1 \cdot w_2 = abcd$

Language :-

A language is the set of all strings that are choosen from the alphabet Σ. It can defined as the $L \subseteq \Sigma^*$ where L is a language

Eg: $\Sigma = \{0\}$

$\Sigma^* = \{\epsilon, 0, 00, 000, ----\}$

$L = \{\epsilon, 0, 00, 000, ---\}$

Eg ① Describe the language consist of any no. of a's over an alphabet $\Sigma = a$

$L = \{\epsilon, a, aa, aaa, ----\}$

$L = \{a^n \mid n \geq 0\}$

Eg ② Describe the language consist of equal no. of 0's followed by equal no. of 1's over

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

$L = \{\epsilon, 01, 0011, 000111, ---\}$

$L = \{0^n 1^n \mid n \geq 0\}$

A language can be classified into two types

① Finite language

② Infinite language

Finite language:-

A language is said to be finite language if it contains a finite no. of strings.

eg: $L = \{ |w| \leq 2 \mid w \in (a,b)^* \}$ length 2

$L = \{\epsilon, a, b, aa, ab, bb, ba\}$

Infinite Language:- $L_3 = \{aaa, aab, aba, abb, baa, bba, bab, bbb\}$

A language is said to be infinite if it contains infinite no. of strings.

eg: $L = \{a^n \mid n \geq 0\}$ starts with 'a'

$L = \{0^n 1^n \mid n \geq 0\}$

Operations on languages:

1. Concatenation of two languages – $A \times B = \{(1,2)(1,3),(2,2),(2,3)\}$
2. Union of two languages – $A \cup B = \{1,2,3\}$
3. Intersection of two languages – $A \cap B = \{2\}$
4. Difference of two languages – $A - B = \{1\}$, $B - A = \{3\}$
5. Compliment of a language (Powerset) $= 2^A = \{\{1\},\{2\},\{1,2\},\phi\}$
6. Kleene closure of a language $[L^*]$ $2^B = \{\{2\},\{3\},\{2,3\},\phi\}$
7. Possitive closure of a language (L^+)

eg:- $A = \{1,2\}$ $B = \{2,3\}$

eg-2: $A = \{2,3\}$; $B = \{3,4\}$

Kleene closure:-

It consists of set of all strings on length '0' or more no. of occurrences. It can be denoted by L^*

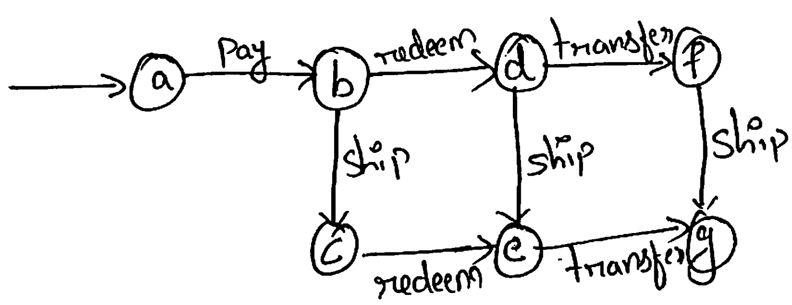
eg: $L^* = L^0 \cup L^1 \cup L^2 \cup \ldots$

Positive closure:-

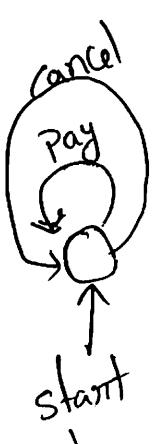
It ~~const~~ contains set of all strings except the empty string (i.e 1 or more occurrences). It can be represented by L^+.

$L^+ = L^* - \{\epsilon\}$ or $L^+ = L^1 \cup L^2 \cup L^3 \cup \ldots$

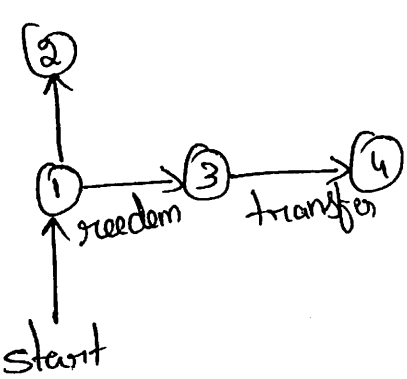
Finite automata representing a customer, a store & a bank



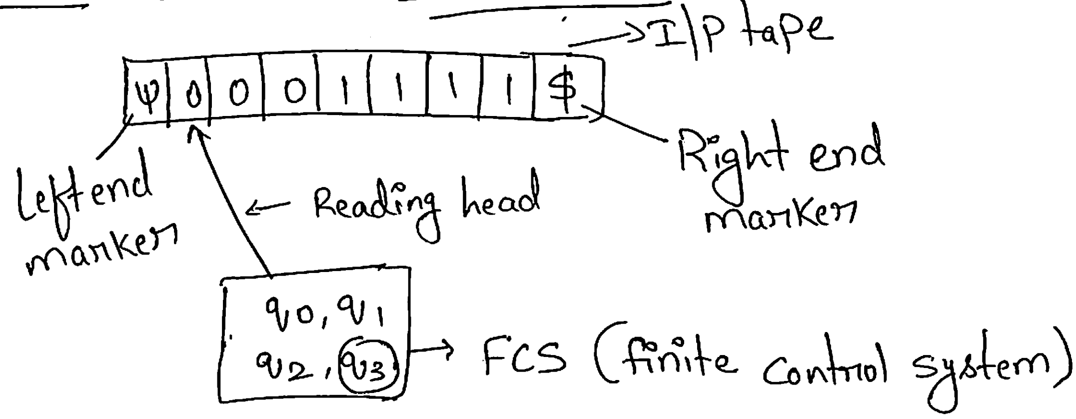
a) Store



b) customer



Model & Behaviour of a FSM or FA



A FA Consists of a finite set of states and a set of transitions from one state to another state that occurs on the input symbols (Choosen from an alphabet 'Σ'.) & gives the output

The read head is placed at the 1st cell immediate right to the left end marker and reads i/p symbol one by one until it gets right end marker ($) & checks the Current state of the FA if it belongs to final state that represents the given string is valid string otherwise it is not accepted.

$$\delta(q_0, w) \in F$$

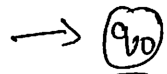
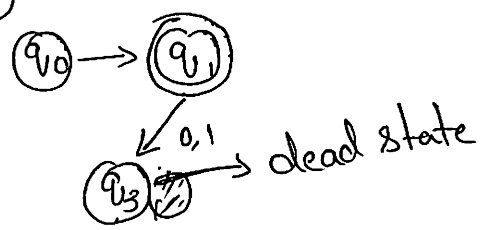
FA is mainly represented in 2ways

1. Transition diagram
2. Transition Table

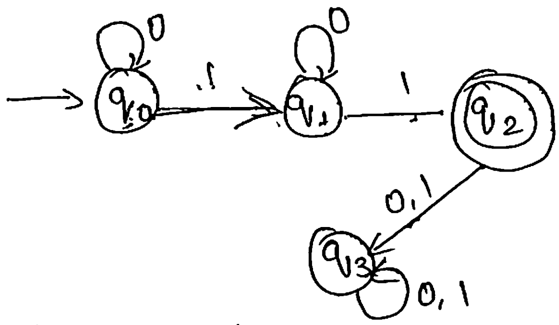
Transition Diagram:

In transition diagram, set of vertices & edges are Present. The vertices of a transition diagram are called as states of FA & edges are called as i/p symbols of the FA

There are different types of states in FA

1. Initial state 
2. Final state — 
3. Dead state — 

eg:-



<u>Transition Table</u> :-

It consists of rows and columns
rows → no. of state
Columns → set of i/p symbols

δ	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_1	q_2
$*q_2$	q_3	q_3
q_3	q_3	q_3

<u>Definition of FA</u>:-

The FA can be described as five tuple notation.

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

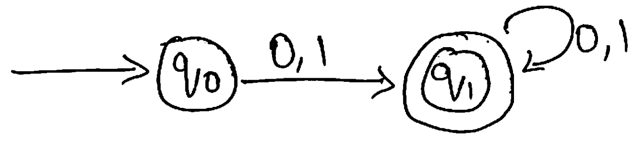
where
Q – Finite & non-empty set of states.
Σ – Sequence of symbols from an alphabet.
δ – Transition or mapping function
q_0 – Initial state
F – Final state

$$\delta : Q \times \Sigma \longrightarrow \underset{\text{(some state of Q)}}{Q}$$

$q_0 \in Q$
$F \in Q$

→ A Language accepted by finite automata then it is 'regular set'. It is represented by
$L(M) = \{ x \mid \delta(q_0, x) \text{ is in } F \}$

Eg:-



$Q = \{q_0, q_1\}$

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

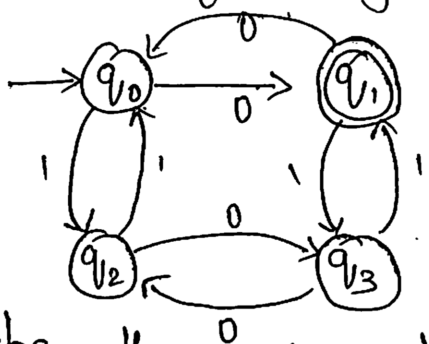
q_0 = initial state

$F = \{q_1\}$

$\delta : Q \times \Sigma \to Q$

δ	0	1
$\rightarrow q_0$	q_1	q_1
(or) $*q_1$	q_1	q_1

→ Consider the following transition diagram



i) Describe the given transaction diagram using 5 tuple notation

ii) Draw the transn table for the given transn dig

iii) Check whether the following strings are accepted or not

a) 01010110

b) 00110

c) 101011010

Ans:- i) $M = (Q, \Sigma, q_0, \delta, F)$

$= (\{q_0, q_1, q_2, q_3\}, \{0,1\}, \{q_0\}, \delta, \{q_1\})$

$\delta(q_0, 0) = q_1$

$\delta(q_0, 1) = q_2$

$\delta(q_1, 0) = q_0$

$\delta(q_1, 1) = q_3$

$\delta(q_2, 0) = q_3$

$\delta(q_2, 1) = q_0$

$\delta(q_3, 0) = q_2$

$\delta(q_3, 1) = q_1$

ii)

	δ	0	1
$\rightarrow$	q_0	q_1	q_2
*	q_1	q_0	q_3
	q_2	q_3	q_0
	q_3	q_2	q_1

iii) a) $\delta(q_0, 01010110)$

$\delta(q_1, 1010110)$

$\delta(q_3, 010110)$

$\delta(q_2, 10110)$

$\delta(q_0, 0110)$

$\delta(q_1, 110)$

$\delta(q_3, 10)$

$\delta(q_1, 0) = q_0 \in F$

$\therefore$ It is an invalid string

b) 00110

$\delta(q_0, 00110)$

$\delta(q_1, 0110)$

$\delta(q_0, 110)$

$\delta(q_2, 10)$

$\delta(q_0, 0) = q_1 \in F$

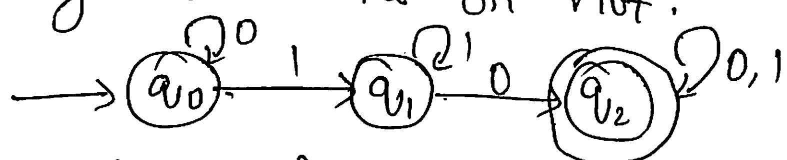
(valid string)

Acceptance of string:-

A string 'x' is accepted by finite automata $M = (Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = P \in F$

$$\delta(q, aw) = \delta(\hat{\delta}(q, a), w)$$

Find whether the string 1011 is accepted by following automata or not.



Ans: $$\delta(q_0, 1011) = \cancel{\delta(\hat{\delta}(q_0, 1))}$$

$$= \delta(\hat{\delta}(q_0, 1), 011)$$

$$= \delta(q_1, 011)$$

$$= \delta(\hat{\delta}(q_1, 0), 11)$$

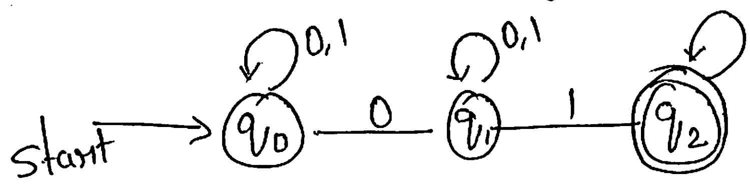
$$= \delta(q_2, 11)$$

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

$$= \delta(q_2, 1) = q_2 \in F$$

$$q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2 \xrightarrow{1} q_2 \xrightarrow{1} q_2 \in F \longrightarrow \text{accepted}$$

Eg:- Consider a string {w | x01y where x & y are sequence of 0's & 1's }



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

Check whether the given strings are accepted or not.

1) 100 — Not accepted
2) 11110 — not accepted
3) 11111001 — Accepted

Types of Finite Automata:

→ Deterministic FA (DFA)
→ Non-deterministic FA (NFA)

DFA:- A Finite Automata is called as a DFA if ~~there~~ there is only one path from the current state to the next state on every specific i/p symbol

***Note:** No. of states for $|w| = 2$ is $(n+2)$
" " " $|w| \geq 2$ is $(n+1)$
" " " $|w| \leq 2$ is $(n+2)$
"

A DFA 'M' can be described by using 5 tuple notations

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

Q – finite & non empty set of states

Σ – set of i/p symbols

q_0 – initial state

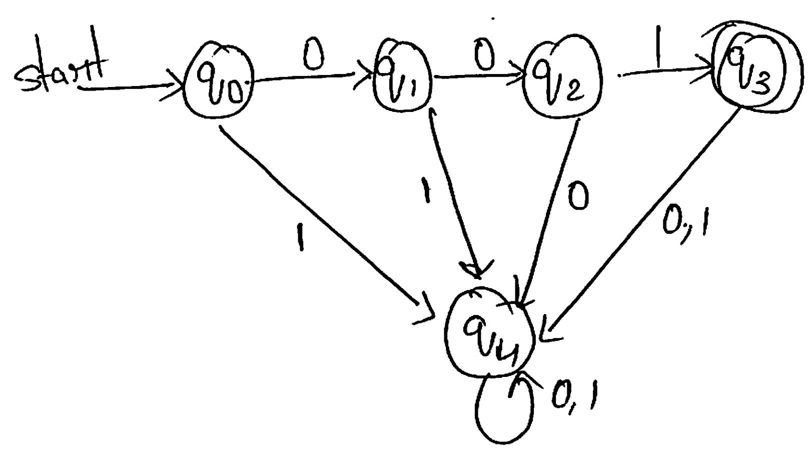
F – final state

δ – transition or mapping function

$\therefore \delta = Q \times \Sigma \longrightarrow Q$

where F is a set of final states $F \subseteq Q$

① Construct a DFA which accept the string 001 over an alphabet $\Sigma = \{0,1\}$

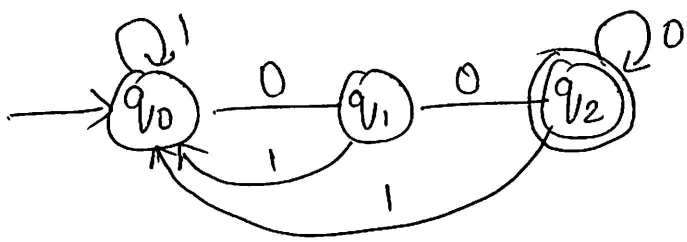


δ	0	1
$\rightarrow q_0$	q_1	q_4
q_1	q_2	q_4
q_2	q_4	q_3
$* q_3$	q_4	q_4
q_4	q_4	q_4

② Construct a DFA to accept the strings ends with 00 only over $\Sigma = \{0,1\}$

lets construct language for given i/p

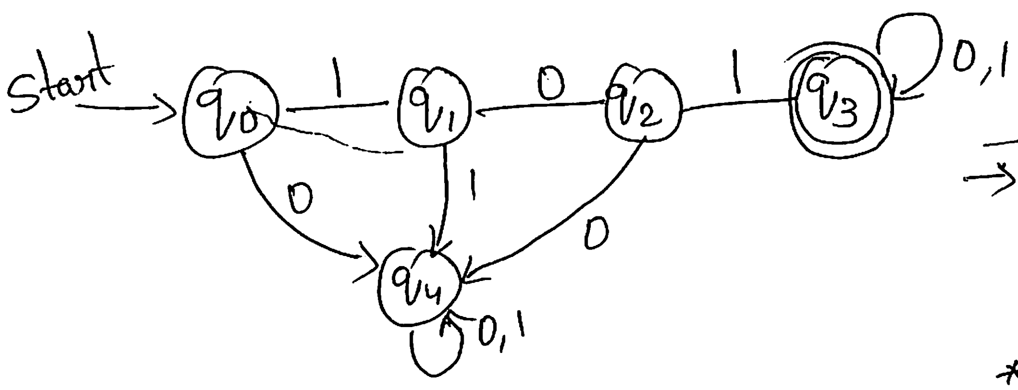
$L = \{00, 100, 0100, 10100, 101100, 110100, \dots\}$



δ	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_2	q_0
$*q_2$	q_2	q_0

(3) Construct a DFA which accepts set of binary strings begins with 101. $\Sigma = \{0,1\}$

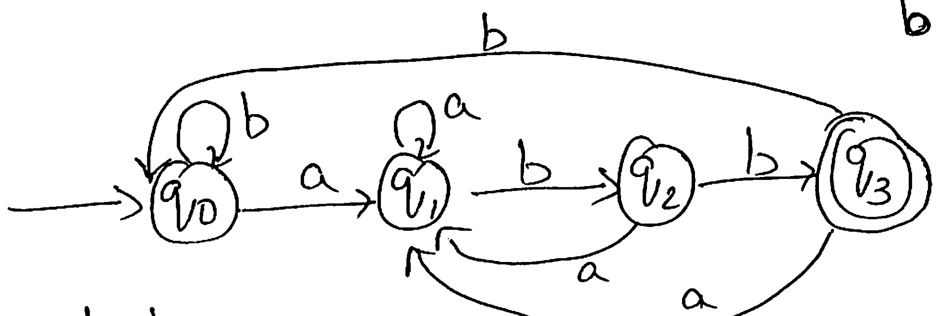
let $L = \{101, 1010, 10110, 101110, \dots\}$



δ	0	1
$\rightarrow q_0$	q_4	q_1
q_1	q_2	q_4
q_2	q_4	q_3
$*\ q_3$	q_3	q_3
q_4	q_4	q_4

(4) Construct a DFA to accept the string a's & b's ending with abb over $\Sigma = \{a,b\}$.

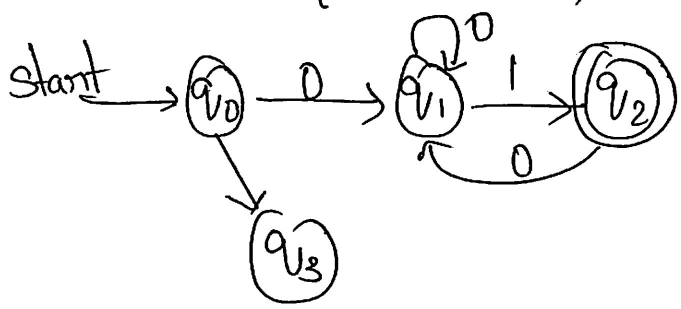
$\therefore L = \{abb, aabb, babb, ababb, bbabb, \dots\}$
babbbbabb



δ	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_3
$*\ q_3$	q_1	q_0

⑤ DFA that starts with '0' & ends with '1' $\Sigma=\{0,1\}$

let $L = \{01, 01101, 011, 0101, 001011, \ldots\}$



⑥ Construct a DFA which accepts the strings on $\Sigma = \{0,1\}$

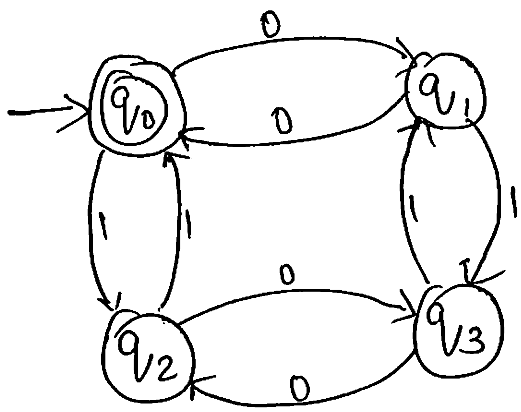
i) Even no. of 0's & even no. of 1's

ii) even no. of 0's & odd no. of 1's

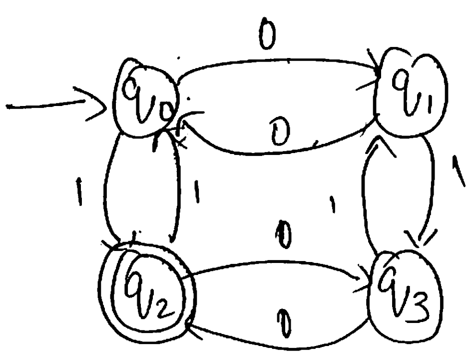
iii) odd no. of 0's & even no. of 1's

iv) odd no. of 0's & odd no. of 1's

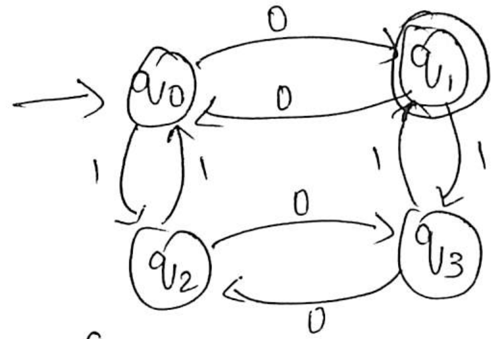
i) $L = \{0011, 1100, 0101, 1010, 1001, \ldots\}$



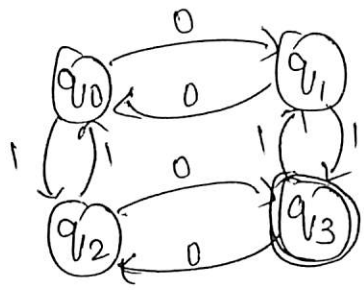
ii) $L = \{001, 100, 010, 00001, 10011, \ldots\}$



iii) L = { 011, 110, 101, 01100, 11000, - - - }



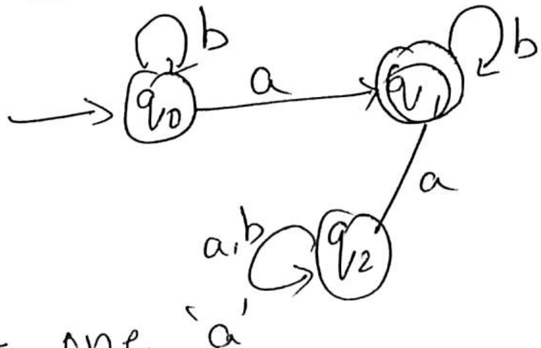
iv) L = { 01, 10, 000111, 111000, 101010, 010101, - - - }



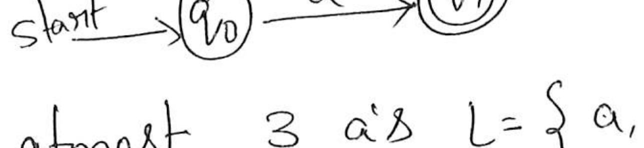
7) Construct a DFA to accept the string a's & b's having

i) Exactly one 'a'
ii) atleast one 'a'
iii) atmost 3 a's
iv) atmost 1 'a'

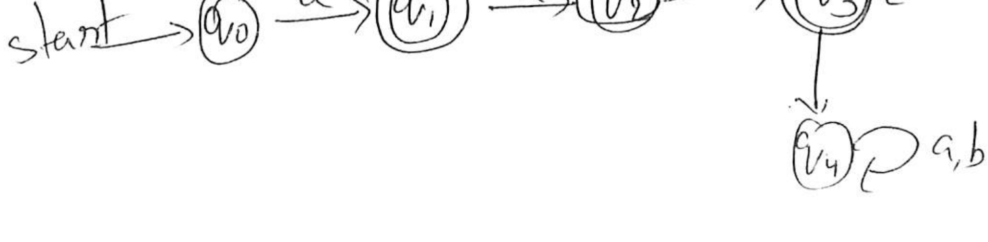
i) Exactly one 'a' : L = {a, abb, abbb, bab, bba, - - -}



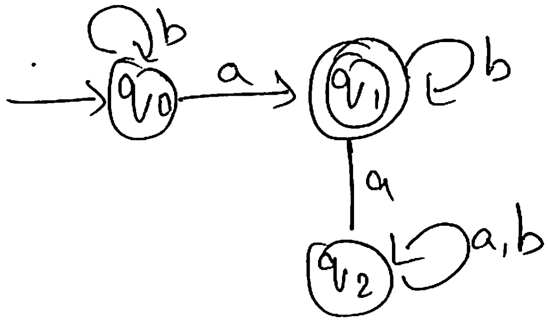
ii) atleast one 'a'



iii) atmost 3 a's L = { a, ab, ba, aab, aaab, baaab, - - }

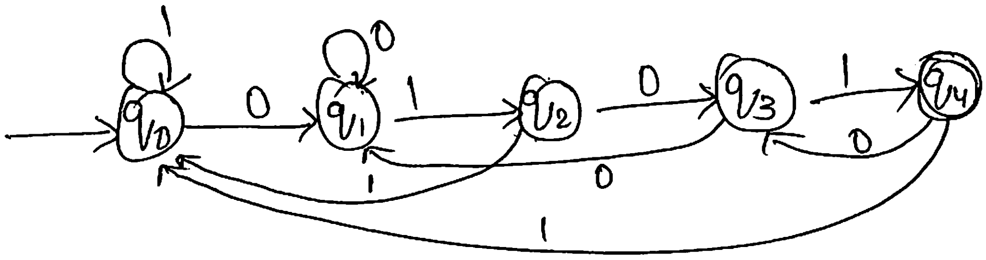


iv) atmost 1 'a' $L = \{a, ab, ba, abb, bab, bba, \ldots\}$



8) Construct a DFA which accepts strings ending with 0101

$L = \{0101, 00110101, 111001011 0101, 1010101, \ldots\}$



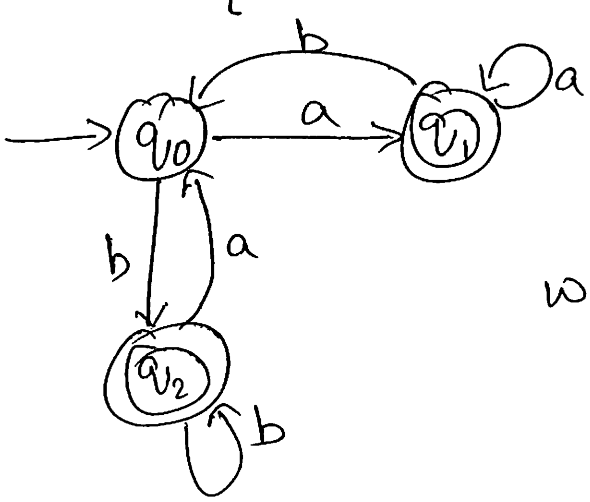
9) Construct a DFA for the following languages

i) $L = \{w \mid w$, w ~~is a~~ starts & ends with same symol, $w \in \{a,b\}^*\}$

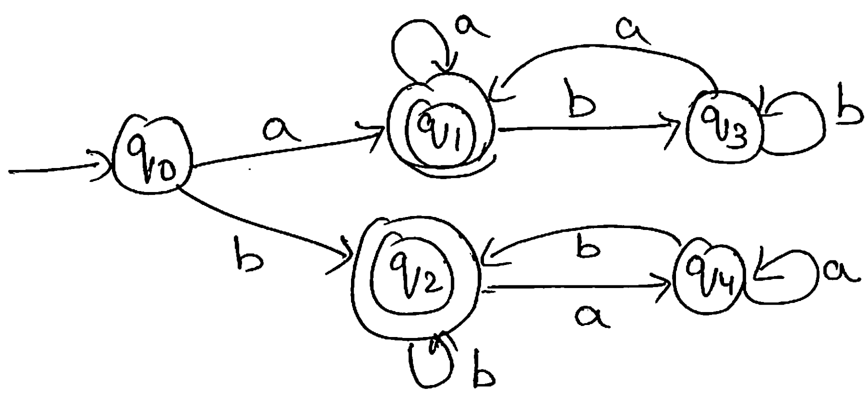
ii) $L = \{w \mid w$, w is a starts & ends with diff symbol, $w \in (a,b)^*\}$

<u>Ans:</u> i) $L = \{a, b, aba, bab, ababa, babbab, \ldots\}$

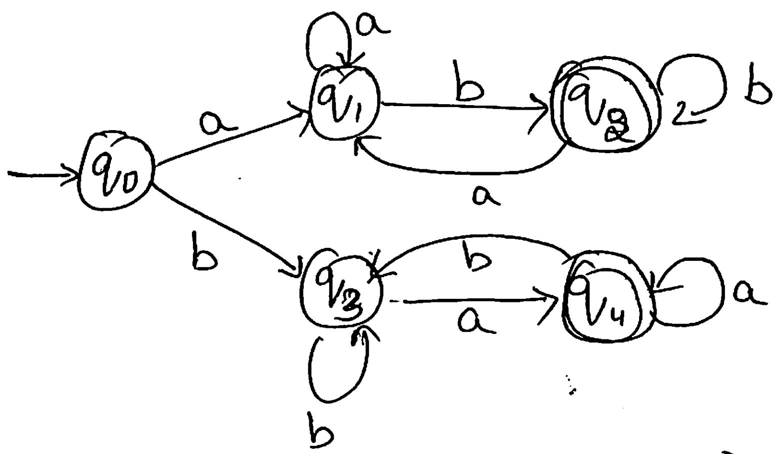
abbaa



where abba does not satisfy



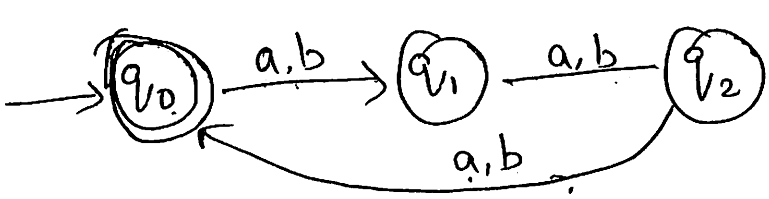
ii) L = {ab, ba, abb, aab, ababab, ...}



⟹ DFA for { w | |w| mod 3 = 0 } for {a,b}*

lets L = {aba} -- |aba| = 3 % 3 = 0 ✓
{ababa} |L| = 5 % 3 = 2 ✓

Possible remainders for MOD 3 are 0, 1, 2



Note: $|w| \cong 1 \bmod 3$ or $|w| \bmod 3 = 1$ are same

$|w| \bmod n = 0$

3 – 3 no. of states

4 – 4 " "

n – n no. of states

→ For $|w| \bmod 3 = 1$ q_1 will be finite state

For $|w| \bmod 3 = 2$ q_2 will be finite state

Generalized representation for $|w| \bmod 3$

∴ remainders are 0, 1, 2

q_0 holds '0'
q_1 holds '1'
q_2 holds '2'

⇒ $Q = \{q_0, q_1, q_2\}$
$\Sigma = \{0, \ldots 9\}$
$q_0 = q_0$
$F = q_0$

Transition table:

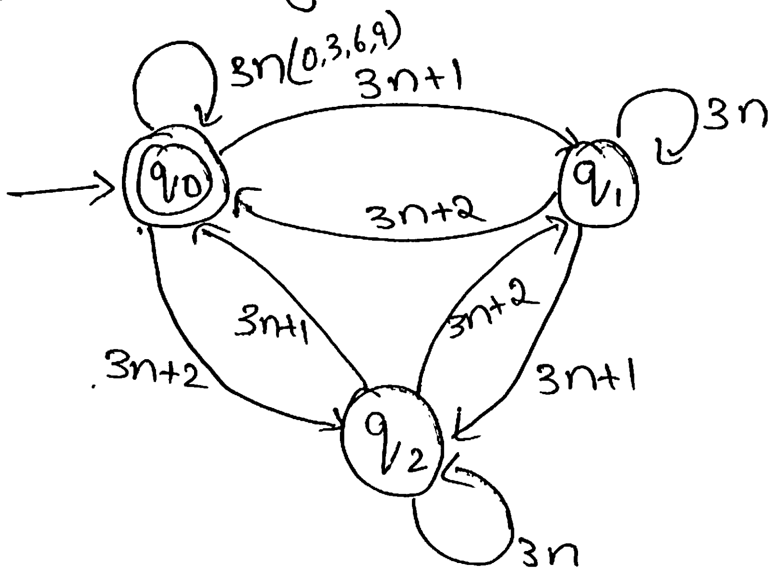
	δ	3n 0,3,6,9	3n+1 1,4,7	3n+2 2,5,8
0	$\rightarrow q_0^*$	q_0	q_1	q_2
1	q_1	q_1	q_2	q_0
2	q_2	q_2	q_0	q_1

row 1:
$3/3 = 0$
$01/3 = 1$
$2/3 = 2$

row 2:
$10/3 = 1$
$11/3 = 2$
$12/3 = 0$

row 3:
$20/3 = 2$
$21/3 = 0$
$22/3 = 1$

Transition diagram:



$q_0(256)$

$3)\overline{2561}$

Verify the acceptance of string

eg:- 14712

$\delta(q_0, 14712)$

$\delta(q_1, 4712)$

$\delta(q_2, 712)$

$\delta(q_0, 12)$

$\delta(q_1, 2) = q_0 \in F$

$\therefore$ string accepted

eg - 421

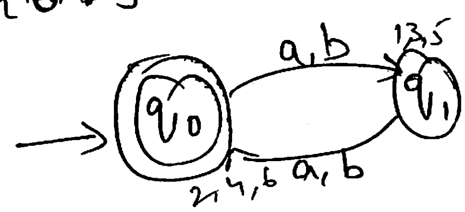
$\delta(q_0, 421)$

$\delta(q_1, 21)$

$\delta(q_0, 1) = q_1 \notin F$

string not accepted

DFA for $|w| \bmod 2$ for $\{a, b\}^*$



Remainder for mod 2 are 0 & 1

$q_0 = 0 \qquad q_1 = 1$

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

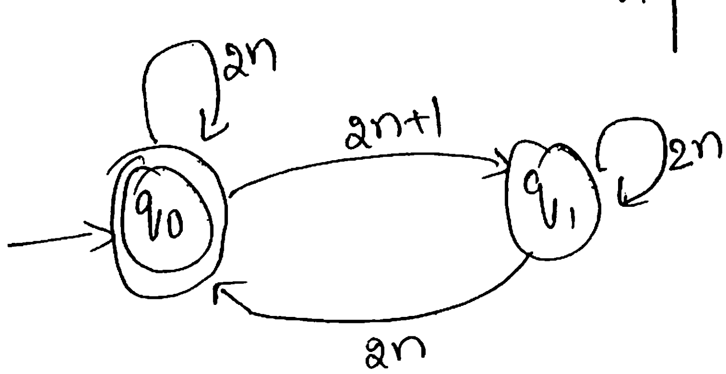
$\Sigma = \{0 - - 9\}$

$q_0 = q_0$

$F = q_0$

Table:

	δ	2n 0,2,4,6,8	2n+1 1,3,5,7,9
0	$\rightarrow q_0^*$	q_0	q_1
1	q_1	q_0	q_1



Acceptance of string

$\delta(q_0, 256)$

$\delta(q_0, 56)$

$\delta(q_1, 6) = q_0 \in F$

string accepted

$\delta(q_0, 5317)$

$\delta(q_1, 317)$

$\delta(q_1, 17)$

$\delta(q_1, 7) = q_1 \notin F$

$\therefore$ string not accepted

Construct DFA for the language

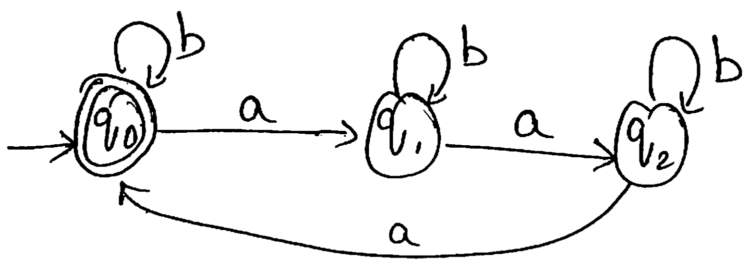
i) $L = \{w \mid n_a(w) \bmod 3 = 0, \ w \in (a,b)^*\}$

ii) $L = \{w \mid n_a(w) \bmod 3 = 1, \ w \in (a,b)^*\}$

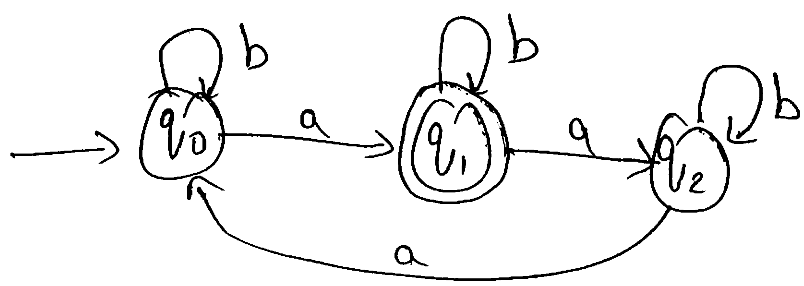
iii) $L = \{w \mid n_a(w) \bmod 3 = 2, \ w \in (a,b)^*\}$

i) $L = \{aaabbb, abbaa, bbaaa, \ldots\}$

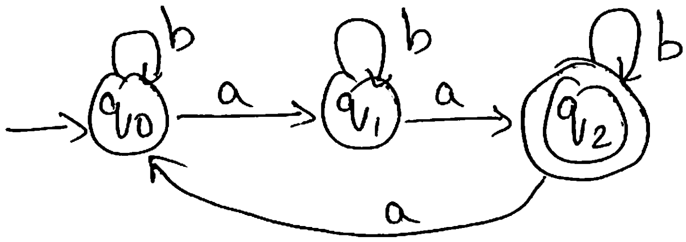
$q_0 = 0, \quad q_1 = 1, \quad q_2 = 2$



ii) $L = \{abb, aabbbaa, aaaab, \ldots\}$



iii) $L = \{aab, aba, abbabaaa - - -\}$

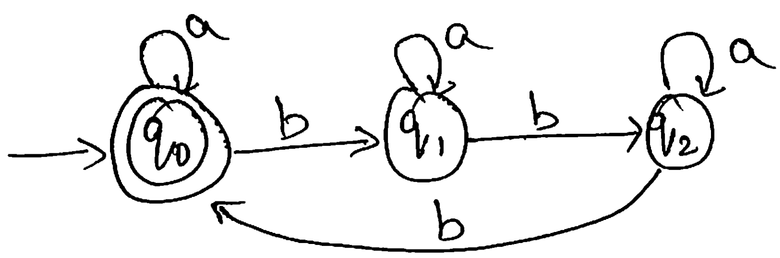


→ iv) $L = \{w \mid n_b(w) \bmod 3 = 0, \ w \in (a,b)^*\}$

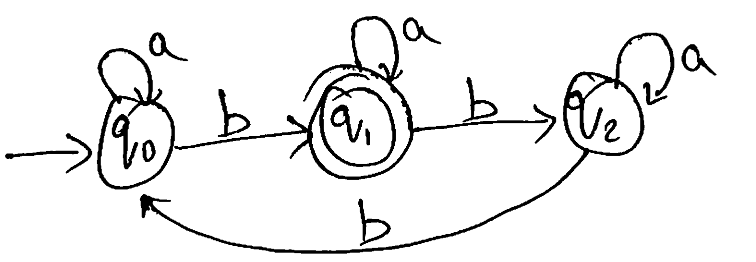
v) $L = \{w \mid n_b(w) \bmod 3 = 1, \ w \in (a,b)^*\}$

vi) $L = \{w \mid n_b(w) \bmod 3 = 2, \ w \in (a,b)^*\}$

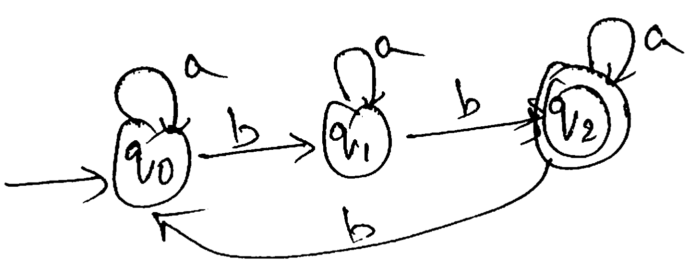
iv) $L = \{a, bbb, abbb, aabbb, abbab, - - -\}$



v) $L = \{b, ab, ba, abbbb, bbaabb, - - -\}$



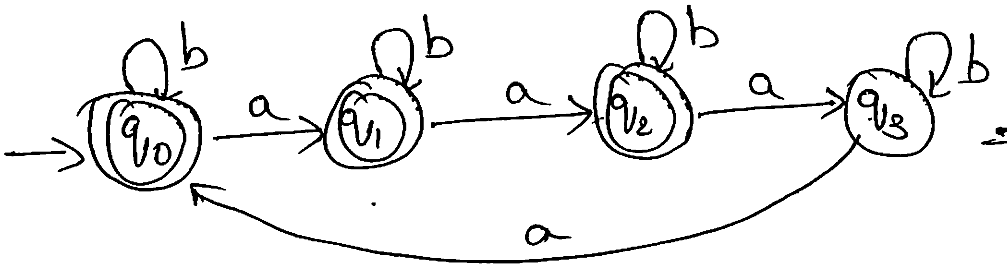
vi) $L = \{bb, abb, bba, abababbab, bbaabbab, - - -\}$



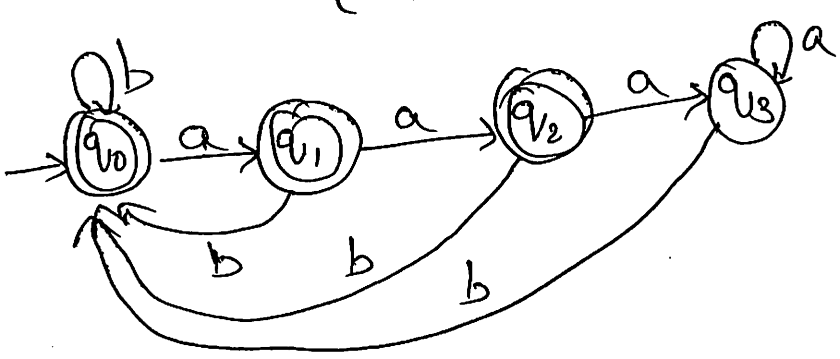
→ To construct DFA for the foll lang.

i) $L = \{ w \mid |w|$ ~~mod 3 = 0~~ does not ends with 3 a's, $(a,b)^* \}$

$L = \{ a, b, aa, baa, ababaa, bbaaaba, \ldots \}$

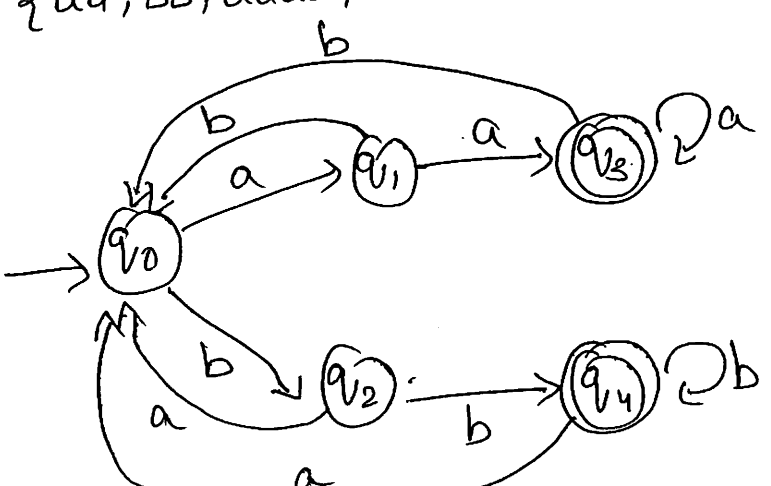


(or)

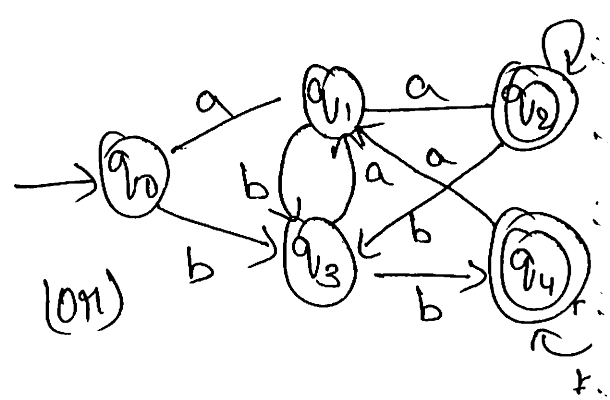


→ Construct a DFA to accept the strings in which each string ends with either aa or bb $\Sigma = \{a,b\}$

$L = \{ aa, bb, aaaa, bbbbb, baa, abb, ababaa, \ldots \}$

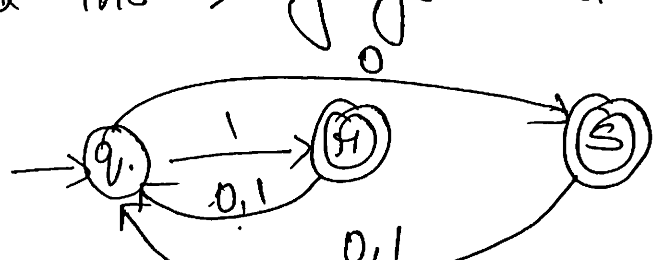


(or)

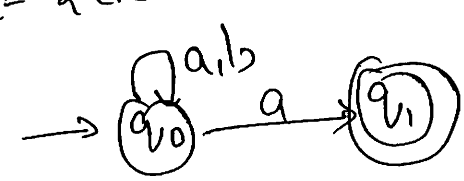


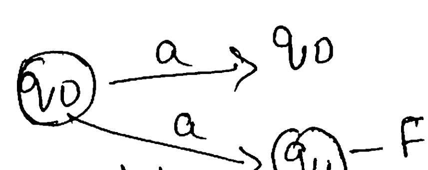
Assignment questions:

- → DFA for even no. of 0's followed by single '1' over $\Sigma = \{0,1\}$.
- → DFA for strings not having more than two a's over $\Sigma = \{a,b\}$.
- → DFA for accepting all the strings of $\{L = 0^m 1^n \mid m \geq 0 \text{ \& } n \geq 1\}$
- → DFA over $\Sigma = \{a,b\}$ for (i) $(ab)^n$ with $n \geq 0$
  ii) $(ab)^n$ with $n \geq 1$
- → $L = \{w \mid w$ does not contains the substring $ab\}$
- → $L = \{w \mid w$ contains niether the substring ab or ba
- → $L = \{w \mid w$ is any string that doesn't contain exactly two a's $\}$
- → $L = \{w \mid w$ is any string except a & $b\}$
- → All strings containing not more than three 0's
- → All strings that has at least two occurrences of '1' between any two occurrences of 0.
- → Find the language accepted by following FA.



$L = \{$ ends with $a\}$





∴ from 1 state on 1 i/p you might go to many states or no state

Non Deterministic finite Automata : (NFA) :

A finite automata is called as NFA if there exists one or more transitions from a state on the same i/p symbol.

It is not compulsory that all states in NFA have to consume all the i/p symbols in 'Σ'.

It can be described as 5 tuple notation where

$M = (Q, \Sigma, \delta, q_0, F)$, $\delta = Q \times \Sigma \rightarrow 2^Q$.

'δ' is a Cartesian Product of states $\times$ symbols giving a state in power set of Q.

eg: $P \xrightarrow{a} q_1$ (or) $P \xrightarrow{a} q_2$ (or) $P \xrightarrow{a} q_3$

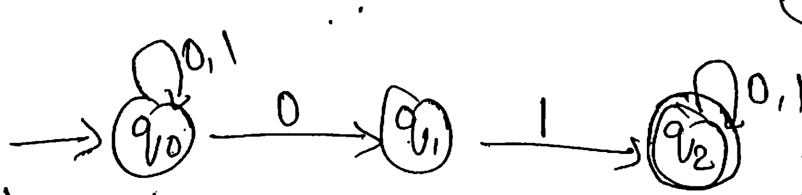
$P \Rightarrow \{q_1, q_2, q_3\}$

$\Rightarrow \{\emptyset\} \{q_1\} \{q_2\} \{q_3\} \{q_1, q_2\} \{q_1, q_3\} \{q_2, q_3\} \{q_1, q_2 ...$

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

$= \{\emptyset, \{q_0\}, \{q_1\}, \{q_0, q_1\}\}$

Ex-1 let $w = x01y$ where x & y are seq of 0's & 1's



δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	—	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$

i/p - 000101

$L = \{110101, 10010 - - - \}$

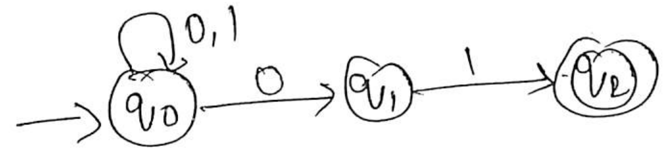
0011

$q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_0$

$\xrightarrow{0} q_1 \xrightarrow{0} \emptyset \times$ $\quad q_1 \xrightarrow{1} q_2 \xrightarrow{1} q_2 - F$

U-1

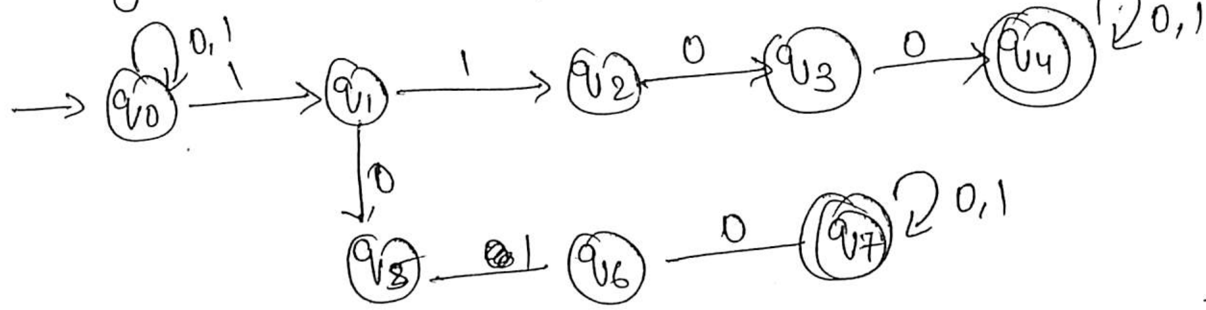
Ex-2: Construct NFA for $w = X01$ $\Sigma = \{0,1\}^*$



δ	0	1
q_0	q_0, q_1	q_0
q_1	–	q_2
q_2	–	–

→ i/p – 0101

Ex-3: Construct an NFA which accept set of strings contains 1100 or 1010 as substring



i/p – 011000

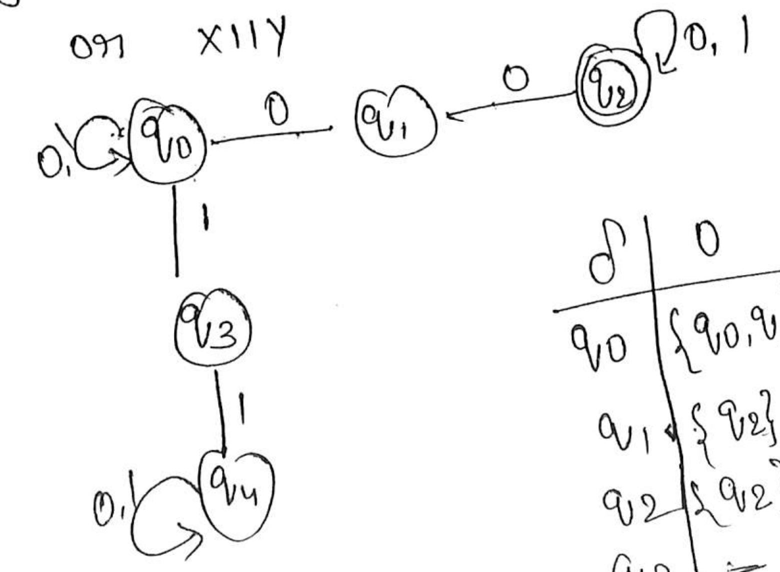
$q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_5 \xrightarrow{0} \times$

$\xrightarrow{1} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_3 \xrightarrow{0} q_4 \xrightarrow{0} q_4$ – F ✓

$q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \times$

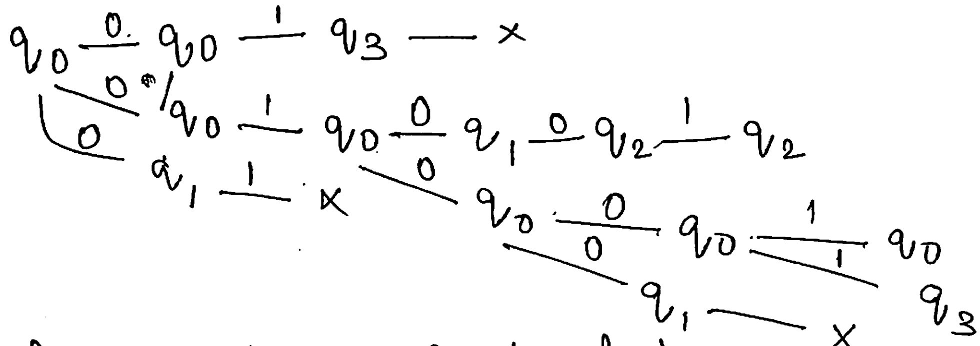
Ex-4: String accepts two consecutive 0's or 1's.

i.e X00Y or X11Y



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

i/p - 01001



$\{q_2, q_0, q_3\}$

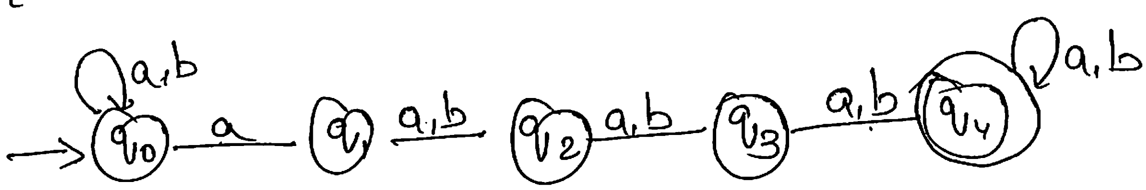
As q_2 is a final state where string can be accep[ted]

$\therefore q_0 - q_0 - q_0 - q_1 - q_2 - q_2$

i/p - 010011

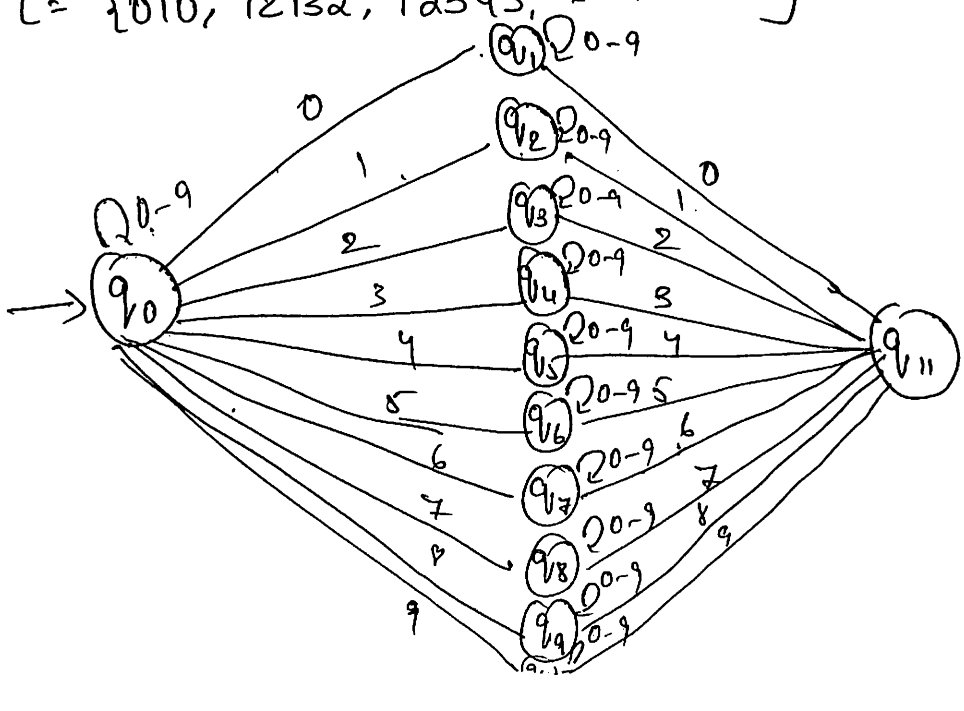
<u>Ex-5:</u> NFA for strings in which the 4th symbol from the right end is always 'a' over $\Sigma = \{a, b\}$

$L = \{abbb, aaaa, abab, abaabb\}$



<u>Ex-6</u> NFA for $\Sigma\{0 - - - 9\}$ such that the final digit has appeared before

$L = \{010, 12132, 12343, - - - - \}$



Conversion of NFA to DFA

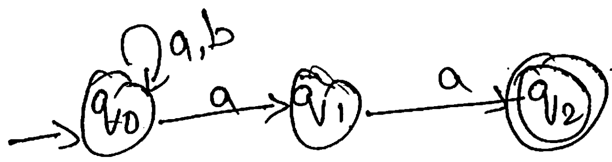
→ A language that can be accepted by a DFA can be accepted by an NFA. Every DFA is an NFA

Conversion of NFA to DFA is called subset construction or powerset construction.

→ Each state in the DFA is associated with a set of states in the NFA.

→ The start state in the DFA corresponds to the start state of the NFA plus all states reachable via ε-transitions.

Eg:



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

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

$\delta'(q_0, a) = \{q_0, q_1\}$ $\quad\quad$ $\delta'(q_2, a) = \phi$

$\delta'(q_0, b) = \{q_0\}$ $\quad\quad$ $\delta'(q_2, b) = \phi$

$\delta'(q_1, a) = \{q_2\}$

$\delta'(q_1, b) = \{\phi\}$

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

$\{q_0, q_1\} \cup \{q_2\}$

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

$\delta'([q_0,q_1], b) = \delta(q_0,b) \cup \delta(q_1,b)$

$\{q_0\} \cup \{\cancel{q_0,q_1}\ \emptyset\} = \{q_0, \cancel{\emptyset}\}\ \{q_0\}$

$\delta'([q_0,q_1,q_2], a) = \delta(q_0,a) \cup \delta(q_1,a) \cup \delta(q_2,a)$

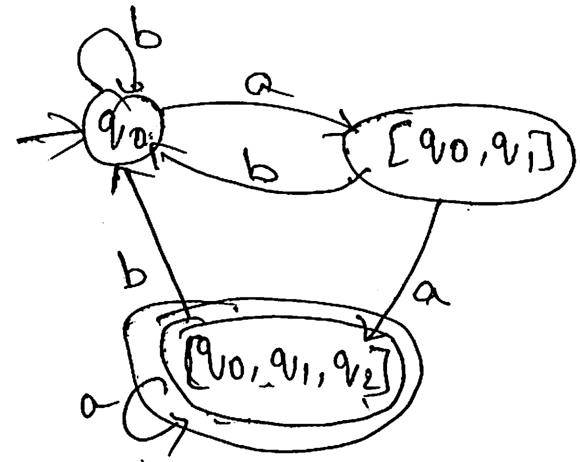
$= \{q_0,q_1\} \cup \{q_2\} \cup \{\emptyset\}$

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

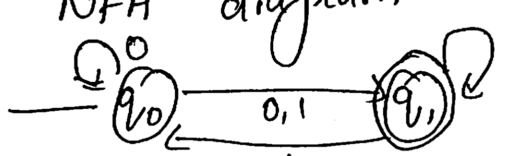
$\delta'([q_0,q_1,q_2], b) = \delta(q_0,b) \cup \delta(q_1,b) \cup \delta(q_2,b)$

$= \{q_0\} \cup \{\emptyset\} \cup \{\emptyset\} = \{q_0\}$

δ	a	b
$\rightarrow q_0$	$\{q_0,q_1\}$	$\{q_0\}$
$\{q_0,q_1\}$	$\{q_0,q_1,q_2\}$	$\{q_0\}$
$\{q_0,q_1,q_2\}$	$\{q_0,q_1,q_2\}$	$\{q_0\}$



$\rightarrow$ Given NFA diagram



$\Rightarrow$

δ	0	1
$\rightarrow q_0$	$\{q_0,q_1\}$	$\{q_1\}$
q_1^*	—	$\{q_1,q_0\}$

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

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

$\delta'(q_1,0) = \emptyset$

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

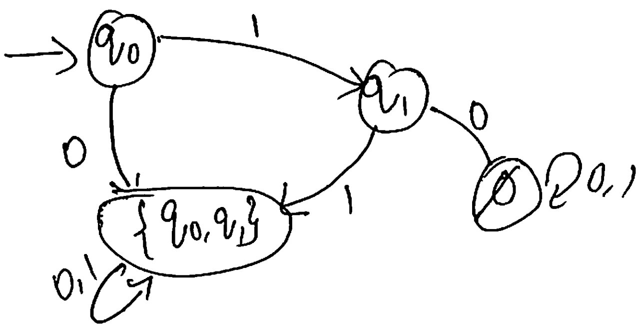
$\delta'(\{q_0,q_1\},0) = \delta'(q_0,0) \cup \delta'(q_1,0)$

$\{q_0,q_1\} \cup \{\emptyset\}$

$= \{q_0,q_1\}$

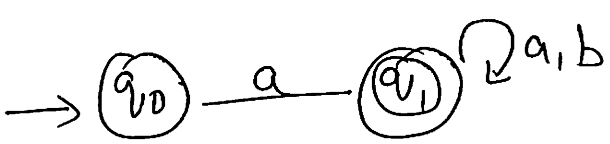
$\delta'(\{q_0,q_1\},1) = \delta'(q_0,1) \cup \delta'(q_1,1)$
$\{q_1\} \cup \{q_0,q_1\}$
$= \{q_0,q_1\}$

δ'	0	1
$\rightarrow q_0$	$\{q_0,q_1\}$	$\{q_1\}$
q_1	$\emptyset$	$\{q_0,q_1\}$
$*\{q_0,q_1\}$	$\{q_0,q_1\}$	$\{q_0,q_1\}$



→ $L = \{\text{starts with `a'}\}$ $\Sigma = \{a,b\}$

NFA –



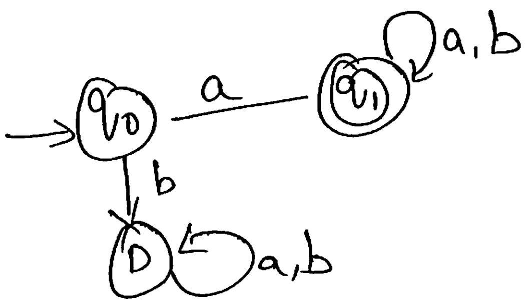
δ	a	b
$\rightarrow q_0$	q_1	–
q_1^*	q_1	q_1

$\xRightarrow{\text{DFA TT}}$

δ'	a	b
$\rightarrow q_0$	q_1	D
q_1^*	q_1	q_1
D	D	D

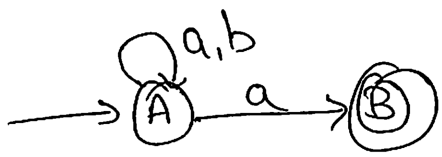
($\because$ no transition it has to redirect to Dead state)

$\therefore$ DFA is



Note :- A DFA transition table can be constructed from a state transition table which is Preferab

Eg: NFA L = { ends with 'a' } $\Sigma = (a,b)^*$

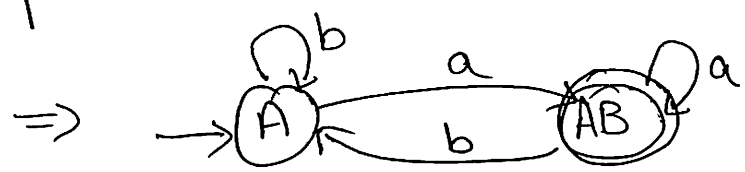


δ	a	b
A	{A,B}	A
B	—	—

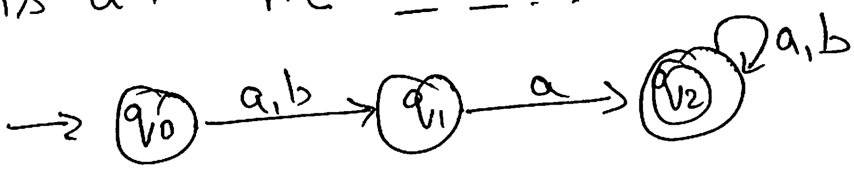
$\xrightarrow[\text{table}]{\text{DFA}}$

δ'	a	b
→ A	[AB]	[A]
AB*	[AB]	[A]

⇒



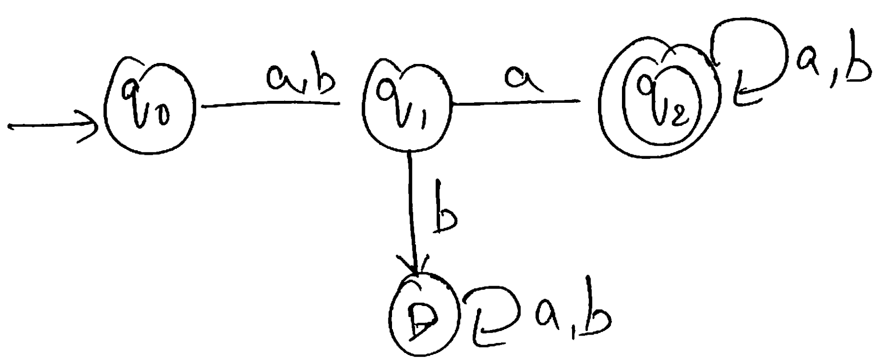
→ Eg :- Given NFA in which second symbol from LHS is 'a'. i.e _ a . . .



δ	a	b
→ q_0	q_1	q_1
q_1	q_2	—
* q_2	q_2	q_2

$\xrightarrow{\text{DFA}}$

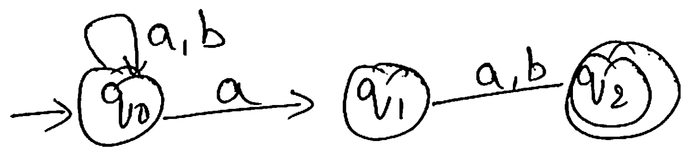
δ'	a	b
→ q_0	q_1	q_1
q_1	q_2	D
q_2	q_2	q_2
D	D	D



Eg → Second symbol from RHS 'a'

$L = \{aa, ab, aaa, baa, bab, \ldots\}$

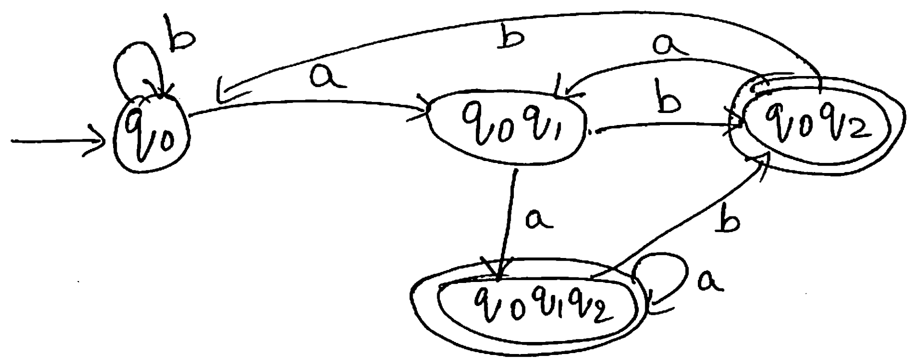
NFA



δ	a	b
$\to q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_2\}$
$*q_2$	–	–

DFA ⟹

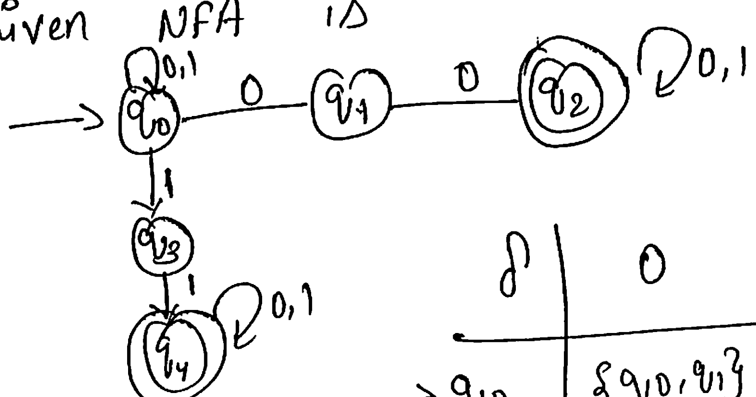
δ'	a	b
$\to q_0$	$[q_0 q_1]$	$[q_0]$
$[q_0 q_1]$	$[q_0 q_1 q_2]$	$[q_0 q_2]$
$[q_0 q_1 q_2]$	$[q_0 q_1 q_2]$	$[q_0 q_2]$
$[q_0 q_2]$	$[q_0 q_1]$	$[q_0]$



i/p bbbaa / bbbab

bbbab

Eg → Given NFA is



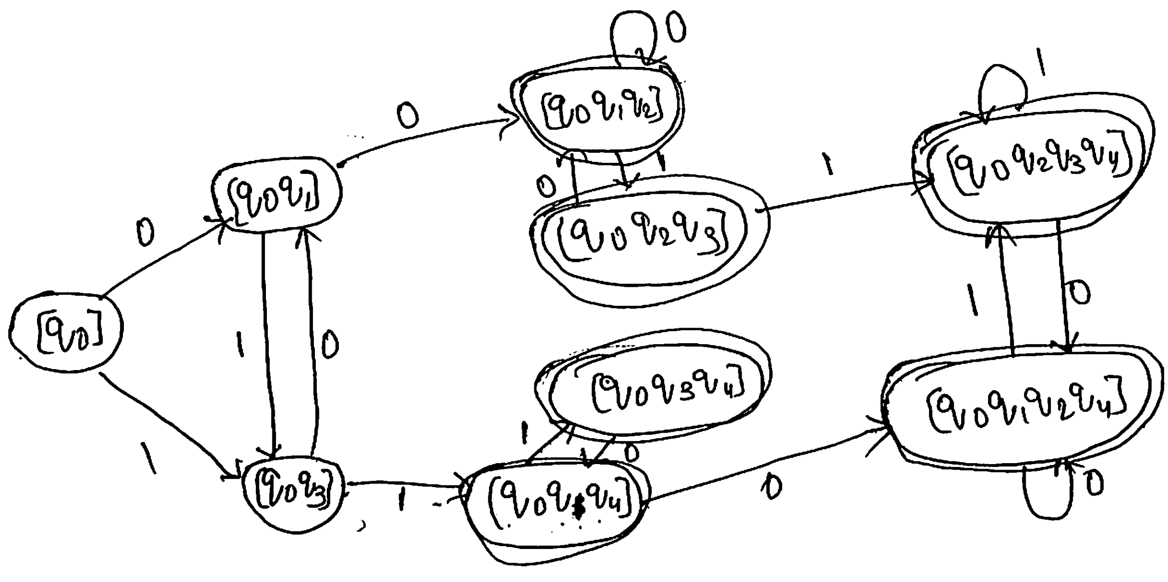
δ	0	1
$\to q_0$	$\{q_0, q_1\}$	$\{q_0, q_3\}$
q_1	$\{q_2\}$	–
q_2	$\{q_2\}$	$\{q_2\}$
q_3	–	q_4
$*q_4$	$\{q_1\}$	$\{q_1\}$

DFA transition table

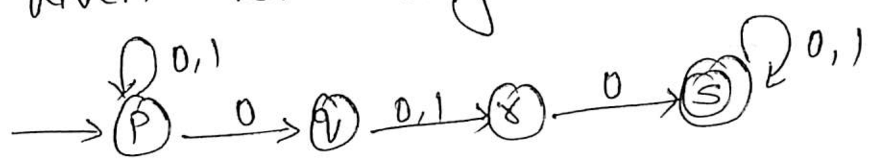
δ'	0	1
$\rightarrow q_0$	$[q_0 q_1]$	$[q_3 q_0]$
$[q_0 q_1]$	$[q_0 q_1 q_2]$	$[q_0 q_3]$
$[q_3 q_0]$	$[q_0 q_1]$	$[q_0 q_3 q_4]$
$*[q_0 q_1 q_2]$	$[q_0 q_1 q_2]$	$[q_0 q_2 q_3]$
$*[q_0 q_3 q_4]$	$[q_0 q_1 q_4]$	$[q_0 q_3 q_4]$
$*[q_0 q_2 q_3]$	$[q_0 q_1 q_2]$	$[q_0 q_2 q_3 q_4]$
$*[q_0 q_1 q_4]$	$[q_0 q_1 q_2 q_4]$	$[q_0 q_3 q_4]$
$*[q_0 q_2 q_3 q_4]$	$[q_0 q_1 q_2 q_4]$	$[q_0 q_2 q_3 q_4]$
$*[q_0 q_1 q_2 q_4]$	$[q_0 q_1 q_2 q_4]$	$[q_0 q_2 q_3 q_4]$

∴ Transition table is equal for transition diagram for DFA.

→ The final states of DFA which consits of $q_2 q_4$ of NFA.



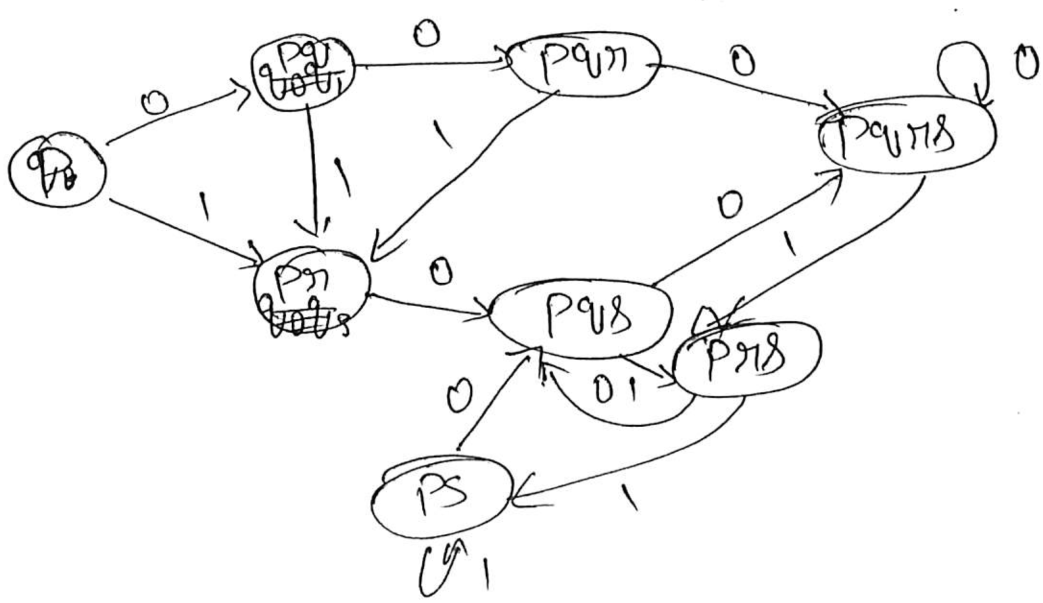
Eg → Given NFA diagram is



δ	0	1
→p	{p,q}	p
q	r	r
r	s	—
*s	s	s

$\xRightarrow{DFA}$

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

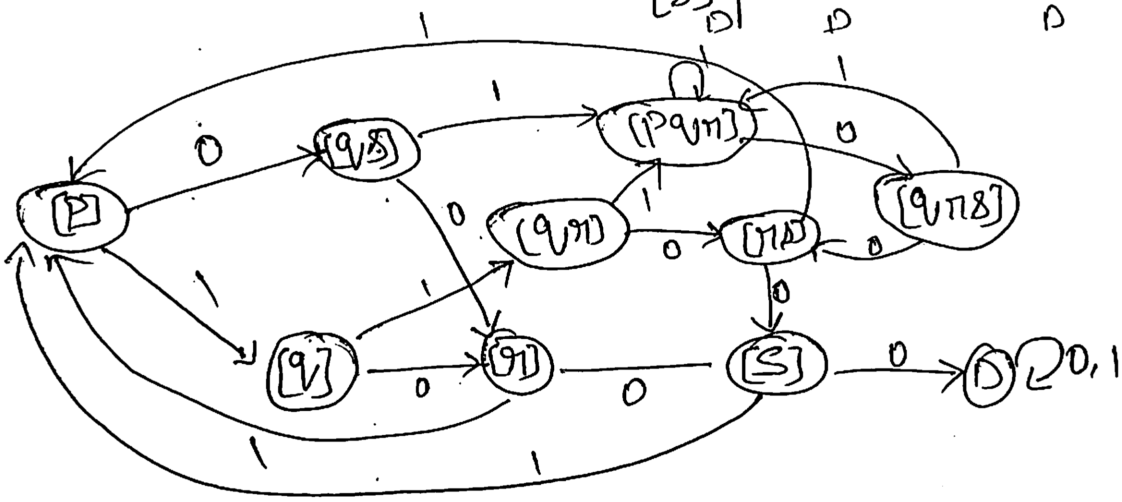


Eg →

δ	0	1
→ p	{q, s}	{q}
q	{r}	{q r}
r	{s}	{p}
* s	—	{p}

DFA ⟹

δ'	0	1
→ p	[q s]	[q]
q	[r]	[q r]
[r]	[s]	[p]
* [q s]	[r]	[p q r]
[q r]	[r s]	[p q r]
* [r s]	[s]	[p]
[p q r]	[q r s]	[p q r]
* [q r s]	[r s]	[p q r]
[s]	D	[p]
D	D	D



* NFA with ϵ-moves:

It is other form of finite automata where NFA $\cong$ ϵ-NFA in which Σ, Q, q_0 will be same & δ, F may change. Defined as $M = (Q, \Sigma, \delta, q_0, F)$

on alphabet: $\delta = Q \times (\Sigma \cup \{\epsilon\})$

on strings: $\hat{\delta} ; Q \times \Sigma^* \longrightarrow 2^Q$

i.e $\delta(q,a) = P$ will give a set of states on input symbol transitions labelled as "a".

$\hat{\delta}(q,a) = P$ will give a set of states on input string transitions labelled as 'a' & 'ϵ'.

NFA with ϵ-moves: $\delta(q,\epsilon) = P$ will give a set of states on i/p symbol with 'ϵ'.

$\hat{\delta}(q,\epsilon) = P$ a set of states that can be reached on 'ϵ' symbol.

ϵ-closure (q):-

will give a set of states 'P' which can be reached only with ϵ symbol.

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

$\rightarrow \hat{\delta}(q,a) = P$

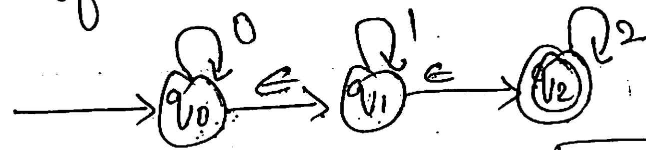
$$\hat{\delta}(q_0, wa) = \epsilon\text{-closure}(P)$$
$$= \epsilon\text{-closure}(\delta(\hat{\delta}(q_0,w),a))$$

Extended ~~Transition function~~:

$$\boxed{\begin{aligned} * \quad \hat{\delta}(q,a) &= \epsilon\text{-closure}(\delta(\hat{\delta}(q,\epsilon),a)) \\ \hat{\delta}(q,\epsilon) &= \epsilon\text{-closure}(q) \end{aligned}}$$

Eg: Conversion of NFA with 'ε' to NFA:

$0^* 1^* 2^*$



δ	0	1	2	ϵ
q_0	q_0	$\emptyset$	$\emptyset$	q_1
q_1	$\emptyset$	q_1	$\emptyset$	q_2
q_2	$\emptyset$	$\emptyset$	q_2	$\emptyset$

Formula

$$\boxed{\delta'(q,a) = \hat{\delta}(q,a)}$$

Transition function of NFA with 'ε'

$\hat{\delta}(q_0,\epsilon) = \{q_0,q_1,q_2\} = \epsilon\text{-closure}(q_0)$

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

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

$\delta'(q_0,0) = \hat{\delta}(q_0,0) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0,\epsilon),0))$

$= \epsilon\text{-closure}(\delta(q_0,q_1,q_2),0)$

$= \epsilon\text{-closure}(\delta(q_0,0) \cup \delta(q_1,0) \cup \delta(q_2,0))$

$= \epsilon\text{-closure}(\{q_0\} \cup \emptyset \cup \emptyset) = \epsilon\text{-closure}(q_0)$

$= \{q_0,q_1,\underline{q_2}\}$ Final state

$\delta'(q_0,1) = \hat{\delta}(q_0,1) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0,\epsilon),1))$

$\Rightarrow \epsilon\text{-closure}(\delta(q_0,q_1,q_2,1))$

$\Rightarrow \epsilon\text{-closure}[\delta(q_0,1) \cup \delta(q_1,1)\} \cup (q_2,1))$

$\Rightarrow \epsilon\text{-closure}(q_1 \cup \emptyset \cup \emptyset)$

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

$= \{q_1,q_2\}$

$\delta'(q_0, 2) = \hat{\delta}(q_0, 2) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, \epsilon), 2))$

$\Rightarrow \epsilon\text{-closure}(\delta(q_0, q_1, q_2), 2))$

$\Rightarrow \epsilon\text{-closure}(\delta(q_0, 2) \cup \delta(q_1, 2) \cup \delta(q_2, 2))$

$\Rightarrow \epsilon\text{-closure}(\emptyset \cup \emptyset \cup q_2)$

$= \{q_2\}$

$\delta'(q_1, 0) = \hat{\delta}(q_1, 0) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), 0))$

$= \epsilon\text{-closure}(\delta(q_1, q_2), 0)$

$= \epsilon\text{-closure}(\delta(q_1, 0) \cup \delta(q_2, 0)$

$= \epsilon\text{-closure}(\emptyset \cup \emptyset) = \emptyset$

$\delta'(q_1, 1) = \hat{\delta}(q_1, 1) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), 1))$

$= \epsilon\text{-closure}(\delta(q_1, q_2), 1)$

$= \epsilon\text{-closure}(\delta(q_1, 1) \cup \delta(q_2, 1))$

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

$= \{q_1, q_2\}$

$\delta'(q_1, 2) = \hat{\delta}(q_1, 2) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), 2))$

$\Rightarrow \epsilon\text{-closure}(\delta(q_1, q_2), 2)$

$= \epsilon\text{-closure}(\delta(q_1, 2) \cup \delta(q_2, 2))$

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

$= \{q_2\}$

$\delta'(q_2, 0) = \hat{\delta}(q_2, 0) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2, \epsilon), 0))$

$= \epsilon\text{-closure}(\delta(q_2), 0)$

$= \epsilon\text{-closure}(\delta(q_2, 0))$

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

$\delta'(q_2,1) = \hat{\delta}(q_2,1) = \epsilon\text{-closure}(\delta'(\hat{\delta}(q_2,\epsilon),1))$

$= \epsilon\text{-closure}(\delta(q_2,1))$

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

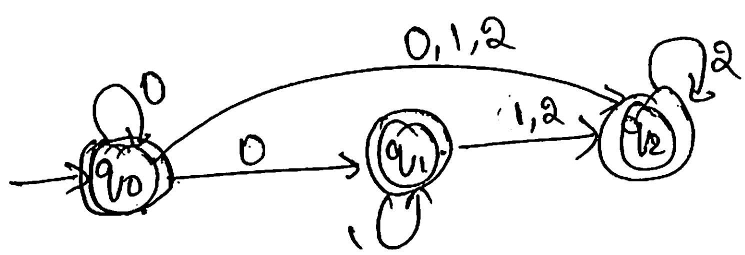
$\delta'(q_2,2) = \hat{\delta}(q_2,2) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2,\cancel{\epsilon}),2))$

$= \epsilon\text{-closure}(\delta(q_2,2))$

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

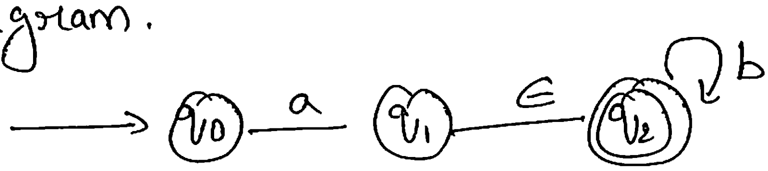
$= \{q_2\}$

δ'	0	1	2
$\rightarrow * q_0$	$\{q_0,q_1,q_2\}$	$\{q_1,q_2\}$	$\{q_2\}$
$* q_1$	ϕ	$\{q_1,q_2\}$	$\{q_2\}$
$* q_2$	ϕ	ϕ	$\{q_2\}$



<u>Note</u>: If ϵ-closure of initial state has final state then all states of initial states are final states

(1) → Find ϵ-closures of all states for the following diagram.

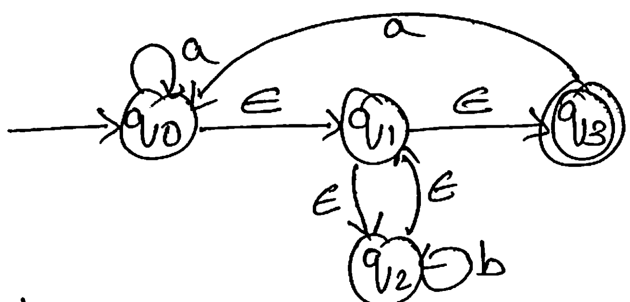


ϵ-closure(q_0) = $\{q_0\}$

ϵ-closure(q_1) = $\{q_1, q_2\}$

ϵ-closure(q_2) = $\{q_2\}$

(2) →



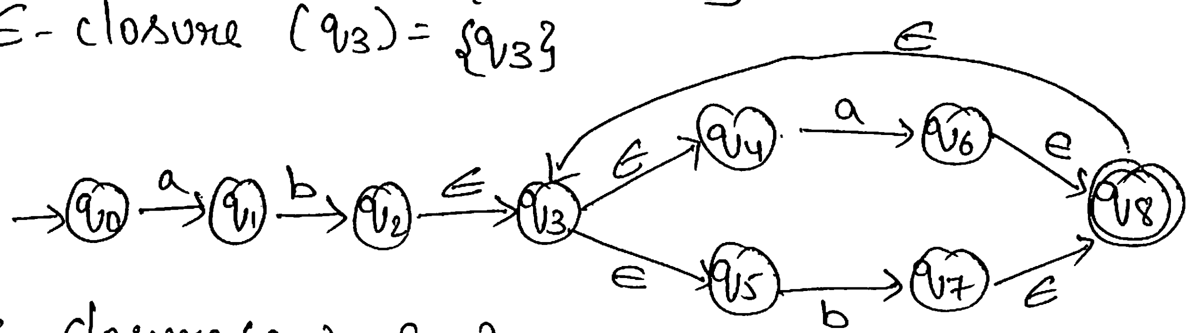
ϵ-closure(q_0) = $\{q_0, q_1, q_2, q_3\}$

ϵ-closure(q_1) = $\{q_1, q_2, q_3\}$

ϵ-closure(q_2) = $\{q_1, q_2, q_3\}$

ϵ-closure(q_3) = $\{q_3\}$

(3) →



ϵ-closure(q_0) = $\{q_0\}$

ϵ-closure(q_1) = $\{q_1\}$

ϵ-closure(q_2) = $\{q_2, q_3, q_4, q_5\}$

ϵ-closure(q_3) = $\{q_3, q_4, q_5\}$

ϵ-closure(q_4) = $\{q_4\}$

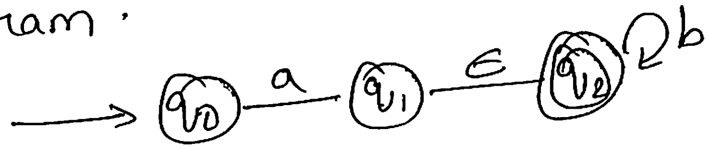
ϵ-closure(q_5) = $\{q_5\}$

ϵ-closure(q_6) = $\{q_6, q_8, q_3, q_4, q_5\}$

ϵ-closure(q_7) = $\{q_7, q_8, q_3, q_4, q_5\}$

ϵ-closure(q_8) = $\{q_8, q_3, q_4, q_5\}$

① Eliminate the ϵ-transitions from the following diagram.



Step 1: ϵ-closure $(q_0) = \{q_0\} = \hat{\delta}'(q_0, \epsilon)$

ϵ-closure $(q_1) = \{q_1, q_2\} = \hat{\delta}'(q_1, \epsilon)$

ϵ-closure $(q_2) = \{q_2\} = \hat{\delta}'(q_2, \epsilon)$

Step 2:- Extended transition function

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

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

$$\begin{aligned}
\delta'(q_0, a) &= \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, \epsilon), a)) \\
&= \epsilon\text{-closure}(\delta(q_0), a) \\
&= \epsilon\text{-closure}(\delta(q_0, a)) \\
&= \epsilon\text{-closure}(q_1) \\
&= \{q_1, q_2\}
\end{aligned}$$

$$\begin{aligned}
\delta'(q_0, b) &= \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, \epsilon), b)) \\
&= \epsilon\text{-closure}(\delta(q_0, b)) \\
&= \epsilon\text{-closure}(\emptyset) = \{\emptyset\}
\end{aligned}$$

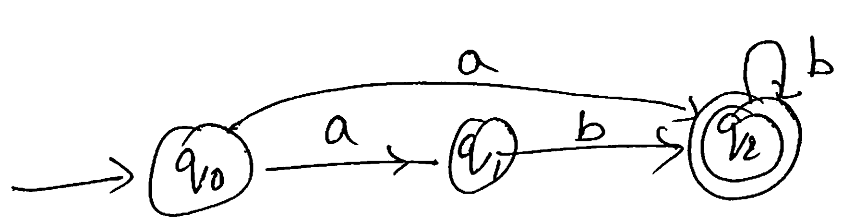
$$\begin{aligned}
\delta'(q_1, a) &= \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), a)) \\
&= \epsilon\text{-closure}(\delta(q_1, q_2), a)) \\
&= \epsilon\text{-closure}(\delta(q_1, a) \cup \delta(q_2, a)) \\
&= \epsilon\text{-closure}(\emptyset \cup \emptyset) = \emptyset
\end{aligned}$$

$$\delta'(q_1, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), b))$$
$$\Rightarrow \epsilon\text{-closure}(\delta(q_1, q_2), b)$$
$$\Rightarrow \epsilon\text{-closure}(\delta(q_1, b) \cup \delta(q_2, b))$$
$$\Rightarrow \epsilon\text{-closure}(\emptyset \cup q_2)$$
$$= \{q_2\}$$

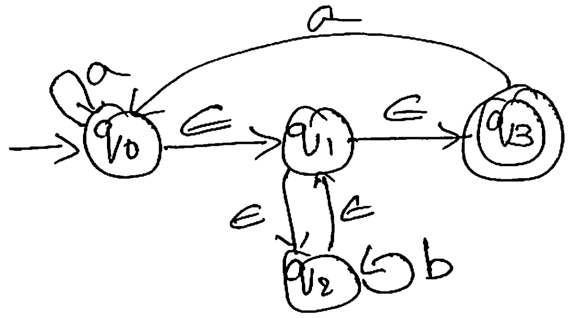
$$\delta'(q_2, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2, \epsilon), a))$$
$$= \epsilon\text{-closure}(\delta(q_2, a))$$
$$= \epsilon\text{-closure}(\emptyset) = \{\emptyset\}$$

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

δ'	a	b
$\rightarrow q_0$	$\{q_1, q_2\}$	$\emptyset$
q_1	$\emptyset$	$\{q_2\}$
q_2	$\emptyset$	$\{q_2\}$



② $\rightarrow$



step 1: $\hat{\delta}(q_0, \epsilon) = \{q_0, q_1 q_2, q_3\} = \epsilon\text{-closure}(q_0)$

$\hat{\delta}(q_1, \epsilon) = \{q_1, q_2, q_3\} = \epsilon\text{-closure}(q_1)$

$\hat{\delta}(q_2, \epsilon) = \{q_2, q_1, q_3\} = \epsilon\text{-closure}(q_2)$

$\hat{\delta}(q_3, \epsilon) = \{q_3\} = \epsilon\text{-closure}(q_3)$

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

$= \epsilon\text{-closure}(\delta(q_0, q_1, q_2, q_3), a)$

$= \epsilon\text{-closure}(\delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a) \cup \delta(q_3$

$= \epsilon\text{-closure}(q_0 \cup \emptyset \cup \emptyset, q_0)$

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

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

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

$= \epsilon\text{-closure}(\delta(q_0, q_1, q_2, q_3), b)$

$= \epsilon\text{-closure}(\delta(q_0, b) \cup (q_1, b) \cup \delta(q_2, b) \cup \delta(q_3, b))$

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

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

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

$= \epsilon\text{-closure}(\delta(q_1, q_2, q_3), a)$

$= \epsilon\text{-closure}(\delta(q_1, a) \cup \delta(q_2, a) \cup \delta(q_3, a))$

$= \epsilon\text{-closure}(\emptyset \cup \emptyset \cup q_0)$

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

$$\delta'(q_1, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), b))$$
$$= \epsilon\text{-closure}(\delta(q_1, q_2, q_3), b)$$
$$= \epsilon\text{-closure}(\delta(q_1, b) \cup \delta(q_2, b) \cup \delta(q_3, b))$$
$$= \epsilon\text{-closure}(\emptyset \cup q_2 \cup \emptyset)$$
$$= \{q_1, q_2, q_3\}$$

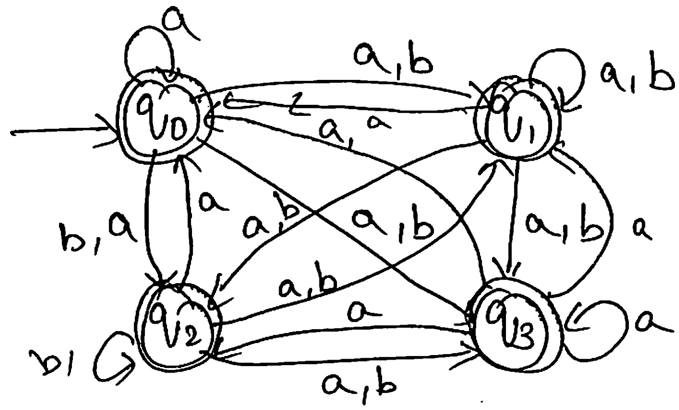
$$\delta'(q_2, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2, \epsilon), a))$$
$$= \epsilon\text{-closure}(\delta(q_1, q_2, q_3), a)$$
$$= \epsilon\text{-closure}(\delta(q_1, a) \cup \delta(q_2, a) \cup \delta(q_3, a))$$
$$= \epsilon\text{-closure}(\emptyset \cup \emptyset \cup q_0)$$
$$= \{q_0, q_1, q_2, q_3\}$$

$$\delta'(q_2, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2, \epsilon), b))$$
$$= \epsilon\text{-closure}(\delta(q_1, q_2, q_3), b)$$
$$= \epsilon\text{-closure}(\delta(q_1, b) \cup \delta(q_2, b) \cup \delta(q_3, b))$$
$$= \epsilon\text{-closure}(\emptyset \cup q_2 \cup \emptyset)$$
$$= \{q_1, q_2, q_3\}$$

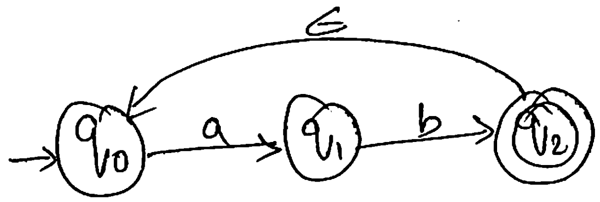
$$\delta'(q_3, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_3, \epsilon), a))$$
$$= \epsilon\text{-closure}(\delta(q_3, a))$$
$$= \epsilon\text{-closure}(q_0)$$
$$= \{q_0, q_1, q_2, q_3\}$$

$$\delta'(q_3, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_3, \epsilon), b))$$
$$= \epsilon\text{-closure}(\delta(q_3, b))$$
$$= \epsilon\text{-closure}(\emptyset)$$
$$= \{\emptyset\}$$

δ'	a	b
$\rightarrow q_0$	$\{q_0, q_1, q_2, q_3\}$	$\{q_2, q_1, q_3\}$
q_1	$\{q_1, q_2, q_3\}$	$\{q_1, q_2, q_3\}$
q_2	$\{q_0, q_1, q_2, q_3\}$	$\{q_1, q_2, q_3\}$
$*q_3$	$\{q_0, q_1, q_2, q_3\}$	$\emptyset$



③ $\Rightarrow$



δ	a	b	ϵ
$\rightarrow q_0$	q_1	–	–
q_1	–	q_2	–
$*q_2$	–	–	q_0

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

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

$\hat{\delta}(q_2, \epsilon) = \{q_0, q_2\} = \epsilon\text{-closure}(q_2)$

$$\begin{aligned}\delta'(q_0, a) &= \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, \epsilon), a)) \\ &= \epsilon\text{-closure}(\delta(q_0, a)) \\ &= \epsilon\text{-closure}(q_1) \\ &= \{q_1\}\end{aligned}$$

$$\begin{aligned}\delta'(q_0, b) &= \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, \epsilon), b)) \\ &= \epsilon\text{-closure}(\delta(q_0, b)) \\ &= \epsilon\text{-closure}(\emptyset) \\ &= \emptyset\end{aligned}$$

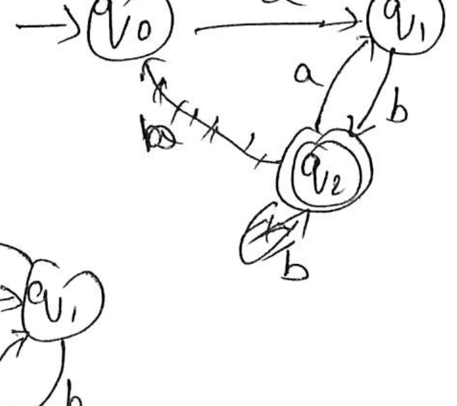
$$\delta'(q_1, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), a))$$
$$= \epsilon\text{-closure}(\delta(q_1, a))$$
$$= \epsilon\text{-closure}(\emptyset)$$
$$= \emptyset$$

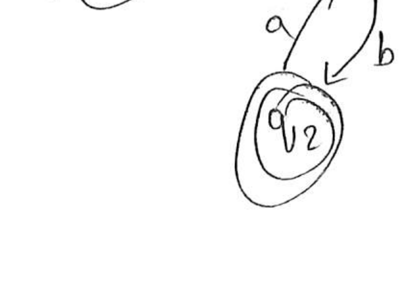
$$\delta'(q_1, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), b))$$
$$= \epsilon\text{-closure}(\delta(q_1, b))$$
$$= \epsilon\text{-closure}(q_2)$$
$$= \{q_0, q_2\}$$

$$\delta'(q_2, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2, \epsilon), a))$$
$$= \epsilon\text{-closure}(\delta(q_0, q_2), a)$$
$$= \epsilon\text{-closure}(\delta(q_0, a) \cup \delta(q_2, a))$$
$$= \epsilon\text{-closure}(q_1 \cup \emptyset)$$
$$= \{q_1\}$$

$$\delta'(q_2, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2, \epsilon), b))$$
$$= \epsilon\text{-closure}(\delta(q_0, q_2), b)$$
$$= \epsilon\text{-closure}(\delta(q_0, b) \cup \delta(q_2, b))$$
$$= \epsilon\text{-closure}(\emptyset \cup \cancel{q_2})$$
$$= \{\cancel{q_0, q_2}\}\ \emptyset$$

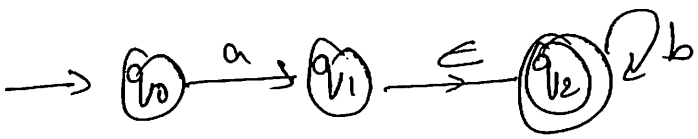
δ'	a	b
$\rightarrow q_0$	q_1	—
q_1	—	$\{q_0, q_2\}$
$* q_2$	$\{q_1\}$	$\cancel{\{q_0, q_2\}}\ \emptyset$





① Conversion ϵ-NFA to NFA Shortcut method for
* Note:-

Given ϵ-NFA



Step 1: Construct the transition table for NFA by using the formula $\epsilon^* a \epsilon^*$

i.e $\delta(A,a) \rightarrow \epsilon^* a \epsilon^*$

i.e ϵ-closure $(\delta(\epsilon\text{-closure}(A), a))$

	a	b
$\rightarrow q_0$	$\{q_1, \underline{q_2}\}$ LFS	$\emptyset$
q_1	$\emptyset$	$\{q_2\}$
$* q_2$	$\emptyset$	$\{q_2\}$

$\delta(q_0, a) = \epsilon^* a \epsilon^*$

$q_0 \xrightarrow{\epsilon^*} q_0 \xrightarrow{a} q_1 < \begin{matrix} q_1 \\ q_2 \end{matrix}$

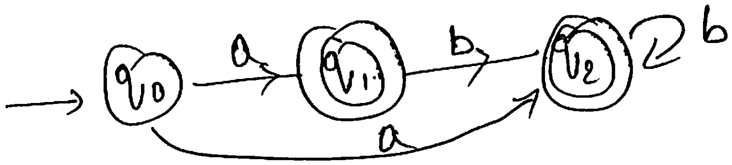
$\delta(q_0, b) = \epsilon^* b \epsilon^*$

$q_0 \xrightarrow{\epsilon^*} q_0 \xrightarrow{b} \emptyset \xrightarrow{\epsilon^*} \{\emptyset\}$

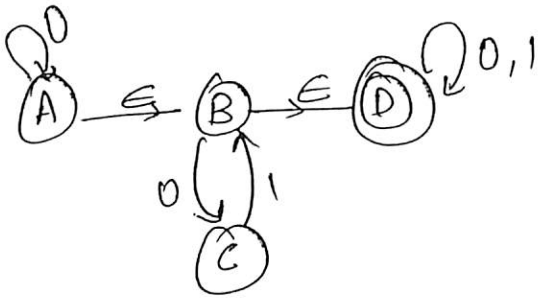
$\delta(q_1, a)$

$\delta(q_1, b)$

⋮

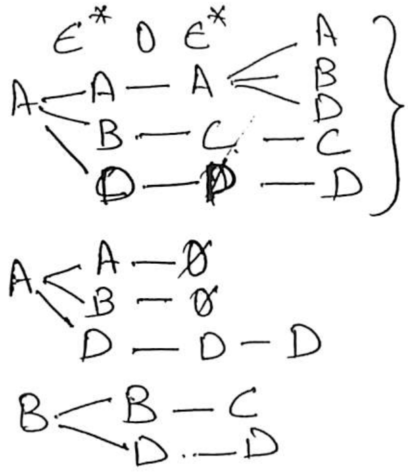


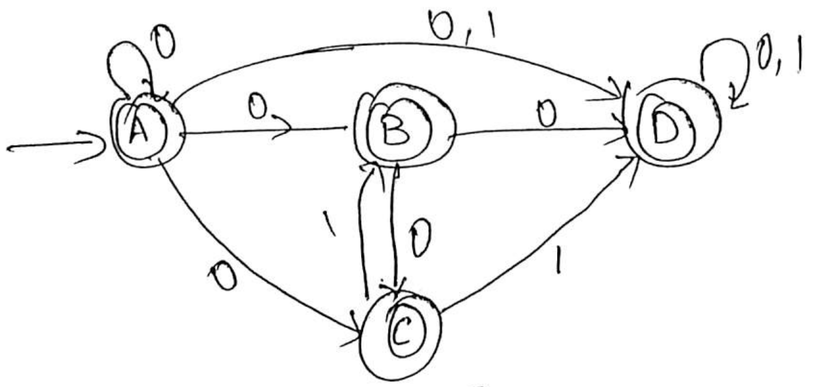
(2) – Given ϵ - NFA



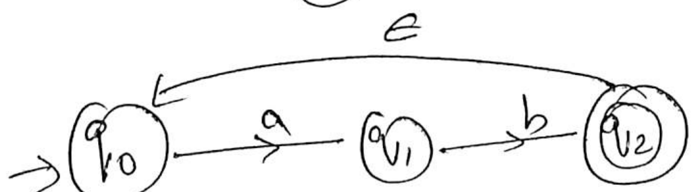
ϵ - NFA to NFA

	0	1
A	{A,B,C,D}	{D}
B	{C,D}	{D}
C	$\emptyset$	{B,D}
D	{D}	{D}





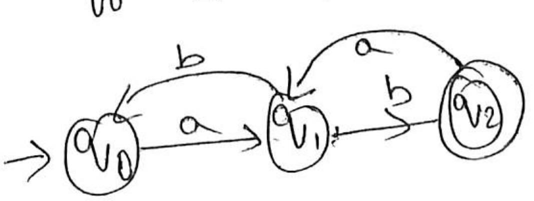
(3)



δ'	a	b
q_0	q_1	$\emptyset$
q_1	$\emptyset$	$\{q_0, q_2\}$
q_2	q_1	$\emptyset$

ϵ^* a ϵ^*

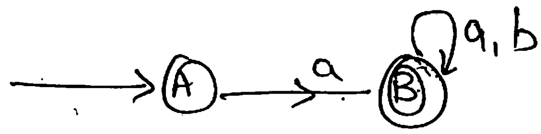
$q_0 \xrightarrow{\epsilon^*} q_0 \xrightarrow{a} q_1 \xrightarrow{\epsilon^*} q_1$



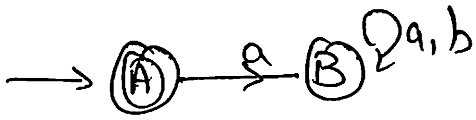
Complementation of NFA

→ In Complement of NFA Non-final states become final states and final states become Non final States. It is denoted as $\overline{M}$.

eg: L = { Starts with 'a' }

M = 

$L_1 = \{a, aab, aba, \ldots\}$

$\overline{M}$ = 

$L_2 = \{\epsilon, \ldots\}$

$$\boxed{\therefore \overline{L_1} \neq L_2}$$

Note: Complementing NFA is different & complementing a language is different

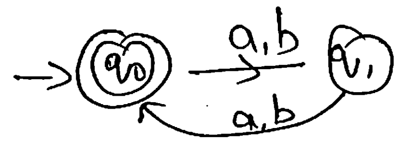
*⇒ when a NFA is Complemented, the language accepted by the NFA need not get complemented.

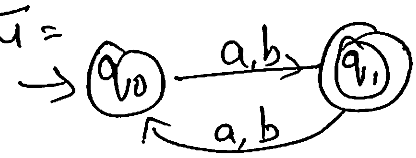
*⇒ when a DFA is complemented, the language accepted by DFA is Complemented.

Complementation of DFA:-

→ Complement of DFA should accept the complement of language accepted by DFA. i.e FS → NFS
NFS → FS

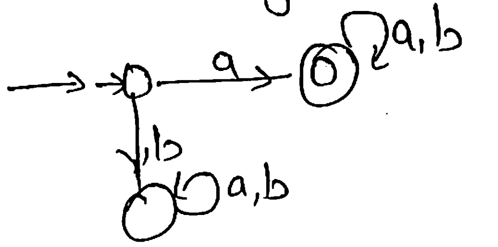
Eg:

M =


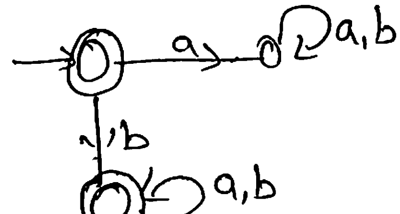
$\overline{M}$ =


$\boxed{\therefore L_1 = L_2}$

eg: starting with 'a'



without starting with 'a'
L = {∈, b, ba . . . }



Note: i) It is only applicable to DFA
ii) $(Q, \Sigma, \delta, q_0, F) \xrightarrow{\text{comp}} (Q, \Sigma, \delta, q_0, Q-F)$

Minimization of DFA:-

If a DFA can be further minimized then it is called as minimization of DFA.

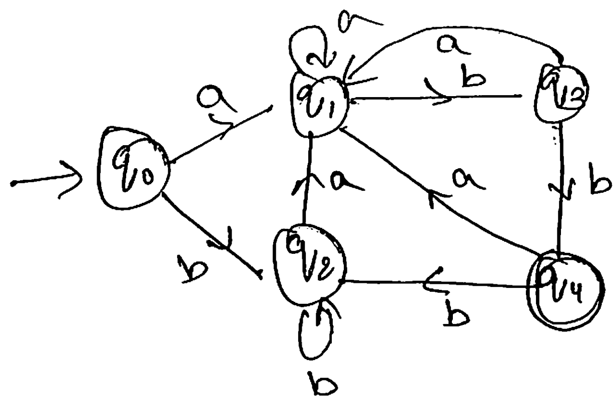
$|w| = 0$ - 0 equivalent
$|w| = 1$ - 1 equivalent
$|w| = 2$ - 2 equivalent

if (P, q) are equivalent if

$\delta(\underset{\text{state}}{P}, \underset{\text{i/p}}{\underline{w}}) \Rightarrow F$ then $\delta(q, w) \Rightarrow F$

$\delta(P, w) \not\Rightarrow F$ then $\delta(q, w) \not\Rightarrow F$

eg:



	a	b
→ q_0	q_1	q_2
q_1	q_1	q_3
q_2	q_1	q_2
q_3	q_1	$*q_4$
$*q_4$	q_1	q_2

⇒ 0 equivalent sets

set of NFS — set of FS

$[q_0 q_1 q_2 q_3]$ $[q_4]$

⇒ 1 equivalent sets

$[q_0 q_1 q_2]$ $[q_3]$ $[q_4]$

(q_0, q_1) on all

$(q_0\ q_2)$

$(q_2\ q_3)$

⇒ 2 equivalent

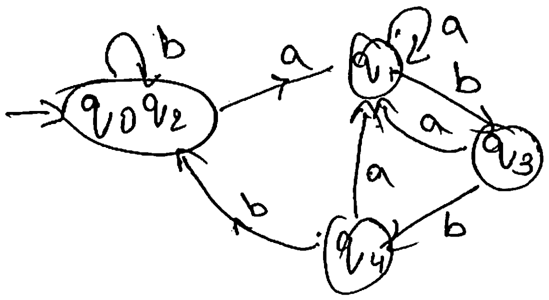
$[q_0 q_2]$ $[q_1]$ $[q_3]$ $[q_4]$

⇒ 3 equivalent

$[q_0 q_2]$ $[q_1]$ $[q_3]$ $[q_4]$

Continue until we ~~go~~ dont get new sets

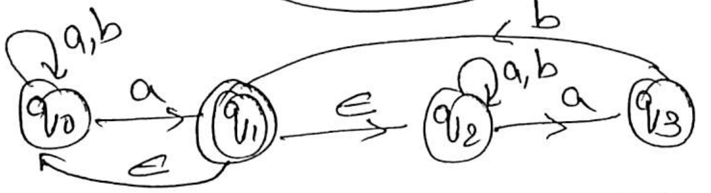
∴ q_0 & q_2 can be combined



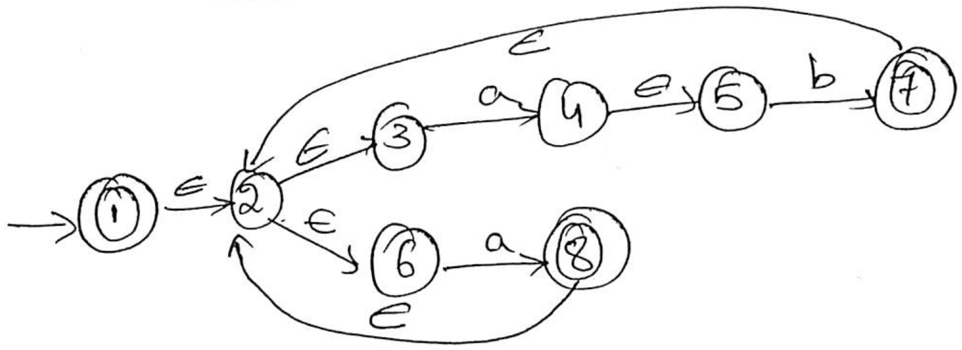
Assignment questions (conversion of ε-NFA to NFA)

1)

2)

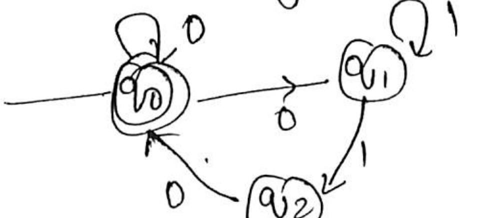


3)



Conversion of NFA to DFA

1)



2)

* Conversion of NFA with ϵ to DFA :-

NFA without ϵ can be converted to its equivalent DFA.

Step 1:- let $M = (Q, \Sigma, \delta, F, q_0)$ is a NFA then

$$M_D = (Q_D, \Sigma, \delta_D, q_0, F_D)$$

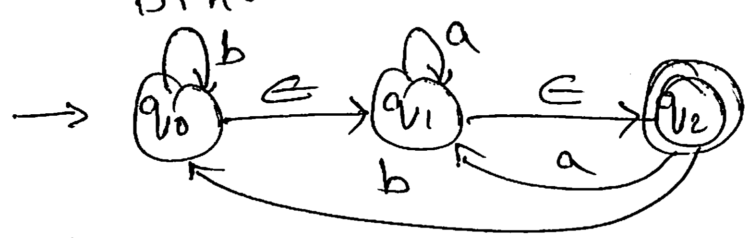
then obtain

ϵ-closure $(q_0) = \{P_1, P_2, P_3 --- P_n\}$ then $[P_1 P_2 P_3 -- P_n]$ becomes a start state of DFA.

Step 2:- then obtain δ transitions on $[P_1, P_2, P_3 -- P_n]$ for each input.

Step 3: The states obtained $[P_1, P_2, P_3, -- P_n] \in Q_D$. The state containing final states in P_i is a final state in DFA.

Eg:- Convert the following NFA with ϵ to equivalent DFA.



Sol: To convert NFA find ϵ-closures

ϵ-closures $(q_0) = \{q_0, q_1, q_2\}$

ϵ-closure $(q_1) = \{q_1, q_2\}$

ϵ-closure $(q_2) = \{q_2\}$