Iⁿᵈ Preeti V Sem

MOORE AND MEALY M

Auto Mata 3 70

Moore Machine :→ A moore machine is a six-t $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$, where.

Q = is a finite set of states

Σ = is the input alphabet

Δ = is the output alphabet

δ = is the transition funct. from $\Sigma \times Q \rightarrow Q$

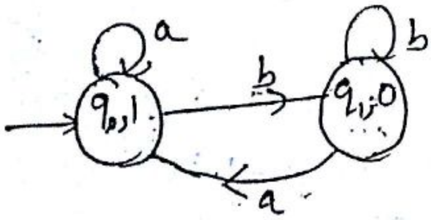
λ = is the output function mapping with

$: Q \rightarrow \Delta$

q_0 = is the intial state.

→ Moore Machine is an output producer.

e.g.



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

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

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

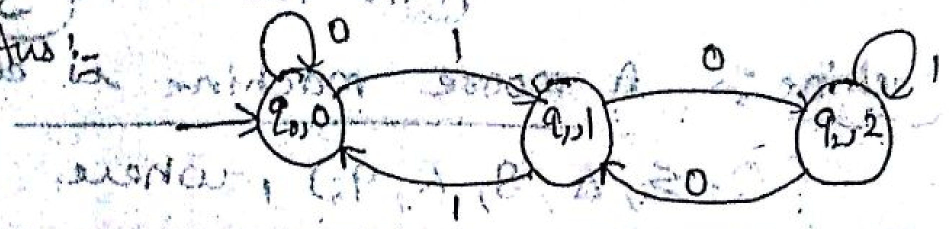
$\lambda = \{(q_0, 1), (q_1, 0)\}$

$q_0 = \{q_0\}$

* In Moore M/C there is no concept of fina state.

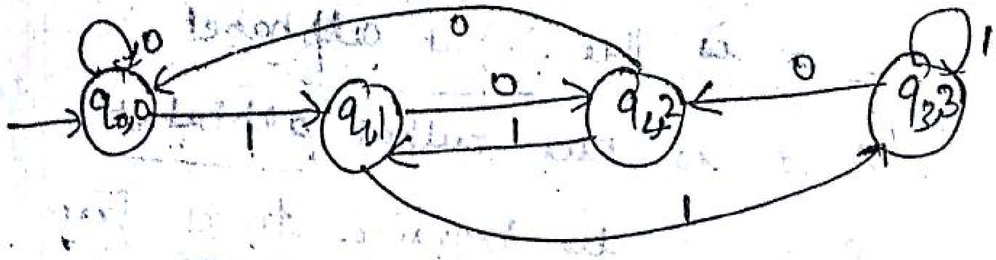
Example:- Mod 3 Moore Machine.

Ans:

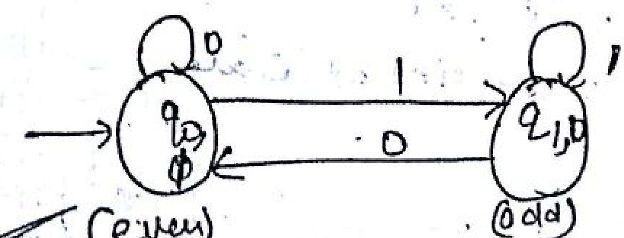


Example:- Mod 4 Moore Machine.

Ans:

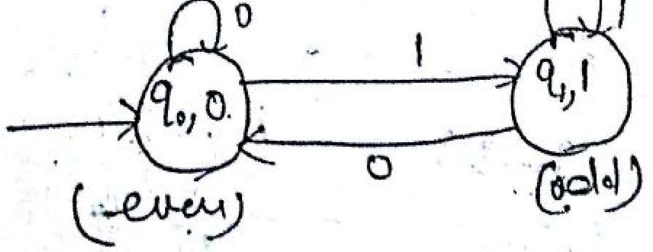


Example:- Draw a moore M/c for even number of



Example:- Draw a moore M/c for Odd number,

Ans



Melay Machine :- A melay Machine is a Six tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$, where

Q = is a finite set of states

Σ = is the input alphabet

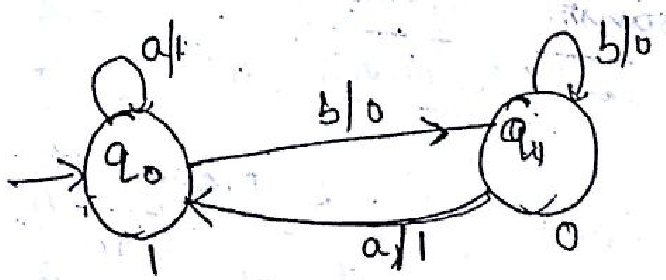
Δ = is the output alphabet

δ = is the transition function from $\Sigma \times Q \rightarrow$

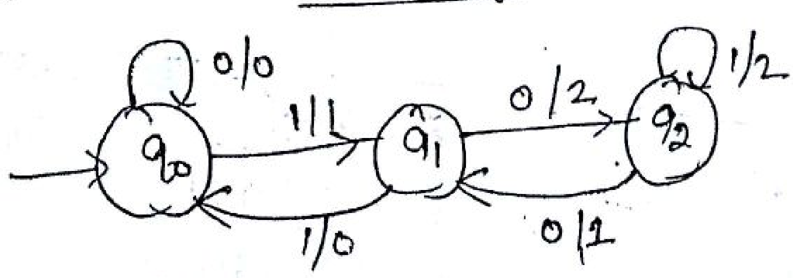
$\lambda = \Sigma \times Q \rightarrow \Delta$

q_0 = initial state

f.e.



Example! Mod 3 mealy M/C :-



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

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

$\Delta = \{0,1,2\}$

$q_0 = \{q_0\}$

$\lambda'(q_0, 0) = 0$. $\quad \delta(q_0, 0) =$

$\lambda'(q_0, 1) = 1 \quad \delta(q_0, 1) =$

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

$\lambda(q_1, 0) = 2 \quad \delta(q_1, 1) =$

$\lambda(q_2, 0) = 1 \quad \delta(q_2, 0) =$

$\lambda(q_2, 1) = 2 \quad \delta(q_2, 1) :$

CONVERT MOORE TO MEALY M

Construction:-

1) we have to define the output function λ' for the Mealy M/C as a function of the present state and the input symbol. we define λ' by:

$$\lambda'(q,a) = \lambda(\delta(q,a))$$ for all states and symbol a.

ii). The transition function is the same as that of given Moore M/C.

Example:-I Construct a Mealy M/C which is equivalent to the moore M/C given table:

Present state	Next state a=0	Next state a=1	output
$\rightarrow q_0$	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

Ans:

Present state	Next state			
	a = 0		a = 1	
	state	output	state	out
→ q_0	q_3	0	q_1	1
q_1	q_1	1	q_2	0
q_2	q_2	0	q_3	0
q_3	q_3	0	q_0	0

Answer.

Example 2:- Construct a Mealy M/C correspond to given Moore M/C.

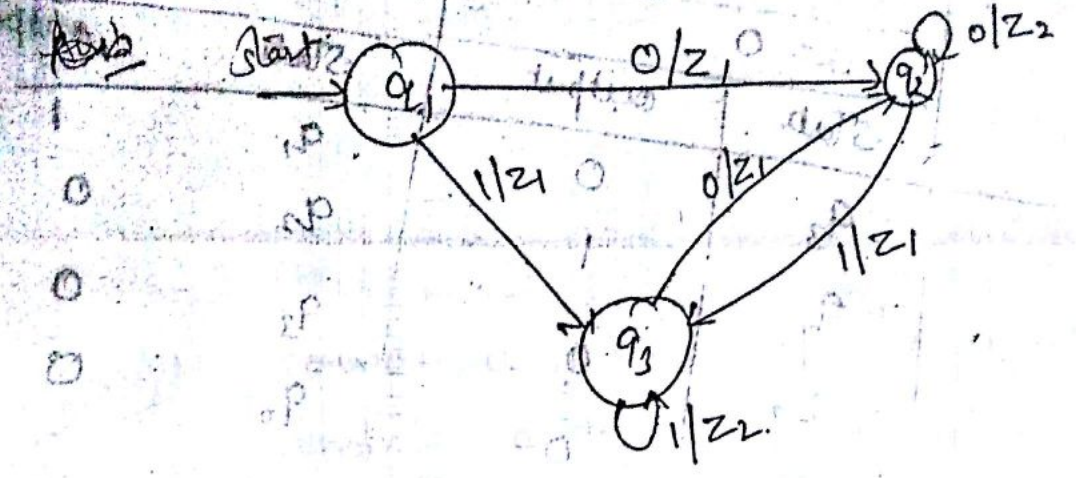
δ	a=0	a=1	output
→ q_1	q_1	q_2	0
q_2	q_1	q_3	0
q_3	q_1	q_3	1

Ans:

δ	a=0	output	a=1	output
→ q_1	q_1	0	q_2	0
q_2	q_1	0	q_3	1
q_3	q_1	0	q_3	1

Answer

Example: Consider a Mealy M/C represent in fig. 6
Construct a Moore M/C equivalent to Mealy

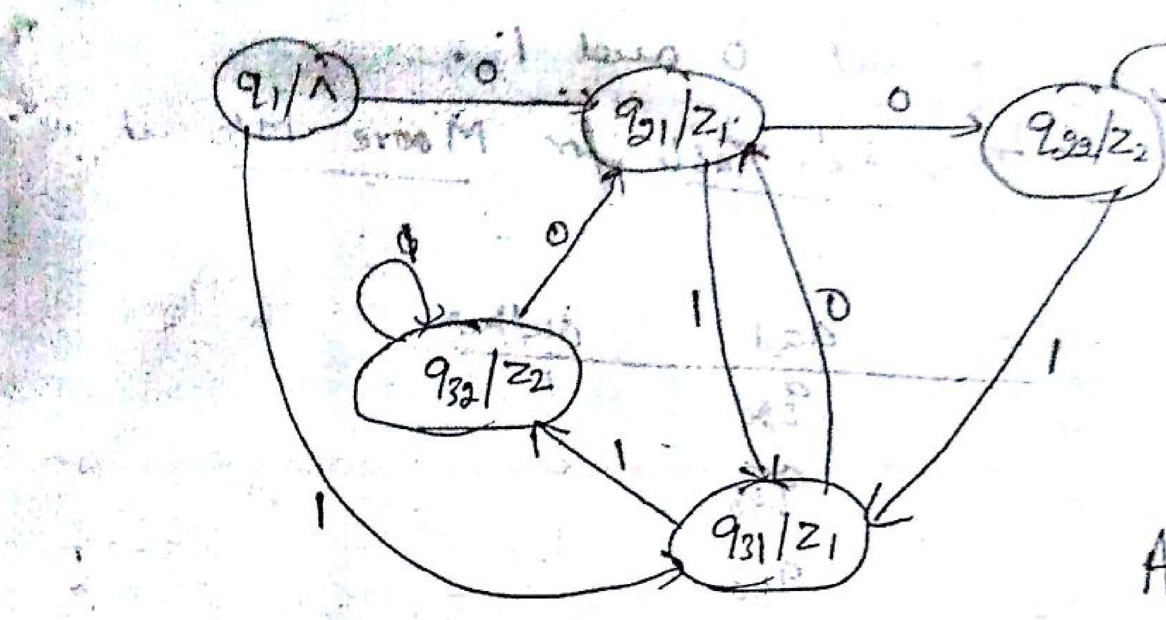


Ans: Transition Table of above Mealy M/c as:

S.	a = 0	output	a = 1	output
$\rightarrow q_1$	q_2	Z_1	q_3	Z_1
q_2	q_2	Z_2	q_3	Z_1
q_3	q_2	Z_1	q_3	Z_2

Now Convert into Moore Machine as:

Present state	a = 0	a = 1	output
q_1	q_{21}	q_{31}	
q_{21}	q_{22}	q_{31}	Z_1
q_{22}	q_{22}	q_{31}	Z_2
q_{31}	q_{21}	q_{32}	Z_1
q_{32}	q_{21}	q_{32}	Z_2



Answer

CONVERT MEALY TO MOORE M

Example I. Consider the Melay M/C in the Transition table. construct a Moore M/C equivalent to the Melay M/C..

	a = 0 STATES	output	a = 1 state	output
$\to q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

Answer:- first we split q_i into several different states. i.e q_1 is associated with one output 1 and q_2 is associated with two different outputs 0 and 1.

Now the Transition Table for Moore M/c is :

STATES	a = 0	a = 1	output
→ q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	0
q_{41}	q_{41}	q_3	1

In above table q_1 produce the output. As q_1 is intial state in above, but that produce output 1 but in moore M/c the output cannot be '1'. So, we require to add New state as q_0 with same ~~ou~~ transitions state as in q_1 but the output produced by q_0 is 0. ∴ New table is

States	a = 0	a = 1	output
→ q_0	q_3	q_{20}	0
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	0
			1

Ans.

PROCEDURE TO CONVERT MEALY TO MOORE

1) we will look into the next state ~~column~~ column for any state q_i and determine number of different outputs associated with
2) split q_i into several different states. Number of such states being number of different output associated with each state.
3) for example-I, q_1 is associated with single out i.e 1 and q_3 is associated with single out i.e 0. There is no need of splitting q_1, q_3.
q_2 is associated with 2 outputs 0,1. So q_2 will split into two states → q_{20}, q_{21} with output 0,1 respectively.
Parallely, there are 2 o/p 0,1 for q_4. So q_4 will split in 2 states – q_{40}, q_{41}
4. The required machine is designed.

Example 2:- Construct a Moore Machine equivalent to Melay machine M. defined by Table!

Present state	a = 0 State	a = 0 output	a = 1 State	a = 1 output
$\rightarrow q_1$	q_1	1	q_2	0
q_2	q_4	1	q_4	1
q_3	q_2	1	q_3	1
q_4	q_3	0	q_1	1

Ans:- Split $\Rightarrow$ q_2 < $q_{20} \rightarrow 0$ (output)
$q_{21} \rightarrow 1$ (output)

q_3 < $q_{30} \rightarrow 0$ (output)
$q_{31} \rightarrow 1$ (output)

So, required Moore Transition table is:

Present state	a = 0	a = 1	output
$\rightarrow q_1$	q_1	q_{20}	1
q_{20}	q_4	q_4	0
q_{21}	q_4	q_4	1
q_{30}	q_{21}	q_{31}	0
q_{31}	q_{21}	q_{31}	1
q_4	q_{30}	q_1	1

As, in moore machine the initial state can not produce output as 1 otherwise moore m/c accepts the Λ-string also. ∴ in our table q_1 state produce the output as 0 ∴, we insert a new state q_0 with output 0 as initial state in previous table as:

states	a=0	a=1	output
$\rightarrow q_0$	q_1	q_{20}	0
q_1	q_1	q_{20}	1
q_{20}	q_4	q_4	0
q_{21}	q_4	q_4	1
q_{30}	q_{21}	q_{31}	0
q_{31}	q_{21}	q_{31}	1
q_4	q_{30}	q_1	1

Answer

MINIMIZATION OF DFA

Defination: Two states q_1 and q_2 are equivale
if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are final
States or both of them are non final st.
for all $x \in \Sigma^*$

Properties of minimization →

① The relations are equivalence and k-equ
lent if they are reflexive, symmetric and transit

② The relation equivalence and k- equiva
-ent can be denoted by Π and Π_k.

③ if q_1 and q_2 are k-equivalent for all
then they are equivalent..

④ If q_1 and q_2 are (k+1) equivalent, then they
are k-equivalent.

⑤ $\Pi_n = \Pi_{n+1}$ for some n.

Scanned by CamScanner

Algorithm to Construct Minimum DFA

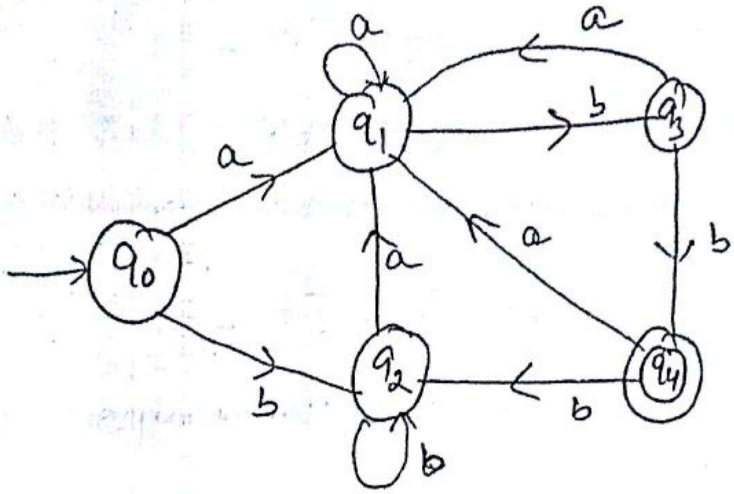
Step I → Construction of π_0 :- By defination of 0-equivalence $\pi_0 = \{Q_1^0, Q_2^0\}$, where Q_1^0 is the set of all final states and $Q_2^0 = Q - Q_1^0$ (set of non-final states)

Step 2 → Construction of π_{k+1} from π_k :- Let Q_i^k be any subset in π_k. If q_1 and q_2 are in Q_i^k, they are $(k+1)$-equivalent provided $\delta(q_1, a)$ and $\delta(q_2, a)$ are k-equivalent. Find out whether $\delta(q_1, a)$ and $\delta(q_2, a)$ are the same equivalence class in π_k for e[illegible] $a \in \Sigma$. If so, then q_1 and q_2 are $(k+1)$-equ[illegible]

Step 3 Construct π_n for $n = 1, 2 - - -$ untill $\pi_n = \pi$[illegible]

Step 4 :- Construction of minimum automation :- f[illegible] the required minimum state automaton, [illegible] state are the equivalence class obtained [in] step 3.

Example I: Construct a minimum state automaton equivalent to the finite automation as.



Answer: Construct the Transition Table as:

States	a	b
$\rightarrow q_0$	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

$Q_1^0 = F = \{q_4\}$, $Q_2^0 = \{q_0, q_1, q_2, q_3\}$

$\therefore$ $\Pi_0 = \{\{q_0, q_1, q_2, q_3\} \{q_4\}\}$ 0 - equivalent

Now check the 1-equivalence of all the states with each other. first check whether q_0 & q_1 are equivalent or not? By giving the input as 'a' to q_0 & q_1 $\delta(q_0, a) = q_1$ $\delta(q_1, a) = q_1$ & with input 'b', $\delta(q_0, b) = q_2$ $\delta(q_1, b) = q_3$. As are the out comes are belongs to the same set in Π_0. $\therefore$ q_0 & q_1 are 1-equivalent to each other.

$$\Pi_1 = \{q_0, q_1\}$$

Now check the equivalence of q_0 & q_2 with in a & b as.

$\delta(q_0, a) = q_1$ && $\delta(q_0, b) =$
$\delta(q_2, a) = q_2$ $\quad$ $\delta(q_2, b) = q$

the outcomes of both the Transition belong to the same set $\therefore$ q_0 & q_2 are also 1-equivalent

$\therefore$

$$\Pi_1 = \{q_0, q_1, q_2\}$$

Now check the equivalence of q_0 & q_3 with input a & b as:

$\delta(q_0, a) = q_1$ $\delta(q_0, b) = q_2$

$\delta(q_3, a) = q_1$ $\delta(q_3, b) = q_4$

from above we get with the 'a' input the outcomes q_1, q_1 belong to the same set in Π_0 but with input 'b' the outcome q_2, q_4 belongs to the different set in Π

so we can say that they are (q_0, q_3) not 1-equivalence to each other.

$\therefore$

(1-equivalence) $\Pi_1 = \{\{q_0\ q_1\ q_2\}\ \{q_3\}\ \{q_4\}\}$

Now find the 2-equivalence as Π_2 from above we get:

(2-equivalence) $\Pi_2 = \{q_0, q_2\}\ \{q_1\}\ \{q_3\}\ \{q_4\}$

Now find the 3-equivalence as Π_3 from above we get:

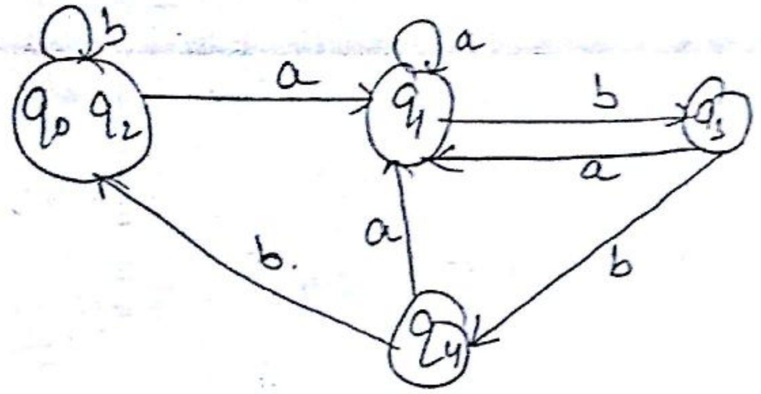
(3-equivalence) $\Pi_3 = \{q_0, q_2\}\ \{q_1\}\ \{q_3\}\ \{q_4\}$

Now, from above we see $\Pi_2 = \Pi_3$ $\therefore$ stop here & got the final answer as

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

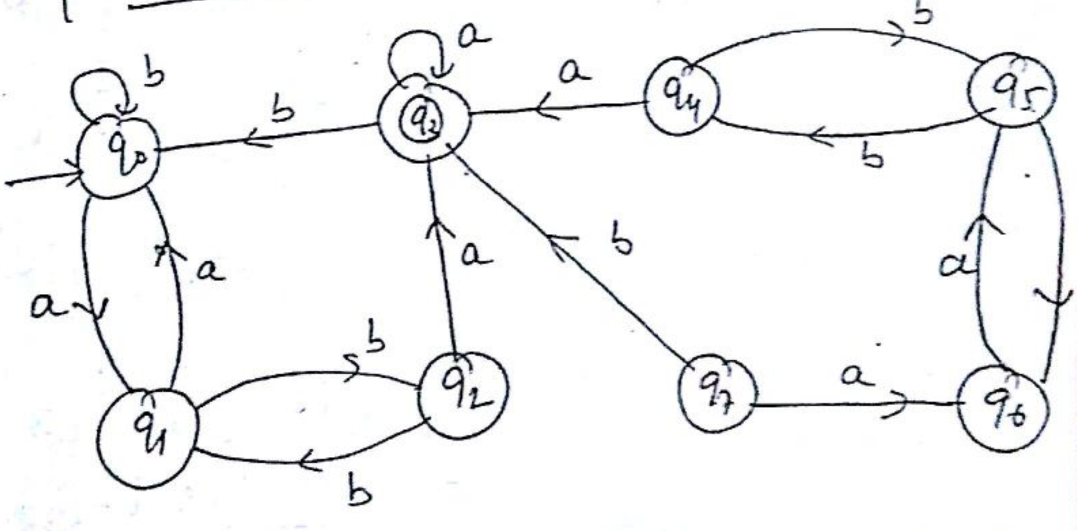
Scanned by CamScanner

So, with this result we found that state $\{q_0, q_2\}$ are merged together & minimise the given DFA as:



Answer

Example 2:- Construct the minimum state automaton equivalent to the transition diagram given.



Answer: Construct the transition - Table of above

State	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_0	q_2
q_2	q_3	q_1

	a	b
*q_3	q_3	q_6
q_4	q_3	q_5
q_5	q_6	q_4
q_6	q_5	q_6
q_7	q_6	q_3

0-equivalence $\Pi_0 = \{\{q_0, q_1, q_2, q_4, q_5, q_6, q_7\}, \{q_3\}$

1-equivalence $\Pi_1 = \{\{q_0, q_1, q_5, q_6\} \{q_2, q_4\} \{q_3\}\}$

2-equivalence $\Pi_2 =$ q_0 is 2-equivalent to q_6 but not to q_1 or q_5, ∴

$$\Pi_2 = \{\{q_0, q_6\} \{q_1, q_5\} \{q_2, q_4\} \{q_7\}\}$$

3-equivalence ⇒ As q_0 is 3-equivalence to q_6 &
q_1 is 3-equivalence to q_5 &
q_2 is 3-equivalence to q_4

∴

$$\Pi_3 = \{\{q_0, q_6\} \{q_1, q_5\} \{q_2, q_4\} \{q_7\}\}$$

As $\Pi_3 = \Pi_2$, Π_2 gives us the equivalence cl
the minimum state automaton is

$M' = (Q', \{a, b\}, \delta', q_0', F')$ where.

Scanned by CamScanner

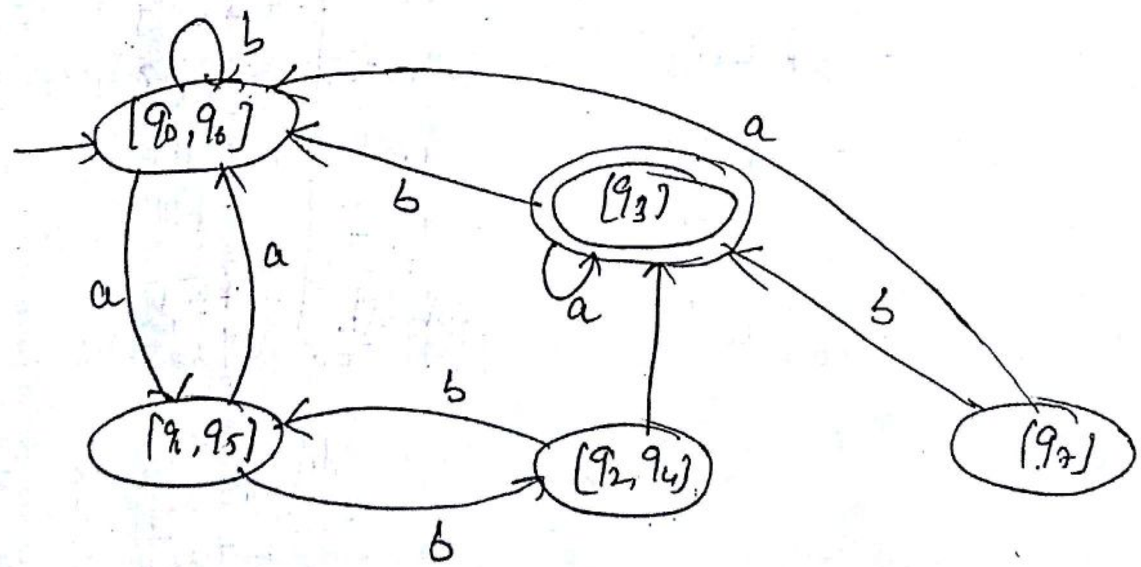
$Q' = (\{q_0\, q_6\}, \{q_1, q_5\}, \{q_2, q_4\}\ \{q_7\})\ q_3'$

$q_0' = [q_0, q_6]$, $F' = [q_3]$

Transition table of minimum state Automation

State/Σ	a	b
$[q_0, q_6]$	$[q_1, q_5]$	$[q_0, q_6]$
$[q_1, q_5]$	$[q_0, q_6]$	$[q_2, q_4]$
$[q_2, q_4]$	$[q_3]$	$[q_1, q_5]$
$[q_3]$	$[q_3]$	$[q_0, q_6]$
$[q_7]$	$[q_0, q_6]$	$[q_3]$

Transition Graph:-



Answer.

Example: Construct a minimum state automaton equivalent to DFA whose transition table is defined as

δ/Σ	a	b
$\rightarrow q_0$	q_1	q_2
q_1	q_4	q_3
q_2	q_4	q_3
q_3	q_5	q_6
q_4	q_7	q_6
q_5	q_3	q_6
q_6	q_6	q_6
q_7	q_4	q_6.

Answer: $\Pi_0 = \{\{q_0, q_1, q_2, q_5, q_6, q_7\} \{q_3, q_4\}\}$

q_0 is not 1-equivalent to q_1, q_2, q_5, q_7 but it is 1-equivalent to q_6. & q_3 & q_4 are 1-equivalent

$\therefore \quad \Pi_1 = \{$

Hence $\{q_0, q_6\} \in \Pi_1$. As q_1 is 1-equivalent to q_2 but not 1-equivalent to q_5 q_6, q_7. So $\{q_1, q_2\}$

As q_5 is not 1-equivalent to q_6 but to q_7. So $\{q_5, q_7\}$

$\therefore \quad \Pi_1 = \{\{q_0, q_6\} \{q_1, q_2\} \{q_5, q_7\}, \{q_3, q_4\}\}$ —— (i)

Now check for 2-equivalent in eq. (i)

q_3 is 2-equivalent to q_4, so $\{q_3, q_4\} \in \Pi_2$

q_0 is not 2-equivalent to q_6 . So $\{q_0\}, \{q_6\} \in \Pi_2$

q_1 is 2-equivalent to q_2 . So $\{q_1, q_2\} \in \Pi_2$

q_5 is 2-equivalent to q_7 . So $\{q_5, q_7\} \in \Pi_2$

Hence.

$$\Pi_2 = \{ \{q_0\}, \{q_6\}, \{q_1, q_2\}, \{q_5, q_7\} \{q_3, q_4\} \}$$

Now, check for 3-equivalent from eq (ii)

q_3 is 3-equivalent to q_4 , so $\{q_3, q_4\} \in \Pi_3$

q_1 is 3-equivalent to q_2 , so $\{q_1, q_2\} \in \Pi_3$

q_5 is 3-equivalent to q_7 ; So, $\{q_5, q_7\} \in \Pi_3$

Hence $\Pi_3 = \{ \{q_0\}, \{q_6\}, \{q_1, q_2\}, \{q_5, q_7\}, \{q_3, q_4\} \}$

As $\Pi_3 = \Pi_2$, the minimum automaton is

$$M' = Q', \{a, b\}, \delta', [q_0], [\{q_3, q_4\}])$$

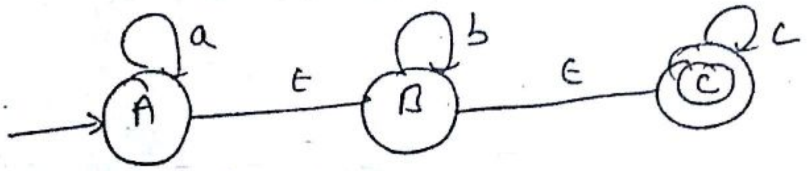
Where δ' is define as!

δ / Σ	a	b
$[q_0]$	$[q_1, q_2]$	$[q_1, q_2]$
$[q_1, q_2]$	$[q_3, q_4]$	$[q_3, q_4]$
$*[q_3, q_4]$	$[q_5, q_7]$	$[q_6]$
$[q_5, q_7]$	$[q_3, q_4]$	$[q_6]$
$[q_6]$	$[q_6]$	$[q_6]$

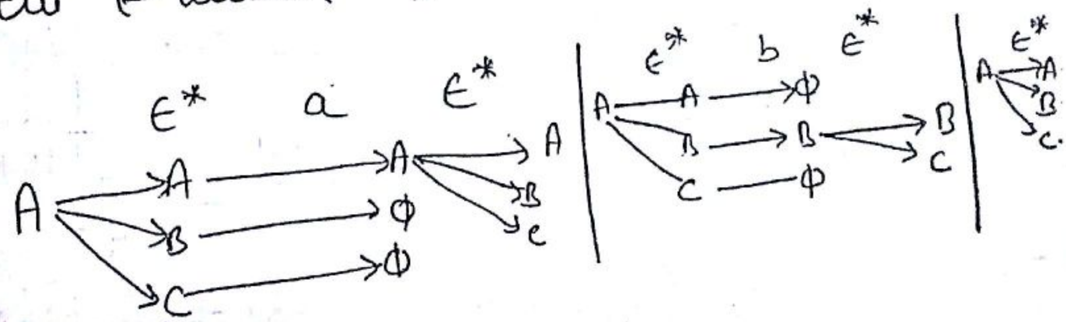
Answer

CONVERT NFA-ε TO NFA

Example 1 :- Consider the NFA with ε-Transition. Convert it in to NFA without ε-moves.



Ans:- 1). find out the ε moves of 'A' & then find the moves with a, b, c & then again find the ε moves. i.e



∴

	a	b	c
A	{A,B,C}	{B,C}	{C}

2) Now find the ε-moves of 'B' state & then find the moves with a, b, c input & then again finds the ε-moves.

$B \xrightarrow{\epsilon^*} B \xrightarrow{a} \phi \xrightarrow{\epsilon^*} \phi$

$B \xrightarrow{\epsilon^*} C \xrightarrow{a} \phi \xrightarrow{\epsilon^*} \phi$

$B \xrightarrow{\epsilon^*} B \xrightarrow{b} B \xrightarrow{\epsilon^*} B, C$

$B \xrightarrow{\epsilon^*} C \xrightarrow{b} \phi$

$B \xrightarrow{\epsilon^*} B \xrightarrow{c} \phi$

$B \xrightarrow{\epsilon^*} C \xrightarrow{c} C \xrightarrow{\epsilon^*} C$

∴

	a	b	c
{A}	{A,B,C}	{B,C}	{C}
{B}	ϕ	{B,C}	{C}

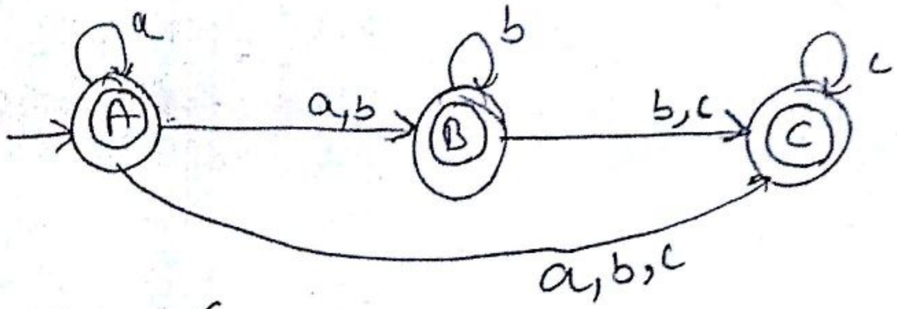
(3) Now, find out the ε-moves of 'c' state & then find the moves with a, b, c input & again then find the ε-closer. we get as:

$C \xrightarrow{\epsilon^*} C \xrightarrow{a} \phi \xrightarrow{\epsilon^*} \phi$ | $C \xrightarrow{\epsilon^*} C \xrightarrow{b} \phi \xrightarrow{\epsilon^*} \phi$ | $C \xrightarrow{\epsilon^*} C \xrightarrow{c} C \xrightarrow{\epsilon^*} C$

∴

	a	b	c
A	{A,B,C}	{B,C}	{C}
B	ϕ	{B,C}	{C}
C	ϕ	ϕ	{C}

∴ the required NFA without null m
is shown as:



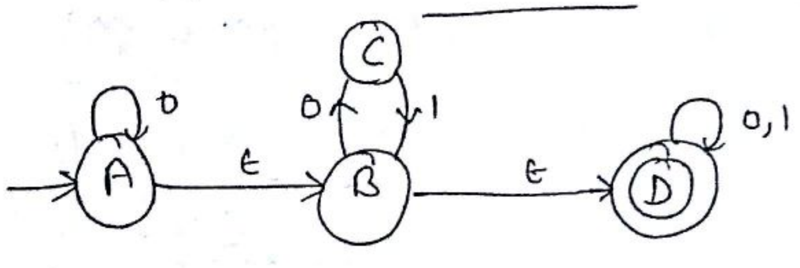
(NFA without ϵ-Moves)

Final state is choosen as consider all the state which are connecting with the final states directly with ϵ-moves.

∴ A & B states are directly connected with the final states with ϵ-moves.

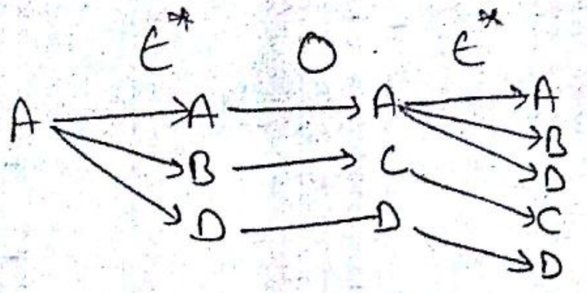
∴ A, B & C are final states.

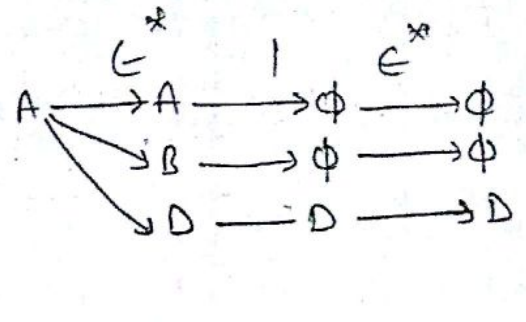
Example 2. Consider a NFA-ϵ, convert it into NFA without ϵ-moves.



Ans:- ① Find out the ϵ-moves of 'A' state then find the moves with (0, 1) in & then again find out the ϵ-mo[ves]

we get as:

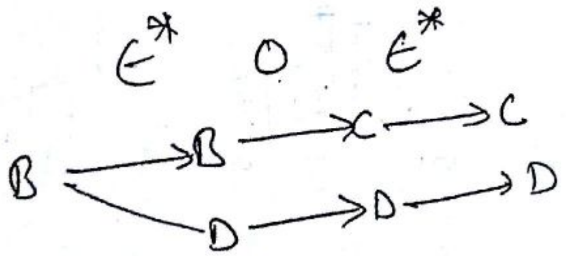


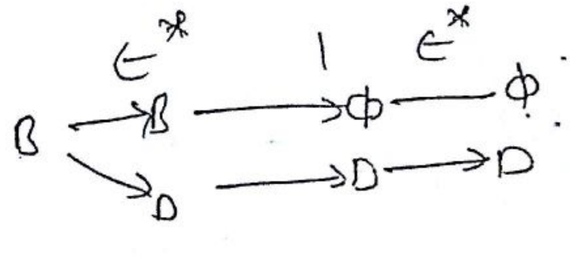


∴

	0	1
→ {A}	{A,B,C,D}	{D}.

(2) Now, find the ε-moves of state B
then find the moves with input {0,1}
& then find the ε-closure. we get as:





∴

	0	1
{A}	{A, B, C, D}	{D}
B	{C, D}	{D}.

③ Now, find the ϵ-moves of state 'c' & then find the moves with input {0,1} & then find the ϵ-closer, we get as:

$$C \xrightarrow{\epsilon^*} C \xrightarrow{0} \phi \xrightarrow{\epsilon^*} \phi$$

$$C \xrightarrow{\epsilon^*} C \xrightarrow{1} B \xrightarrow{\epsilon^*} B, D$$

	a	b
A	{A, B, C, D}	{D}
B	{C, D}	{D}
C	ϕ	{B D}

4. Now, find the ϵ-move of state 'D' & then find the moves with input {0,1} & then find the ϵ-closer, we get as:

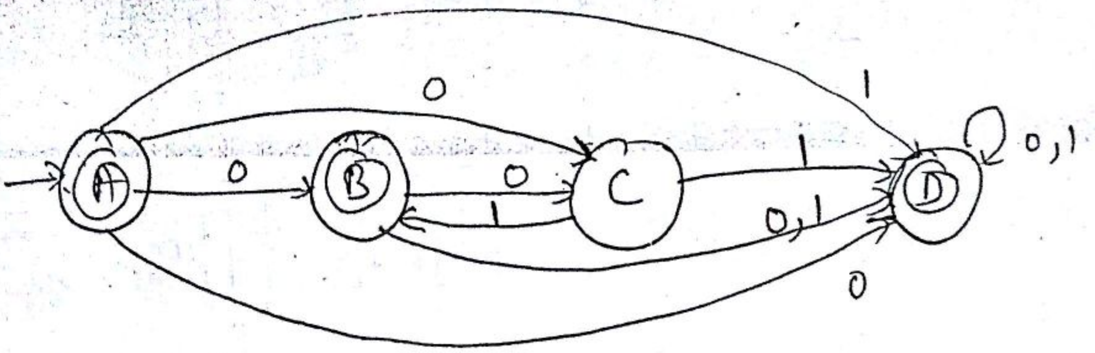
$$D \xrightarrow{\epsilon^*} D \xrightarrow{0} D \xrightarrow{\epsilon^*} D$$

$$D \xrightarrow{\epsilon^*} D \xrightarrow{1} D \xrightarrow{\epsilon^*} D$$

	a	b
A	{A, B, C, D}	{D}
B	{C, D}	{D}
C	ϕ	{B, D}
D	{D}	{D}

Scanned by CamScanner

∴ required NDFA is



(NFA without ε-moves)

Make the A B & D as final state ∵ there is direct ε-move from D to B & A.

Scanned by CamScanner

KLEENE'S THERORЕМ

M.Imp

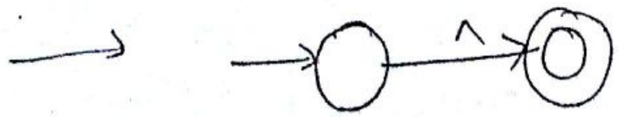
Statment :- Any regular language can be accept
by a finite Automation.

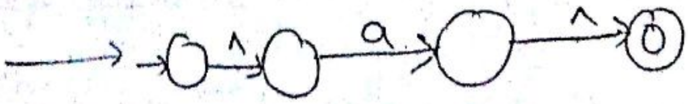
Proff :- → To prove above it is sufficent to
show that every regular language can b
accepted by NFA-Λ.

→ The set of regular language over the alph
Σ indefined contain the basic language
φ , {Λ} and {a} a∈Σ. are closed under
operation of UNION , concatenation and k

→ We prove this all by using structural i
that every regular language can be acc
by NFA-Λ.

① φ 

2) {ε} 

3) {a} 

→ The induction hypothesis shows that the langu L₁ & L₂ can be accepted by NFA-Λ.

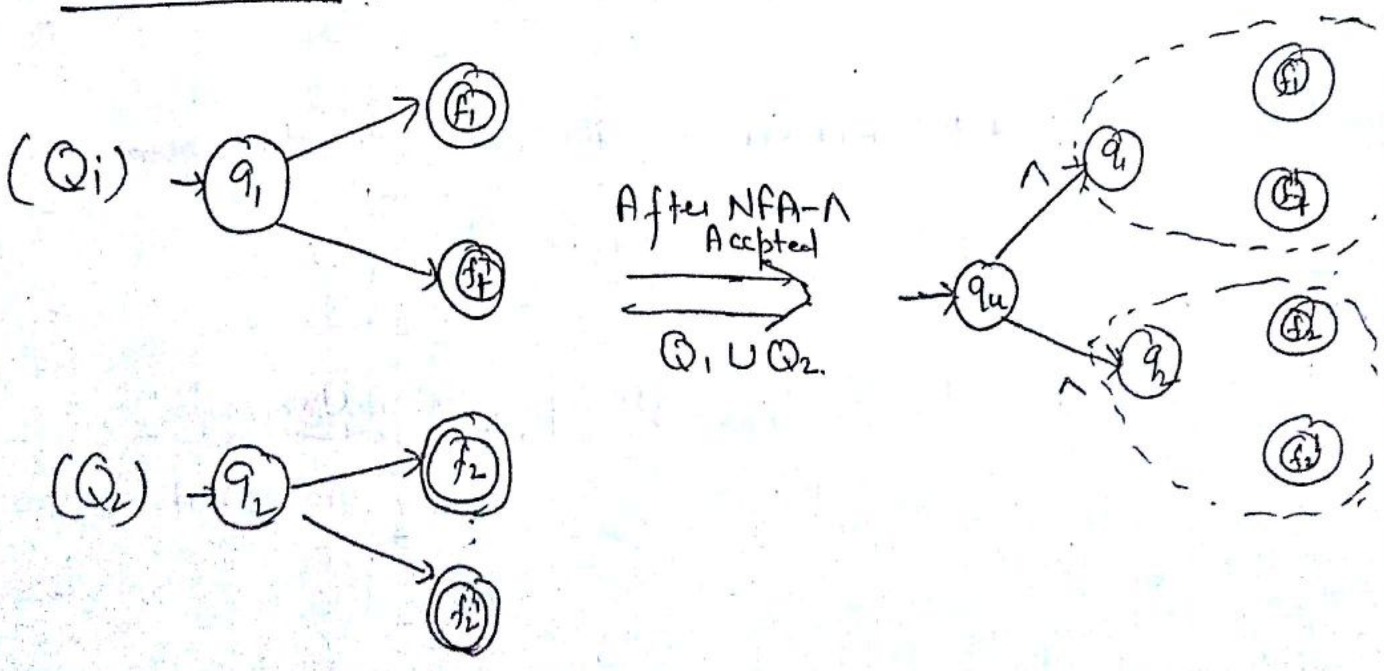
→ To prove operation UNION, concatenation & kleene's we suppose that L_1 & L_2 are recognised by the NFA-Λ as M_1 and M_2 machine where both $i=1$ & $i=2$ as:

$$M_i = (Q_i, \Sigma, q_i, A_i, S_i)$$

→ Let M_U, M_c, M_k represent the language $L_1 \cup L_2$, $L_1 \cdot L_2$, L_1^* accepted by NFA-Λ.

① Construction of M_U :→

(M/c for UNION) $M_U = (Q_U, \Sigma, q_u, A_u, S_0)$.



where $Q_u = Q_1 \cup Q_2 \cup \{q_u\}$

$A_u = A_1 \cup A_2$

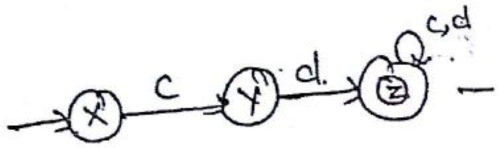
$\delta_u = \delta_u(q_u, \Lambda) = \{q_1, q_2\}$, $\delta_u(q_u, a) = \phi$

and for each $q \in Q_1 \cup Q_2$ and $a \in \Sigma \cup \{\Lambda\}$

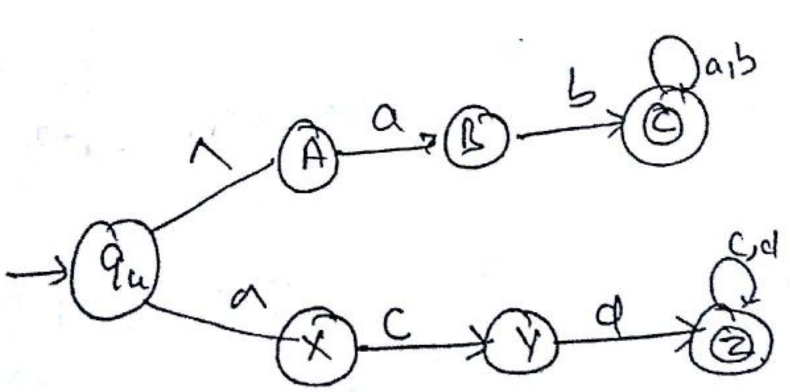
$$\delta_u(q, a) = \begin{cases} \delta_1(q, a) & \text{if } q \in Q_1 \\ \delta_2(q, a) & \text{if } q \in Q_2 \end{cases}$$

for Example :- start with 'ab' NFA is

Start with 'cd' NFA is



If we want to take the UNION of above (i) & (ii), then, it add the initial state with Λ-move as.



Fig(1)

L_1 = { start with 'ab' } , L_2 = { start with c,d. }

L_1 = { ab, abb, aba, abab--- } L_2 = { cd, cdc, cdd, cddc

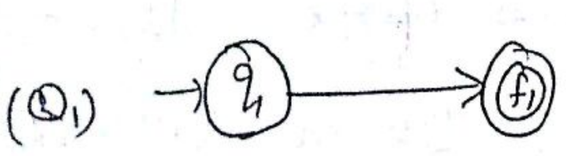
When we take the $L_1 \cup L_2$ then the lo accept all the string that are in L_1 & L_2

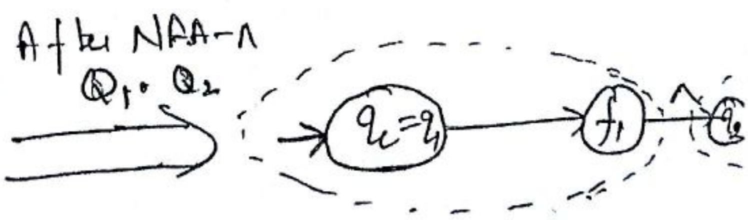
$L_1 \cup L_2$ = {ab, cd, aba, abb, cdc, cdc

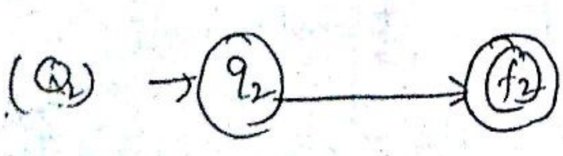
Corresponding to this language $L_1 \cup L_2$, if we can design a NFA-Λ, then we can say that the language is regular. & By making the NFA in previous fig.(1) by using the NF that accept all the ~~language~~ string that under the language $L_1 \cup L_2$.

2 Construction of M_c :-

$$M_c = (Q_c, \Sigma, q_c, A_c, \delta_c)$$







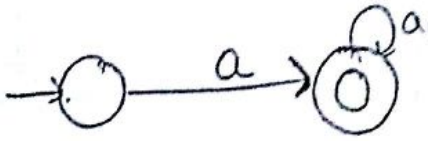
where $Q_c = Q_1 \cup Q_2$

$q_c = q_1$

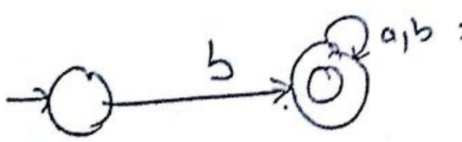
$A_c = A_2$ or f_2

$\delta_c(q, a) = \delta_1(q, a)$ for every $a \in \Sigma$ and

$\delta_c(q, \Lambda) = \delta_1(q, \Lambda) \cup \{q_2\}$

For Example :- $Q_1 =$ 

$= L_1 = \{a, aa, ab --$

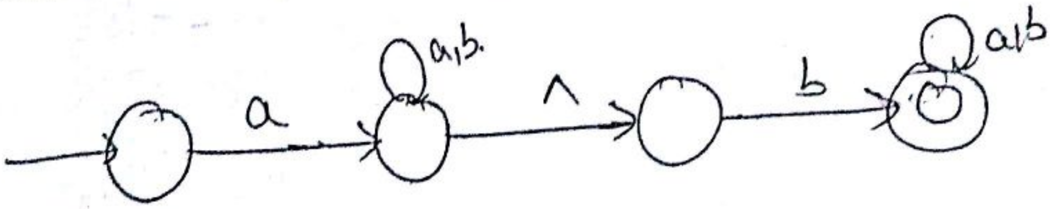
$Q_2 =$ 

$= L_2 = \{b, bb, ba, b$

$L_1 \cdot L_2 = \{a, aa, ab \ldots\} \{b, bb, ba, bb --\}$

$= \{ab, abb, aba, abb, aab, aabb --\}$

$Q_1 \cdot Q_2$ with Λ-moves as :

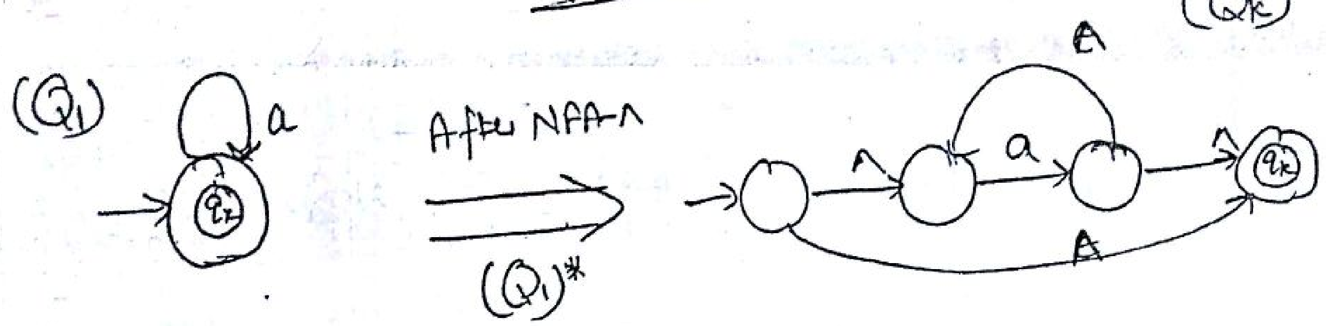


The above NFA-Λ accept all the strings which are under $L_1 \cdot L_2$. ∴ we can say that the language is closed under the concatenation ope

& the $L_1 \cdot L_2$ is regular language.

3) Construction of Kleen's closure:→

$M_K = (Q_K, \Sigma, q_K, A_K, \delta_K)$



Where $Q_K = Q_1 \cup \{q_K\}$

$A_K = \{q_K\}$

Suppose $x \in L_1^*$ ① if $x = \Lambda$, then clearly x accepted by Q_K..

② If $x = x_1 x_2 \cdots x_m$, it follows $\Lambda x_1 \Lambda x_1 \Lambda x_1 \Lambda = x$. is accepted by M_K.

(3) If $x = x_1$ then it follow the $\Lambda x_1 \Lambda = x$ is accepted by M_K.

Since the Construction of NFA-Λ that recogni L in each of three Cases, the proof is Complete

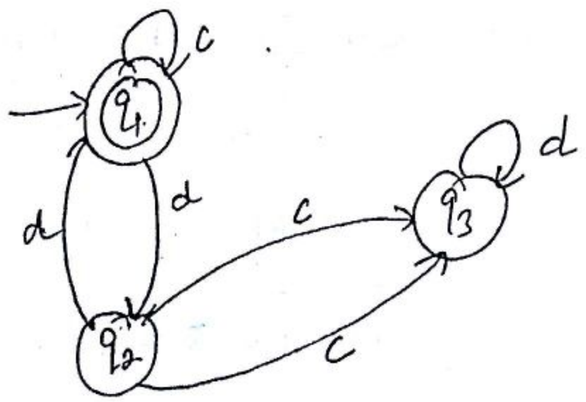
(Hence proved)

Scanned by CamScanner

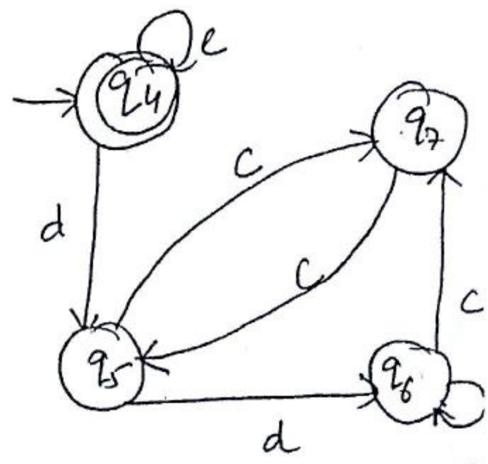
EQUALITY OF 2 FA's

→ Two FA's are not equal if the final state of one FA pairs with the non-final state of another FA.

Example I: Consider two FA's, prove whether they are equivalent or not?



(FA-I)



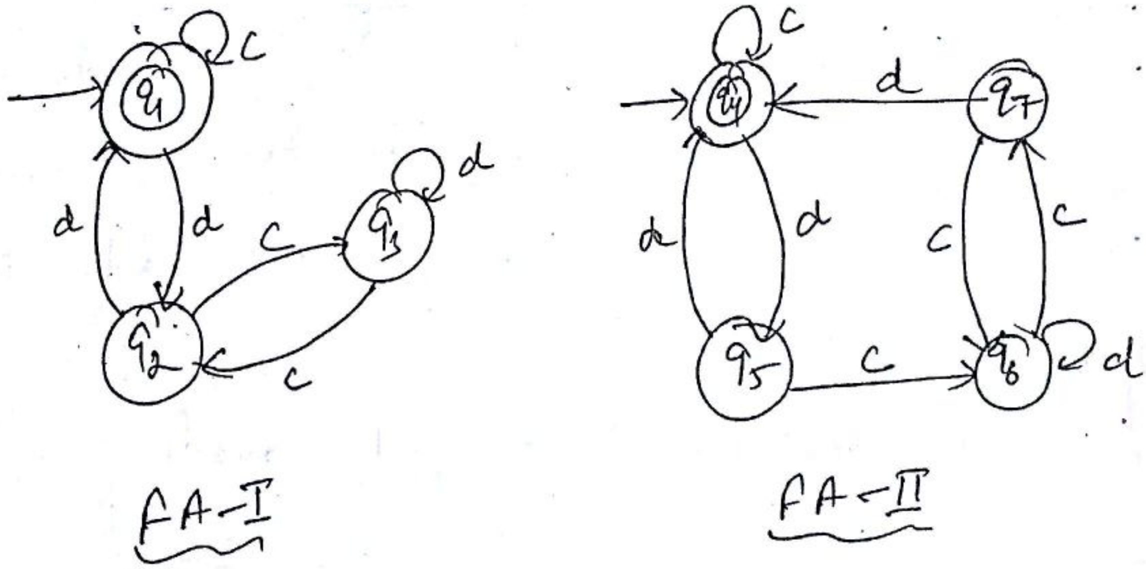
FA-2

Ans:

	c	d
→(q_1, q_4)	(q_1, q_4)	(q_2, q_5)
(q_2, q_5)	(q_3, q_7)	(q_1, q_6)

Because q_1 is the final state of FA-1 and it is in the pair with q_8 which a non-final state. So two FA's are not equal. Final state of FA_1 should only pa with final state of FA 2.

Example 2: Prove two FA's are equivalent?



Ans:

	c	d
$\{q_1\ q_4\}$	$\{q_1, q_4\}$	$\{q_2, q_5\}$
$\{q_2\ q_5\}$	$\{q_3, q_6\}$	$\{q_1, q_4\}$
$\{q_3\ q_6\}$	$\{q_2, q_7\}$	$\{q_3, q_6\}$
$\{q_2, q_7\}$	$\{q_3, q_6\}$	$\{q_1, q_4\}$

final state of FA-I is in pair only wi final state of FA-2. Hence 2 FA" are equal.

LIMITATIONS OF FSM :-

1) The memory is limited and read only. A f
automata can read only from the input tap

2) only string recognizing power is there.
decides the validity of the input by reach
the final state.

3) The FSM cannot move backward, hence
can not recall its previous input

4) There is no output writing capability. f
finite automaton cannot write its output
It uses some fixed set of outputs to sl
its result.

5) Control instructions are finite. It is simi
to the control of a computer system, wl
is also finite

6) It is formal description of a simple
automatic machine.

Application Of FSM :-

1) Lexical Analyzers :- It is a part of compil and used to recognize the validity of i program, whether the input program is grammatical constructed or not.

2) Text Editors :- Smart and speedy Text editors co constructed using FA.

3) Spell Checkers :- They are designed using f. In our computer system, there is a dictioni and FA can recognize the correctness of a wor & give the appropriate suggestion to the user w data from the dictionary.

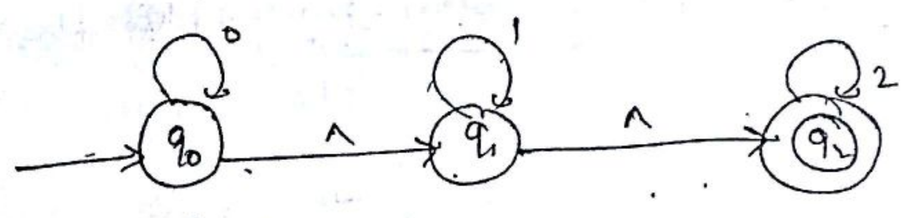
4) Sequential Circuits Design :- FA is used in sequential circuit design.

ELIMINATE Λ-MOVES from N

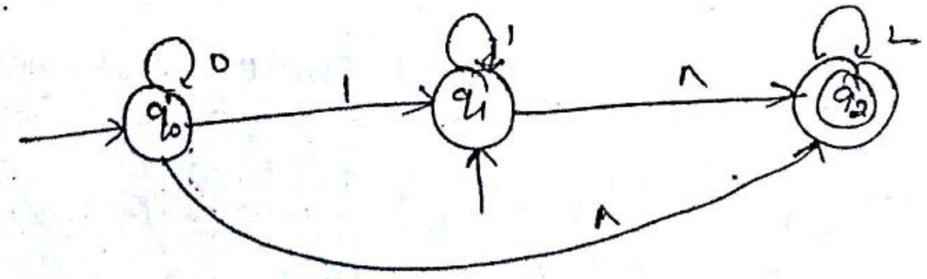
Procedure:- Suppose we want to replace Λ-move from Vertex V_1

1) Find all the edges starting from V_2.
2) Duplicate all these edges starting from V_1, wo changing the edge labels.
3) If V_1 is an initial state, make V_2 also as initial state.
4) If V_2 is a final state, make V_1 also as final state.

Example:- Eliminate Λ-moves from given:

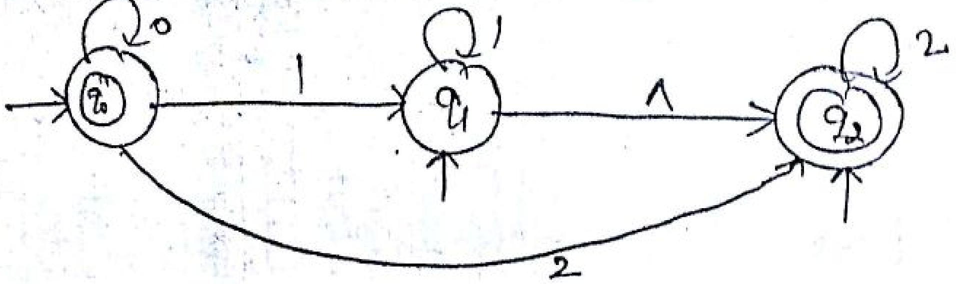


Ans: 1) First eliminate Λ-moves from q_0 to q_1,

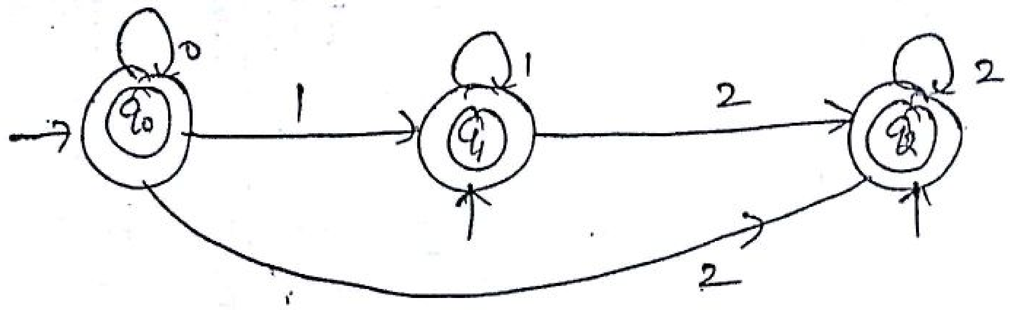


2) Now, eliminate Λ-moves from q_0 & q_2

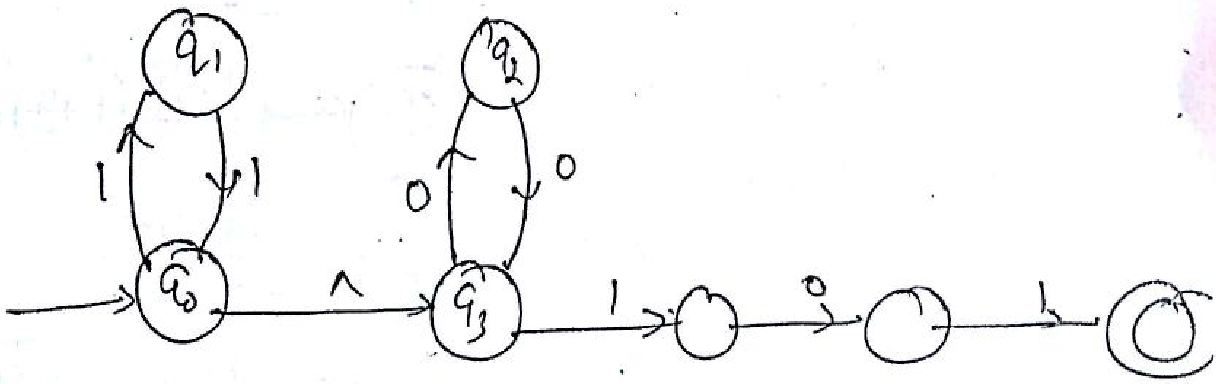
we get!



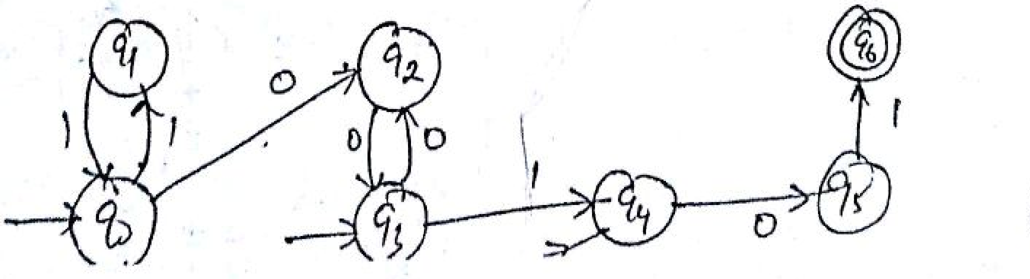
3) Now, eliminate Λ-move from q_1 to q_2,



Example:2. Eliminate Λ-move from given!



Answ! There is null moves from q_0 to q_3, so eliminate, we get as!



(Answ

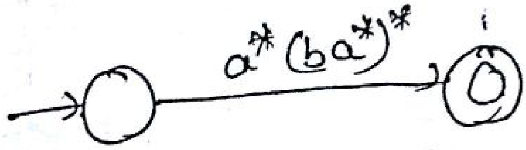
1) Prove that:

$$(a+b)^* = a^*(ba^*)^*$$

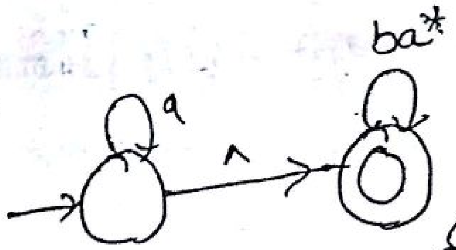
Ans: Construct a FA of both the R.E & prove the equivalent of two FA. Then we say $(a+b)^* = a^*(ba^*)^*$

Step I: Construct FA for $a^*(ba^*)^*$

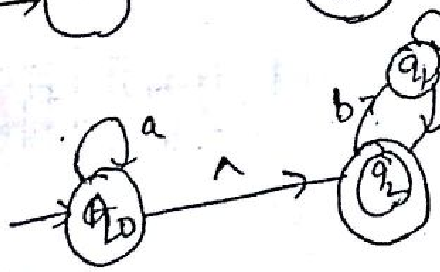
1)



2)

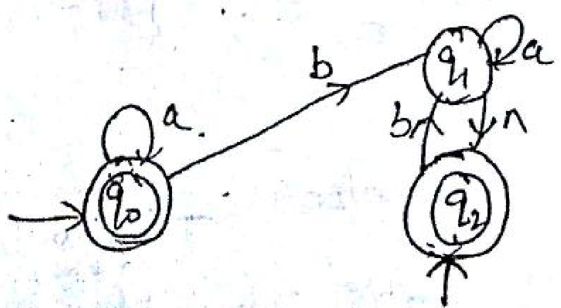


3)

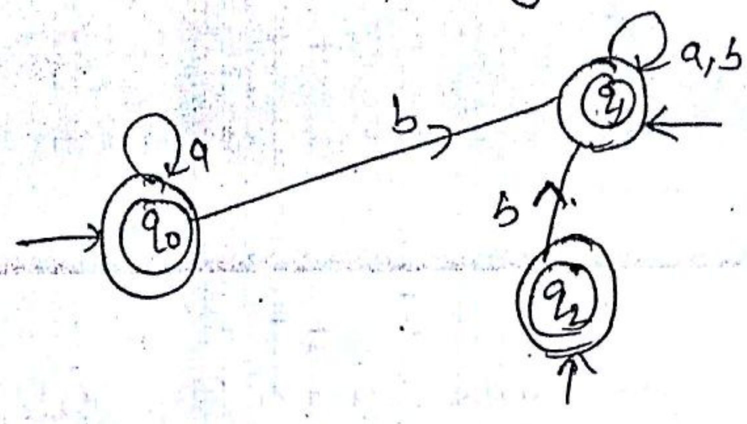


As, above contain the Λ-moves, so, for we eliminate the Λ-moves as:

Step I: First we eliminate the Λ-move between q_1 & q_2, we get.



Step 2 :- Now, eliminate $\wedge$ move between q_1 & q_2, we get,



Now, make the Transition Table of above

	a	b
*[q_0 q_1 q_2]	[q_0 q_1]	[q_1]
*[q_0 q_1]	[q_0 q_1]	[q_1]
*[q_1]	[q_1]	[q_1]

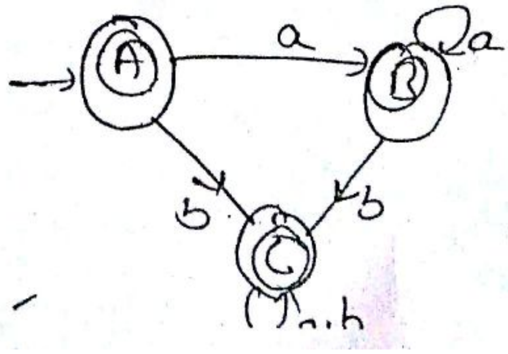
Suppose $\longrightarrow$ [q_0 q_1 q_2] $\Rightarrow$ A

[q_0 q_1] $\Rightarrow$ B

[q_1] $\Rightarrow$ C

	a	b
*A	B	C
*B	B	C
*C	C	C

$\Longrightarrow$

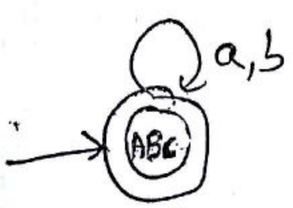


Now, Minimise the above FA, we get

$\Pi_0 = \{A, B, C\}$

$\Pi_1 = \{A, B, C\}$

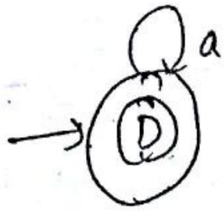
As $\Pi_1 = \Pi_0$ ∴ we combine state A, B & C & form a new string state as:



$\Rightarrow$

	a	b
*[ABC]	[ABC]	[AB

Now, take the L.H.S of expression $(a+b)^*$ after its construction, we will get the FA as.



$\Rightarrow$

	a	b
*[D]	[D]	[D]

Now, To prove these two FA's are equi
Merge the final state of Both FA's &
Check, the final states are in same pa
then we say two FA's are equivalent
∴ two R.E are same

	a	b
[[ABC], [D]]	[[ABC], (D)]	[[A BC], [D]]

So, from above, we get the final states two FA are in same pair. So we can say that these two FA's are equivalent & we also says two given R.E are also equivalent to each other.

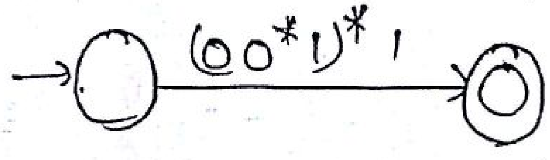
Prove that:-

1) $(00^*1)^*1 = 1+0(0+10)^*11$

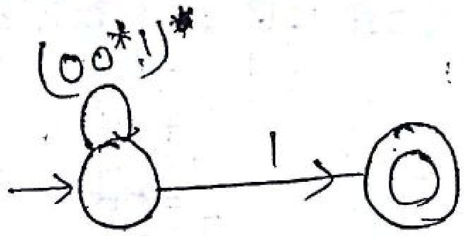
Ans: $(00^*1)^*1$

step 1 Design a FA of R.E $(00^*1)^*1$ as:

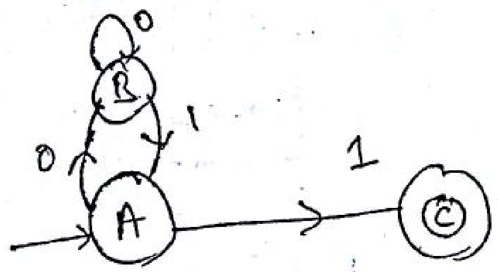
1)



2)



3)



Now apply Arden's theorem on above FA, o

$A = \Lambda + B\cdot 1$ —(i)

$B = A\cdot 0 + B\cdot 0$ —(ii)

$C = A\cdot 1$ —(iii)

As, 'c' state is final state, so we re

the regular Expression in 'c' state only..

Put the value of 'A' in 'B', we get

$$B = (\Lambda + B \cdot 1)\, 0 + B \cdot 0$$

$$= 0 + B \cdot 10 + B \cdot 0$$

$$\underset{\uparrow\atop R}{B} = \underset{\uparrow\atop Q}{0} + \underset{\uparrow\atop R}{B}\,(\underset{\uparrow\atop P}{10+0})$$

{Arden's theorem}

$$B = 0\,(10+0)^*$$

Put the value of B in 'A', we get:

$$A = \Lambda + 0\,(10+0)^*\, 1$$

Put the value of 'A' in 'c', we get

$$C = (\Lambda + 0\,(10+0)^*\, 1)\, 1$$

$$= 1 + 0\,(10+0)^*\, 11$$

$\therefore$ from the left hand side, we desire the right hand side of R.E. $\therefore$ We can say

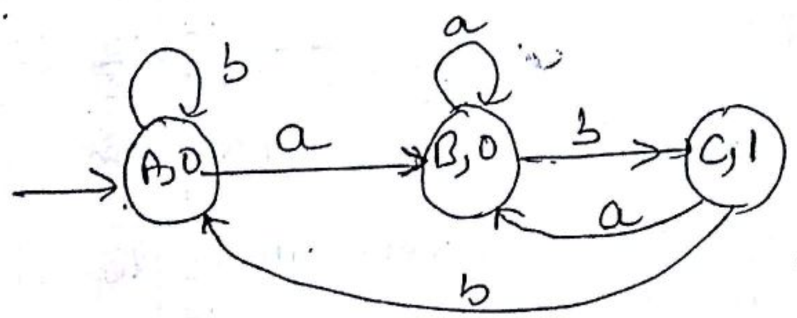
(3) Construction of Moore & Mealy M/c

Example I:- Construct a moore M/C that takes s of all strings over {a,b} as i/p and print '1' as o/p for every occurrance of 'ab' a a substring.

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

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

Now, we will construct a FA that ends with "ab". Because, we need to count the last occurrence of 'ab' on Moore machine as:



Now Calculate the output of given a string as 'ab' on above:

$A \xrightarrow{a} B \xrightarrow{b} C$

$\downarrow \quad \downarrow \quad \downarrow$

0 0 1 (outputs)

∴ To accept a string 'ab', the moore will produce a output as 001 Ans.

Scanned by CamScanner

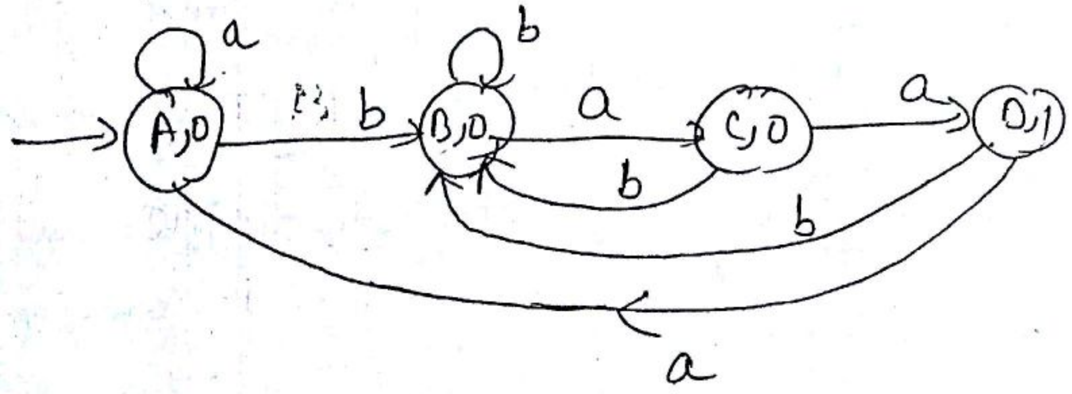
Example 2. Construct a moore m/c that takes set of all strings over $\{a, b\}$ and counts no. of occurrences of substring 'baa'.

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

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

Now, we construct a FA that ends a 'baa' ∴ it will count every occurrence of 'baa' string & corresponding to this it will print value as '1'. f. E. if the string is forms baa, baa, ∴ output is 11.

Design of Moore :-



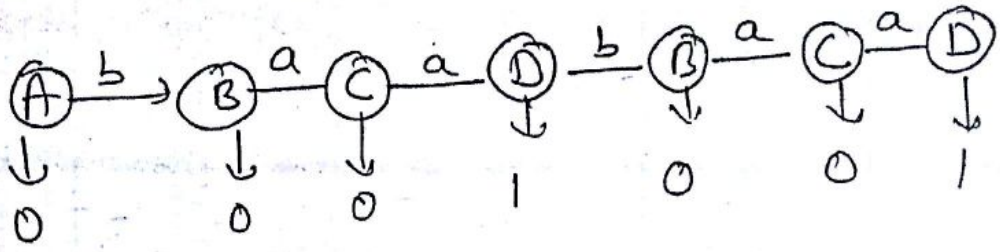
Now suppose the input string is baa then output is:

$A \xrightarrow{b} B \xrightarrow{a} C \xrightarrow{a} D$

(output)

∴ Required output is 0 0 0 1.

f·E accept a string baa baa, Now the output produce is:



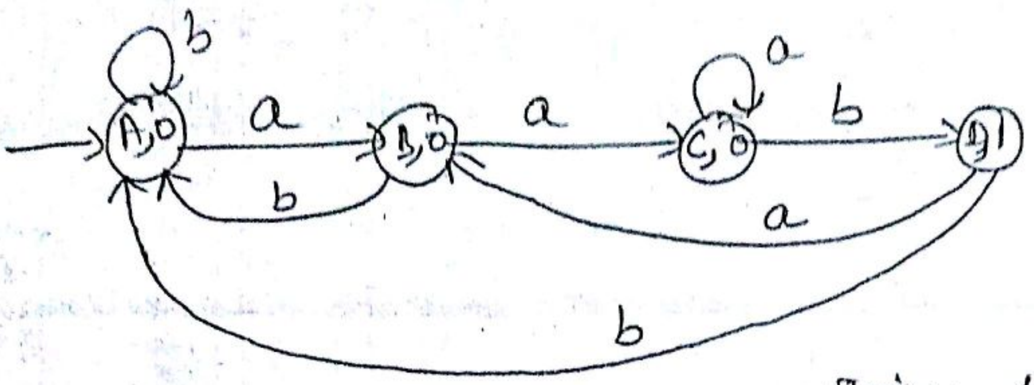
∴ Required output is 0001 001. Ans

Imp. Example 3: Design a Moore's Machine which will count how many times substring 'aab' occurs in a long input string. Count be maintained by printing 1 each time 'aab' occurs.

Ans:
$\Sigma = \{a, b\}$
$\Delta = \{0, 1\}$

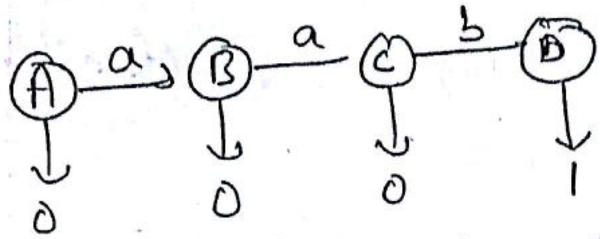
Now, we need to construct a DFA which accept a string that ends with 'aab'. & print the value as '1' if any number of time aab will occur. f.E if the string is given as aab aab then the m/c will produce the output as 11.

Design of Moore m/c :-



Now suppose the input string is aa.

then output is:



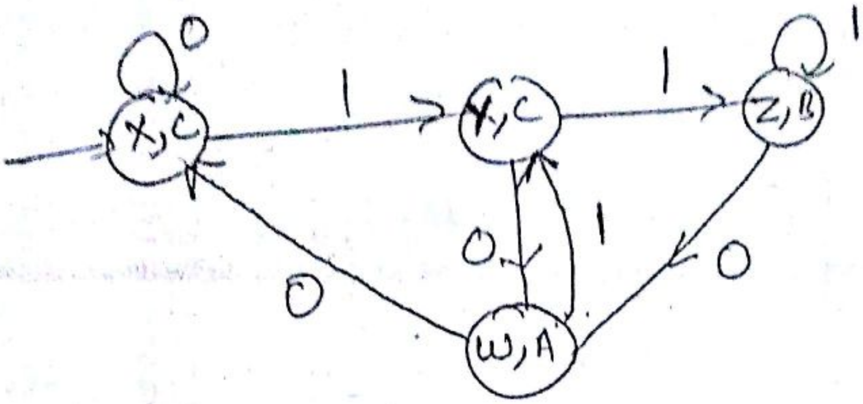
0001 Ans

$\therefore$ Required output is 0001 Ans

Example: 1 Construct a Moore m/c that Tak. set of all strings over {0,1} and pr 'A' as o/p if Input ends with '10' produce 'B' as o/p if I/P ends with otherwise produce 'C'.

Ans: $\Sigma = \{0,1\}$
$\Delta = \{A, B, C\}$

Design of Moore M/C:-



i.e, accept a string 111 & the outp[ut] is :

$X \xrightarrow{1} Y \xrightarrow{1} Z \xrightarrow{1} Z$

C   C   B   B (output)

accept a string 110 & the output is

$X \xrightarrow{1} Y \xrightarrow{1} Z \xrightarrow{0} W$

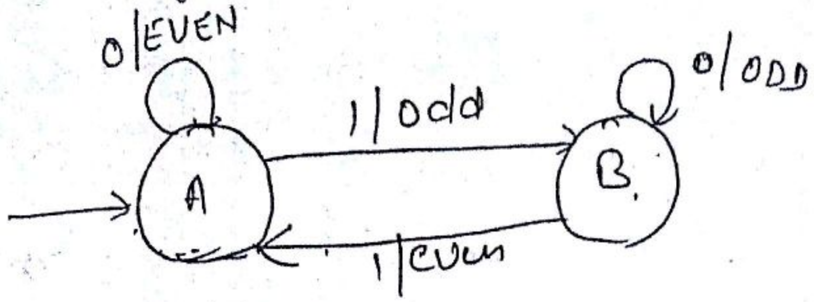
C   C   B   A (output)

Answer

Example 5: Construct a Mealy Machine which ca[n] output EVEN, ODD according as the total number of 1's encountered is even or odd. The input symbols are 0, 1.

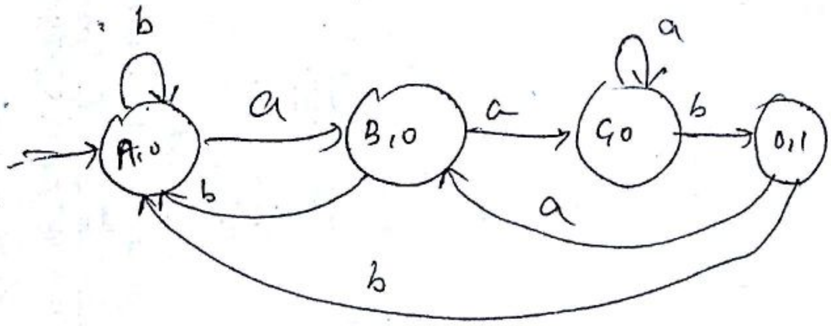
Ans: $\Sigma = \{0, 1\}$, $\Delta = \{EVEN, ODD\}$

∴ Required Melay Machine is design a



(Answer)

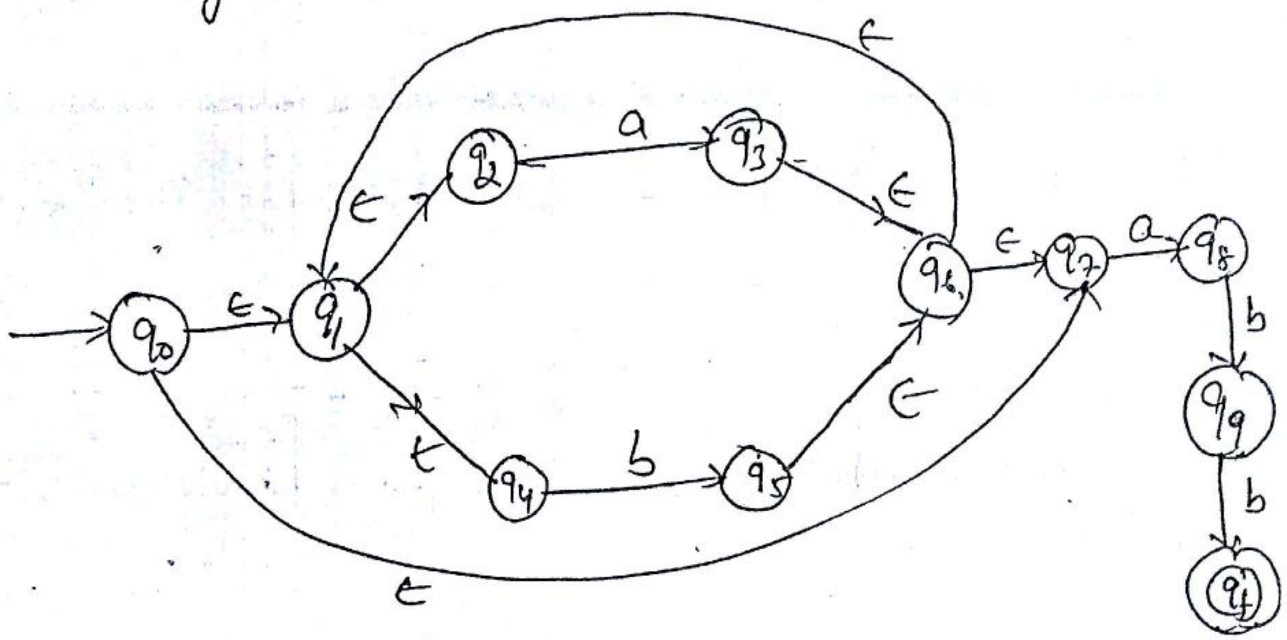
aab.



..(4) (PIET)

NFA-Λ To DFA

Example: Consider the NFA with ∈-moves given in fig. Construct an equivalent DFA.



Ans: The above NFA-Λ is of $(a+b)^*ab$

Let the equivalent DFA $M = (Q, \Sigma, \delta, A, F)$

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

$A = \in\text{-closure (Starting State of NFA)}$

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

$A = \{q_0, q_1, q_2, q_4, q_7\}$

$\delta(A, a) = \in\text{-closure}(\text{move}(A, a))$

$= \in\text{-closure}(\text{move}(\{q_0, q_1, q_2, q_4, q_7\}, a))$

$= \epsilon\text{-closure}(q_3, q_8)$

$= \{q_3, q_8, q_6, q_7, q_1, q_2, q_4\}$

Let $B = \{q_3\ q_8, q_6, q_7, q_1, q_2, q_4\}$

$\delta(A, b) = \epsilon\text{ closure}(\text{move}(A, b))$

$= \epsilon\text{-closure}(\text{move}(\{q_0, q_1, q_2, q_4, q_7\},$

$= \epsilon\text{-closure}(q_5)$

$= \{q_5, q_6, q_7, q_1, q_2, q_4\}$

Let $C = \{q_5, q_6, q_7, q_1, q_2, q_4\}$

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

$= \epsilon\text{-closure}(\text{move}(\{q_3, q_8, q_6, q_7, q_1, q_2, q_4\}.$

$= \epsilon\text{-closure}(q_3, q_8)$

$= \{q_3, q_8, q_6, q_7, q_1, q_2, q_4\} = B$

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

$= \epsilon\text{ closure}(\text{move}(\{q_3, q_8, q_6\ q_7\ q_1\ q_2\ q_4$

$= \epsilon\text{-closure}(q_5, q_9)$

$= \{q_5, q_9, q_6, q_7, q_1, q_2, q_4\}$

Let $D = \{q_5, q_9, q_6, q_7, q_1, q_2, q_4\}$

$\delta(C, a) = \epsilon\text{-closure}(\text{move}(\{q_5, q_6, q_7, q_1, q_2, q_4\}$.

$= \epsilon\text{-closure}(q_3, q_8)$

$= \{q_3, q_8, q_6, q_7, q_1, q_2, q_4\} = B$

$\delta(C, b) = \epsilon\text{-closure}(\text{move}(\{q_5, q_6, q_7, q_1, q_2, q_4\}, b)$

$= \epsilon\text{-closure}(q_5)$

$= \{q_5, q_6, q_7, q_1, q_2, q_4\} = C$

$\delta(D, a) = \epsilon\text{-closure}(\text{move}(\{q_5, q_9, q_1, q_2, q_4, q_6, q_7\}$

$= \epsilon\text{-closure}(q_3, q_8)$

$= \{q_3, q_8, q_6, q_7, q_1, q_2, q_4\} = B$

$\delta(D, b) = \epsilon\text{-closure}(\text{move}(\{q_5, q_9, q_1, q_2, q_4, q_6, q_7$

$= \epsilon\text{-closure}(q_5, q_f)$

$= \{q_5, q_f, q_1, q_2, q_4, q_6, q_7\}$

Let $E = \{q_5, q_f, q_1, q_2, q_4, q_6, q_7\}$

$\delta(E, a) = \epsilon\text{-closure}(\text{move}(\{q_5, q_f, q_1, q_2, q_4, q_6$

$= \epsilon\text{-closure}(q_3, q_8)$

$= \{q_3, q_8, q_6, q_7, q_1, q_2, q_4\} = B$

Scanned by CamScanner

$$\delta(E, a) = \epsilon\text{-closure}(\text{move}(\{q_5, q_7, q_1, q_2, q_4, q_6, q_7\}),$$

$$= \epsilon\text{-closure}(q_5)$$

$$= \{q_5, q_6, q_7, q_1, q_2, q_4\}$$

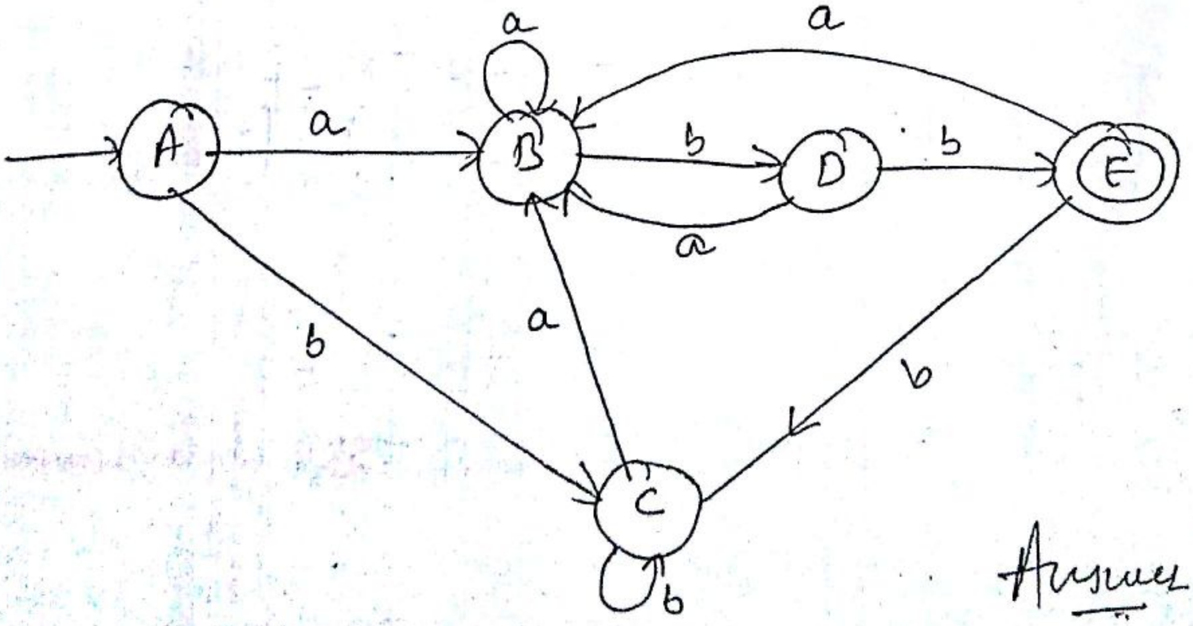
$$= C$$

Transition Table :- $Q = \{A, B, C, D, E\}$

$F = \{E\}$

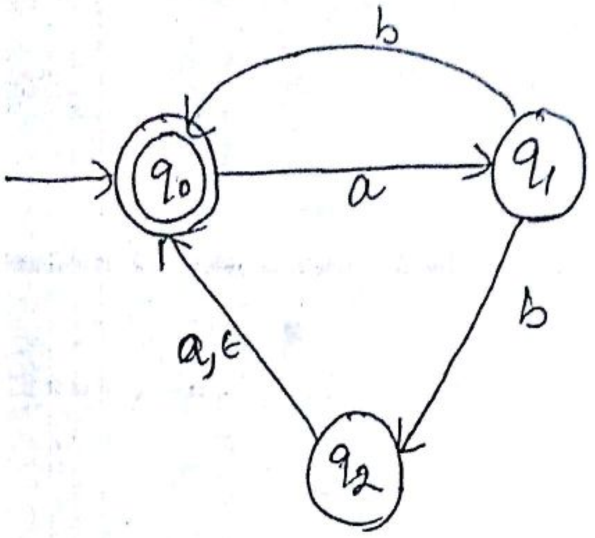
	a	b
→ A	B	C
B	B	D
C	B	C
D	B	E
*E	B	C

Transition Diagram :-



Answer

Example 2: A NFA for $L = \{ab \cup aba\}^*$ is shown in fig. Construct an equivalent DFA



Ans: Let equivalent DFA is M and

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

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

$s = \{q_0\}$

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

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

$= \{q_1\}$

let $A = \{q_1\}$

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

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

$= \phi$

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

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

Let $B = \{q_0, q_2\}$

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

Let $C = \{q_0, q_1\}$

$$\delta(B, b) = \epsilon\text{-closure}(\text{move}(\{q_0, q_2\}, b))$$
$$= \epsilon\text{-closure}(\phi)$$
$$= \phi$$

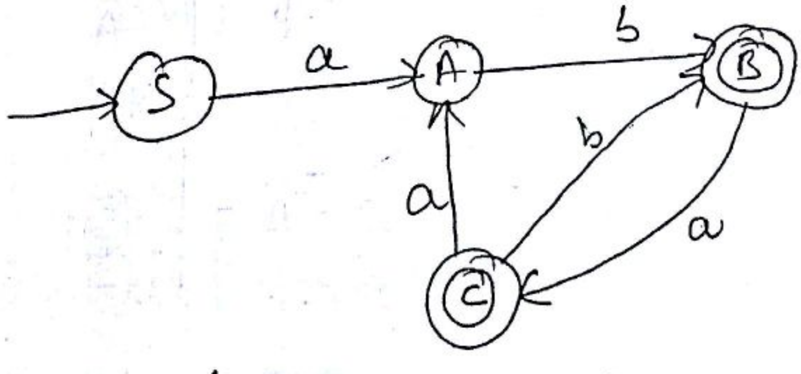
$$\delta(C, a) = \epsilon\text{-closure}(\text{move}(\{q_0, q_1\}, a))$$
$$= \epsilon\text{-closure}(q_1)$$
$$= \{q_1\}$$
$$= A$$

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

Transition Table :-

δ/Σ	a	b
$\rightarrow S$	A	ϕ
A	ϕ	B
B	C	ϕ
C	A	B

Transition Diagram :-



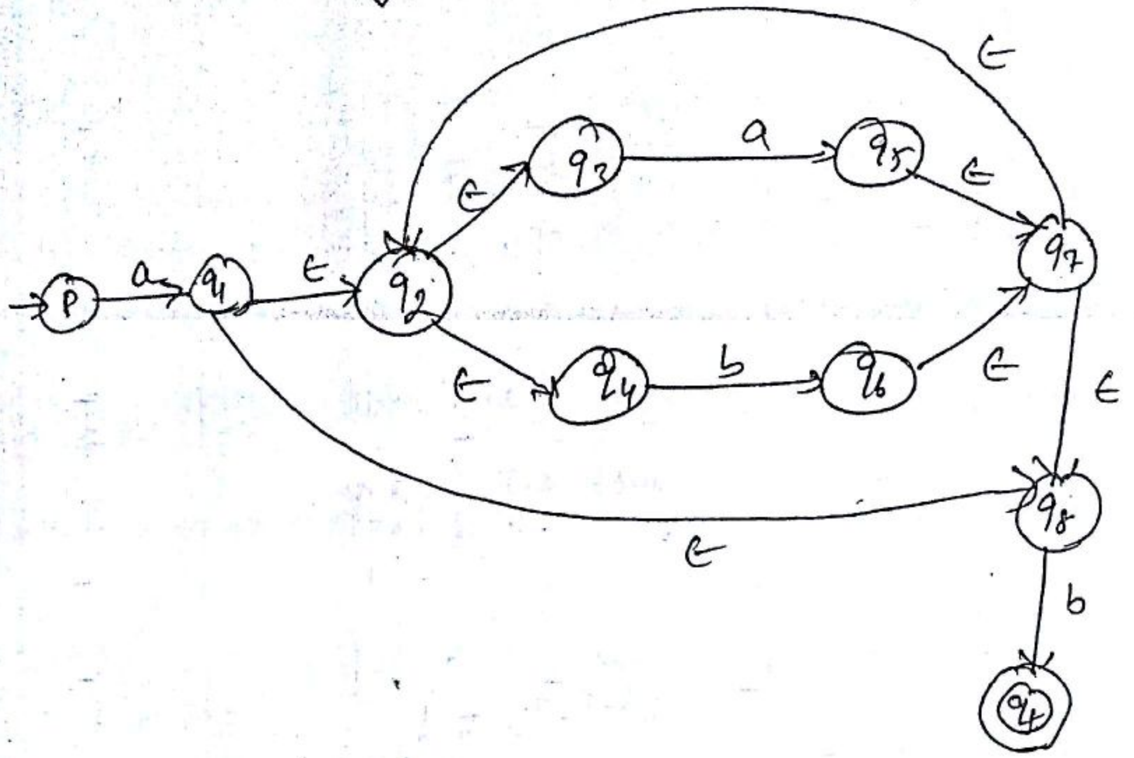
(Equivalent DFA)

Where $Q = \{S, A, B, C\}$, $\Sigma = \{a, b\}$, $F = \{B, C$

S is initial state.

Scanned by CamScanner

Example 3: Consider the NFA with ϵ-moves given in fig. Construct an equivalent DFA.



Answer:- Let $M = (Q, \Sigma, \delta, S, F)$ be the eqi DFA.

Removing Null moves from given NFA:-

$S =$ ϵ-closure (Starting state of NFA)

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

$= \{P\}$

$\delta(S, a) = \epsilon\text{-closure}(\text{move}(\{P\}, a))$

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

$= \{q_1, q_2, q_3, q_4, q_8\}$

Let $A = \{q_1, q_2, q_3, q_4, q_8\}$

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

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

$= \phi$

$\delta(A,a) = \epsilon\text{-closure}(\text{move}(\{q_1,q_2,q_3,q_4,q_8\},a))$

$= \epsilon\text{-closure}(q_5)$

$= \{q_5, q_7, q_8, q_2, q_3, q_4\}$

let $B = \{q_5, q_7, q_8, q_2, q_3, q_4\}$

$\delta(A,b) = \epsilon\text{-closure}(\text{move}(\{q_1, q_2, q_3, q_4, q_8\}$

$= \epsilon\text{-closure}(q_6, q_f)$

$= \{q_6, q_f, q_7, q_8, q_2, q_3, q_4\}$

let $C = \{q_6, q_f, q_7, q_8, q_2, q_3, q_4\}$

$\delta(B,b) = \epsilon\text{-closure}(\text{move}(\{q_5\ q_7\ q_8\ q_2\ q_3\ q_4\}, b$

$= \epsilon\text{-closure}(q_6, q_f)$

$= \{q_6, q_f, q_7, q_8, q_2, q_3, q_4\}$

$= C$

$\delta(B,a) = \epsilon\text{-closure}(\text{move}(\{q_5, q_7, q_8, q_2, q_3, q$

$= \epsilon\text{-closure}(q_5)$

$= \{q_5, q_7, q_8, q_2, q_3, q_4\}$

$= B$

$\delta(C, a) = \epsilon\text{-closure}(\text{move}(\{q_6, q_f, q_7, q_8, q_2, q_3, q_4\}, a)$

$= \epsilon\text{-closure}(q_5)$

$= \{q_5, q_7, q_8, q_2, q_3, q_4\}$

$= B$

$\delta(C, b) = \epsilon\text{-closure}(\text{move}(\{q_6\, q_f, q_7, q_8, q_2, q_3, q_4\}$

$= \epsilon\text{-closure}(q_6, q_f)$

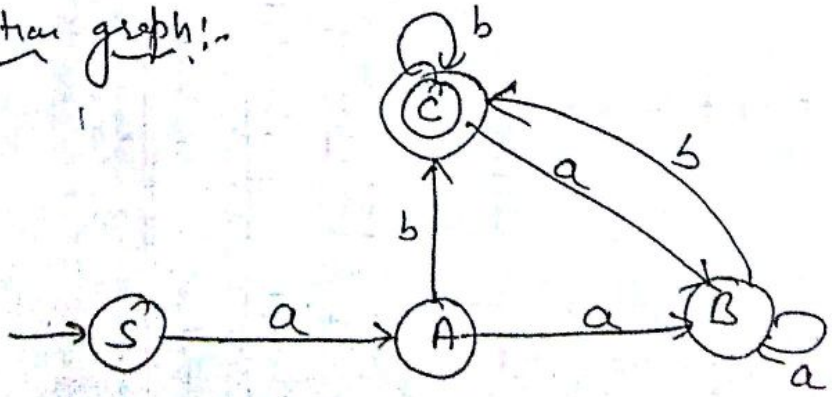
$= \{q_6, q_f, q_7, q_8, q_2, q_3, q_4\}$

$= C$

Transition Table for DFA :-

δ/Σ	a	b
$\rightarrow$ S	A	–
A	B	C
B	B	C
C	B	C

Transition graph:-



Example: accept a Binary string divisible by 4.

Ans: Divisible by 4 ∴ 4 states formed as q_0 q_1 q_2 q_3

Short cut method:

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

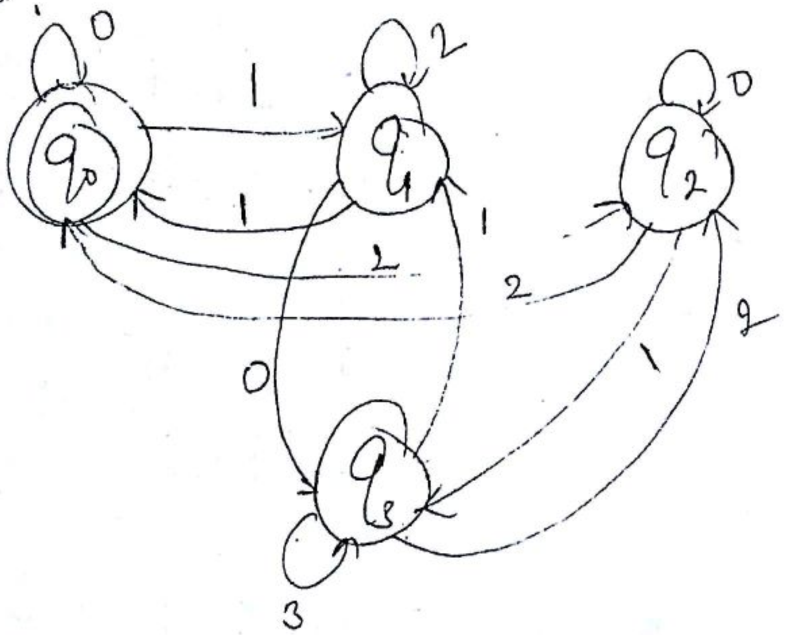
write all the states in 0, 1 Transition & repeat all the state. till the end.

Example Accept a ternary string {0,1,2}, divisible by 4.

Ans. Short cut Method:-

	0	1	2
q_0	q_0	q_1	q_2
q_1	q_3	q_0	q_1
q_2	q_2	q_3	q_0
q_3	q_1	q_2	q_3

DFA is:



we reach q_3 with string 10, on see 0

$100 \Rightarrow 1\times3+0 = 3\ 0 \Rightarrow 3\times3+0 = 9$

Divide by 9 with 4 get the remainder as 1 so move to q_1 state when apply 0.