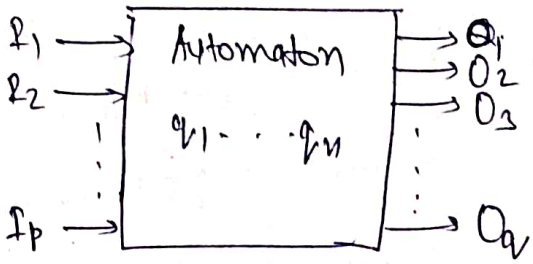
UNIT-1
Automata

Definition of Automata: An Automata is define as a system where energy, material, or information are transform and used to performing some function without direct participation of man.

Eg: Automatic machine tools, Automatic Packing machines & Automatic photo printing machine etc.



(i) **I/P :-** At each of the discrete instants of time $t_1, t_2 \ldots t_m$ the I/p values $I_1, I_2, \ldots I_p$, each of which can take a finite no. of fixed values from the I/p alphabate Σ.

(ii) **O/P :-** $O_1, O_2 \ldots O_q$ are the o/p of the model, each of which can take a finite no. of fixed values from an o/p O

(iii) **State :-** At any state of time the automaton can be in one of the state $q_1, q_2, \ldots q_n$

(iv) **State Relation:-** The next state of the automaton at any instant of time is determined by the present state and the present I/p.

(v) **O/p relation:-** The o/p is related to either state only or to both I/p and the state

Theory of Computation (TOC) :

In Computation, any task that is perform by any Calculator or Computer.

Symbole : a, b, c, 0, 1, 2, ... ♀ ♂

$|\Sigma|$ Alphabate :: {a, b} {0, 1} {0, 1, .. 9}
{a, b, c}

String :: For $\Sigma = \{a, b\}$

1-bit = {a, b} $= 2^1 = 2$

2-bit = {aa, ab, ba, bb} $= 2^2 = 4$

3-bit = {aaa, aab, aba, abb, baa, bab, bba, bb

$= 2^3 = 8$

no. of string $|\Sigma|^n$, where n Size of String

Language :. An English language is collection of Words, but the mic language is collection of string.

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

L_1 = Set of all string of length 2.
= {aa, ab, ba, bb}

L_2 = Set of all string of length 3
{aaa, aab, aba, abb, baa, bab, bba, bbb}

L_3 = Set of all string, where each string start with a
{a, aa, ab, aba, aabb, abba

Powers of Σ (Sigma) :

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

Σ^0 = set of all string over Σ of length zero, $\{\epsilon\}$

Σ^1 = set of all string over Σ of length 1, $\{a, b\}$

Σ^2 = for length $\{aa, ab, ba, bb\}$ = $\Sigma \cdot \Sigma = \{a, b\} \cdot \{a, b\}$

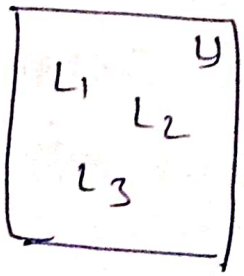
Σ^3 = for length 3

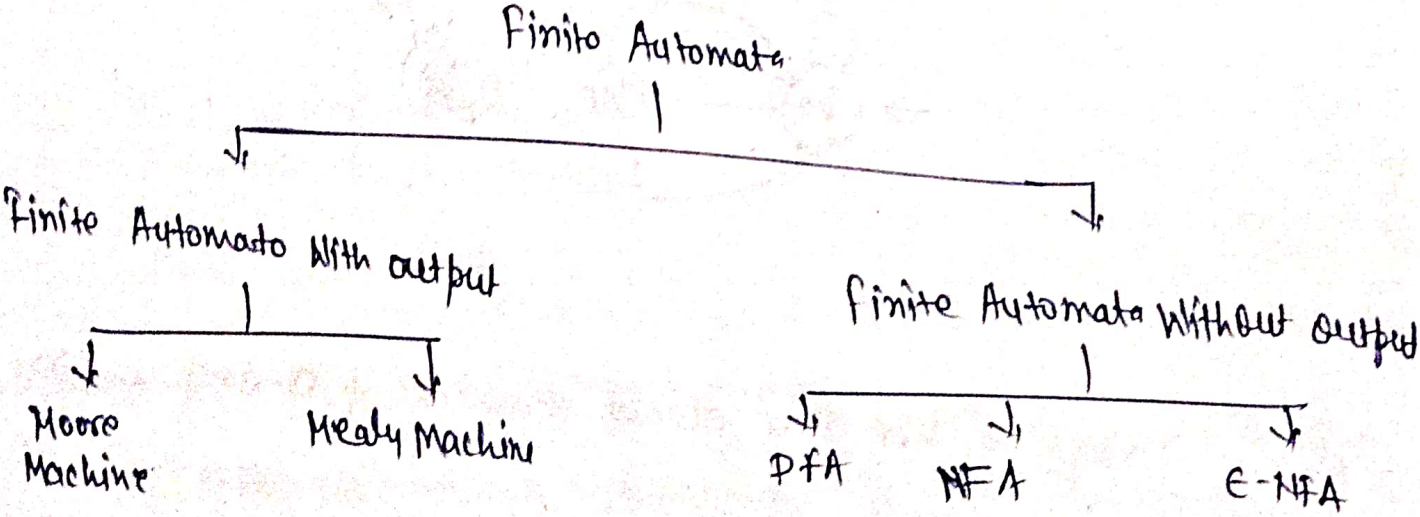
$\{aaa, aab, aba, abb, baa, bab, bba, bbb\}$

$\Sigma \times \Sigma \times \Sigma$ =

Σ^* = $\Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots\ldots$

= Universal set





Deterministic Finite Automata:

5-tuples $(Q, \Sigma, \delta, q_0, q_f)$

$Q \rightarrow$ is a finite non-empty set of state

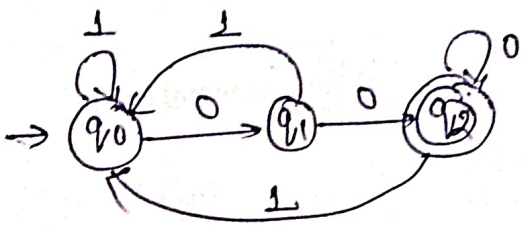
$\Sigma \rightarrow$ is a finite non-empty set of I/P alphabate

$\delta \rightarrow$ is a transition function which map $Q \times \Sigma$ into Q and is usually called discrete transition function.
- this is the function which describes the change of state during the transition

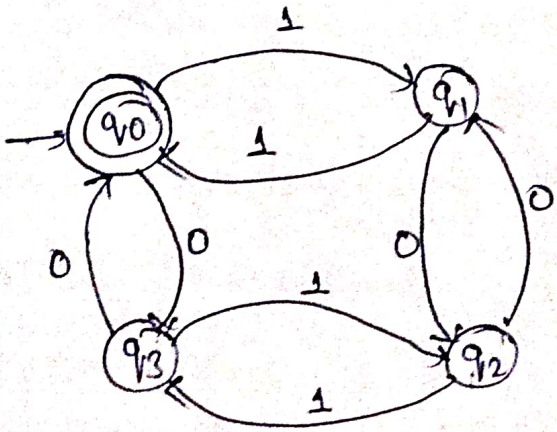
$q_0 \in Q$ is the initial state

$q_f \subseteq Q$ is the final state

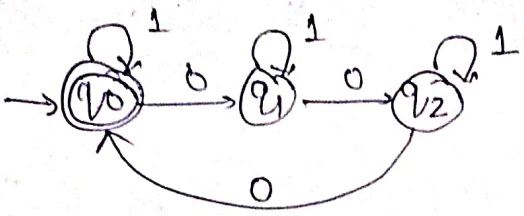
Q: Design a DFA that accepts set of string such that every string ends with 00, over alphabate $\Sigma = \{0,1\}$



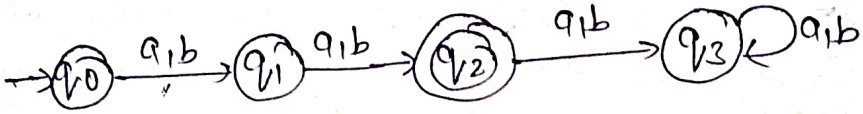
Q. Design a DFA that accept even no. of 0 and 1 over $\Sigma = \{0,1\}$.



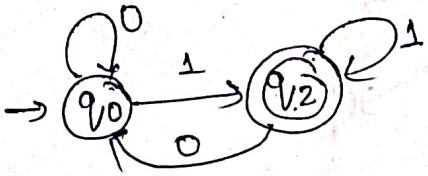
Construct A DFA that accept set of String where the no. of 0's in every string is multiple of 3 over alphabet $\Sigma = \{0,1\}$



Q. Construct a DFA which accepts set of all string ove $\Sigma = \{a,b\}$ of length = 2.



Q. Construct a DFA that accept set of strings such that every string ends with 1, over Alphabate $\Sigma = \{0,1\}$



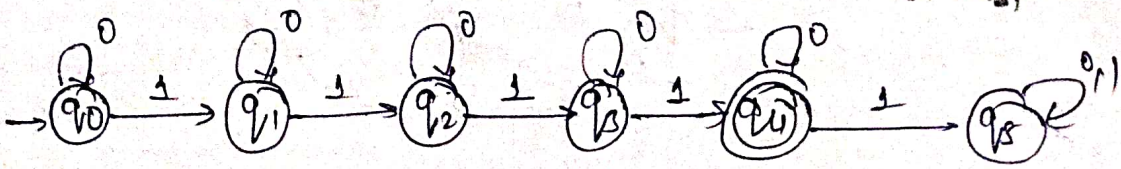
Q Construct a DFA which accept set of string Containing (i) exactly four 1's in every string over alphabate $\Sigma = \{0,1\}$

At least

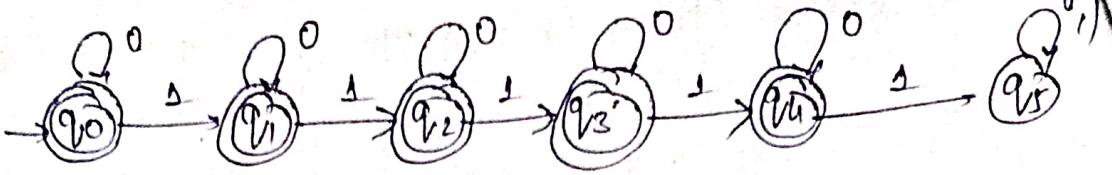
At Most

(ii) At Most four 1's

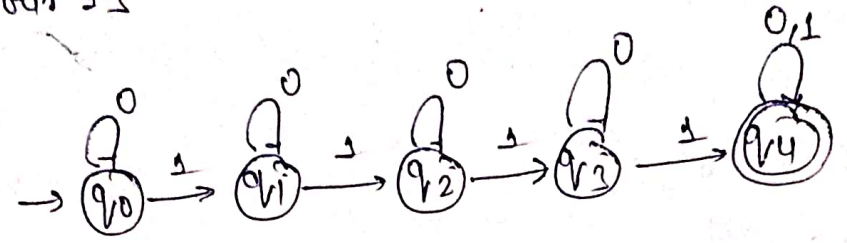
(iii) At least four 1's



(ii) At Most four 1's

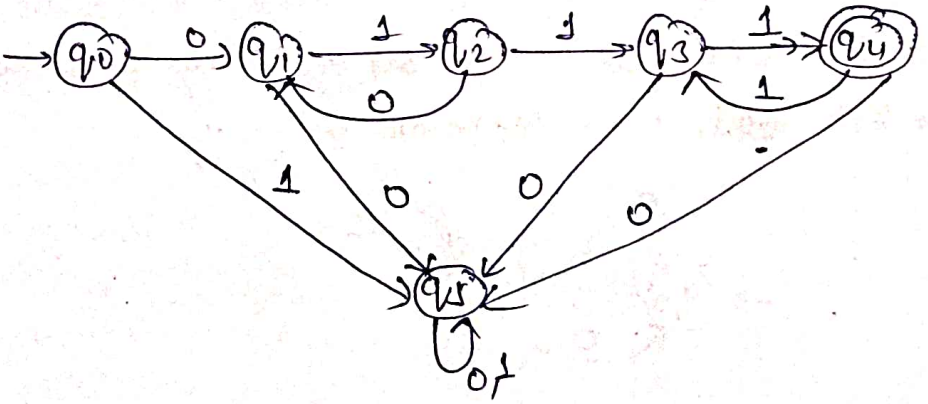


(iii) At least four 1's

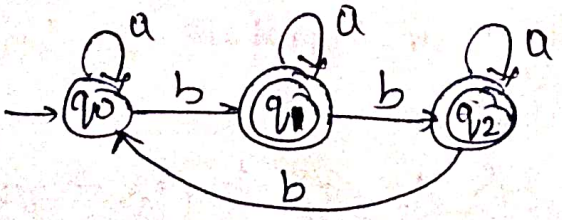


Q. Design a FA for the language.

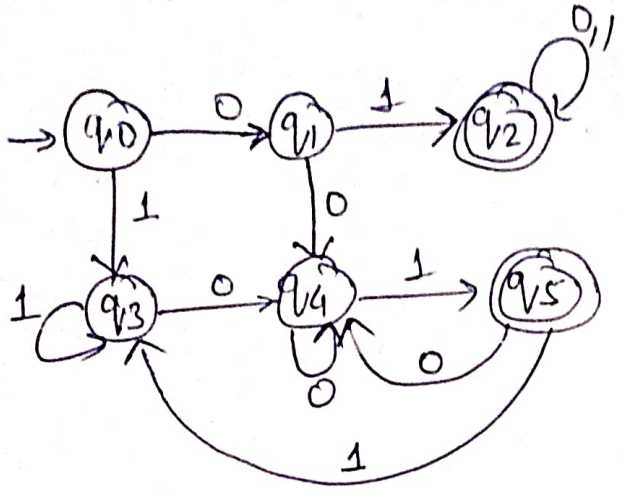
$L = \{ (01)^i 1^{2j} \mid i \geq 1, j \geq 1 \}$ Where $w \in (0,1)^*$



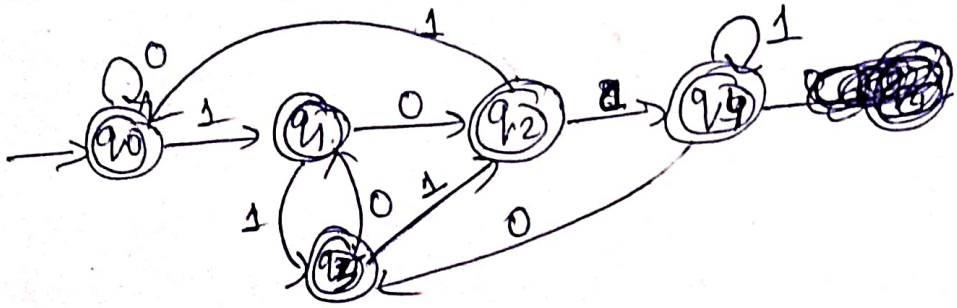
Q. Design a FA for the language $L = \{ w \in (a,b)^* \mid n_b(w) \bmod 3 > 0 \}$



Draw a FA that accept a substring start with 01 or end with 01 for I/P Alphabate {0,1}



Q. Draw the DFA for $L = \{ |W|\%5 \mid W \in (0,1)^* \}$

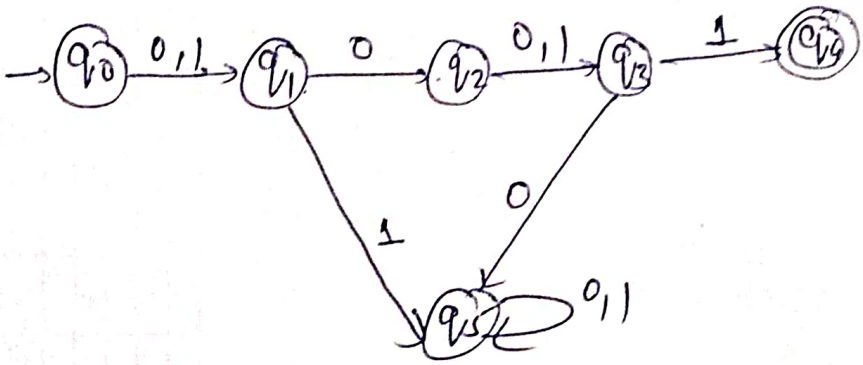


0000 → 0
0001 → 1
0010 - 2
0011 - 3
0100 - 4
0101 - 5
0110 - 6
0111 - 7
1000 - 8
1001 - 9
1010 - 10
1011 - 11
1100 - 12
1101 - 13
1110 - 14
1111 - 15

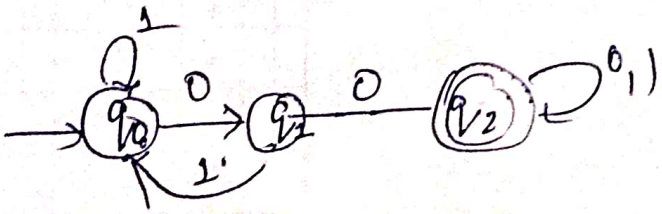
Q Design a FA which accepts the language.

$L = \{ w \in (0,1)^* \mid$ second symbol of w is 0 & fourth input is 1 $\}$

Sol.



Q. For the given FA Write the language and also give the transition table.



$L = \{ w \in (0,1)^+ \mid$ every string w of the language containing 00 as substring $\}$

Minimization of DFA :-

Minimization of DFA is required to obtain the minimal version of any DFA which consists of the minimum no. of states possible.

Two states 'A' and 'B' are said to be equivalent if

$\delta(A, x) \vdash f$ OR $\delta(A, x) \not\vdash f$

$\delta(B, x) \vdash f$ $\delta(B, x) \not\vdash f$

Where x is any input string.

if $|x| = 0$, then A & B are said to be 0-equivalent

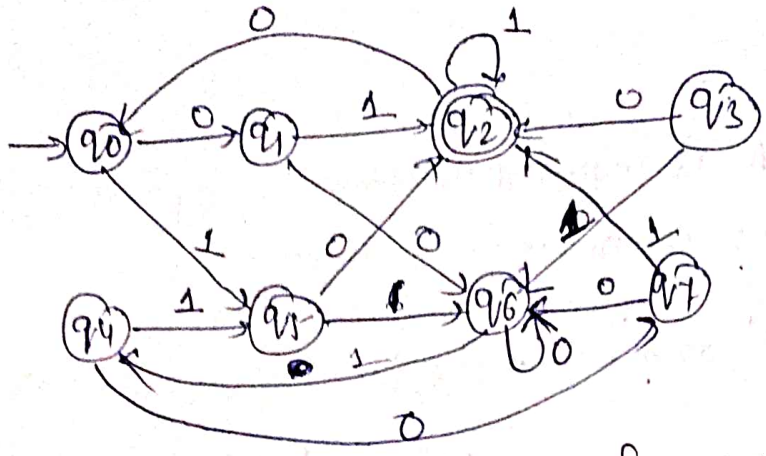
if $|x| = 1$, then A & B " " " 1-equivalent

if $|x| = 2$, " " " " 2-equivalent

⋮

if $|x| = n$, then A & B are said to be n-equivalent

Ex:



First we remove q_3 which is not reachable state.

Transition Table:-

State	I/P 0	1
$\rightarrow q_0$	q_1	q_5
q_1	q_6	q_2
(q_2)	q_0	q_2
q_3	q_2	q_6 X
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

0-equivalence:

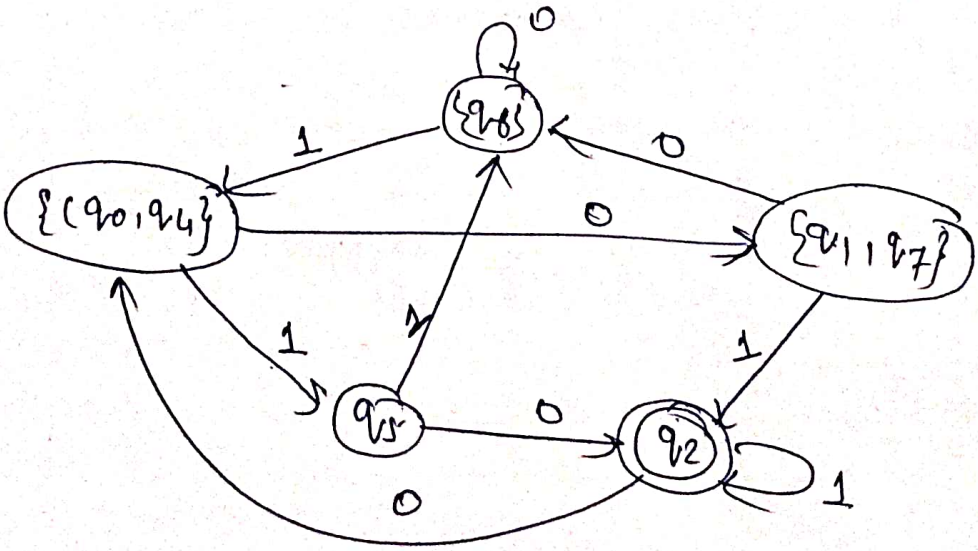
$Q_0 = \{q_0, q_1, q_4, q_5, q_6, q_7\}\ \{q_2\}$

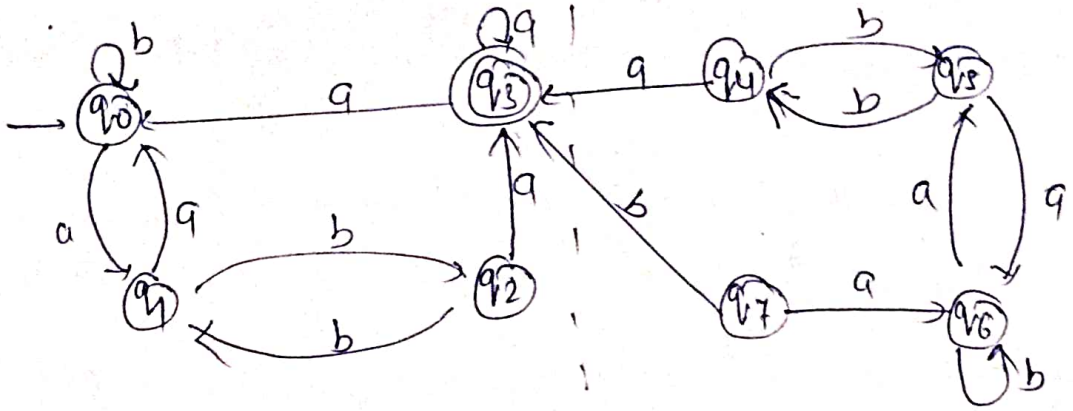
1-equivalence:

$\theta_1 = \{q_0, q_4, q_5\}\ \{q_1, q_7\}\ \{q_6\}\ \{q_2\}$

2-equivalence:

$\theta_2 = \{q_0, q_4\}\ \{q_5\}\ \{q_1, q_7\}\ \{q_6\}\ \{q_2\}$





0- equivalence:

$Q_0 = \{q_0, q_1, q_2\}\ \{q_3\}$

1- equivalence

$\theta_1 = \{q_0, q_1\}\ \{q_2\}\ \{q_3\}$

2- equivalence

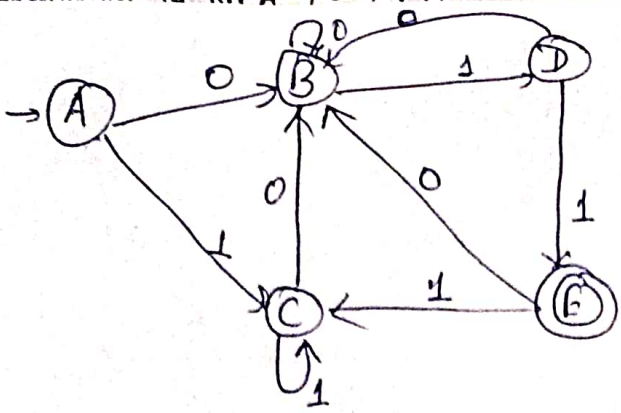
$\theta_2 = \{q_0\}\ \{q_1\}\ \{q_2\}\ \{q_3\}$

Transition table:

State	i/p a	b
→ q_0	q_1	q_0
q_1	q_0	q_2
q_2	q_3	q_1
(*) q_3	q_3	q_0
Not reachable state: q_4	q_5	q_5
q_5	q_6	q_4
q_6	q_5	q_6
q_7	q_6	q_3

Conversion of NFA to DFA:

5.



Transition Table:

State	I/P 0	1
→A	B	C
B	B	D
C	B	C
D	B	E
(E)	B	C

0-equivalence

$\pi_0 \rightarrow$ { A, B, C, D } { E }

1-equivalence

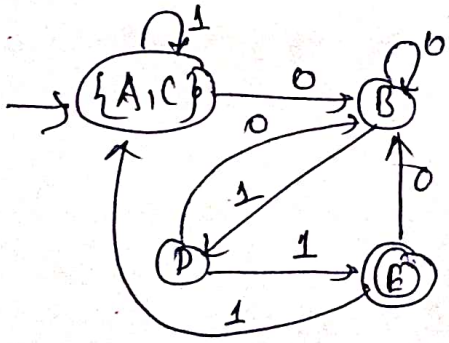
$\pi_1 \rightarrow$ { A, B, C } { D } { E }

2-equivalence

$\pi_2 \rightarrow$ { A, C } { B } { D } { E }

3-equivalence

$\pi_3 \rightarrow$ { A, C } { B } { D } { E }



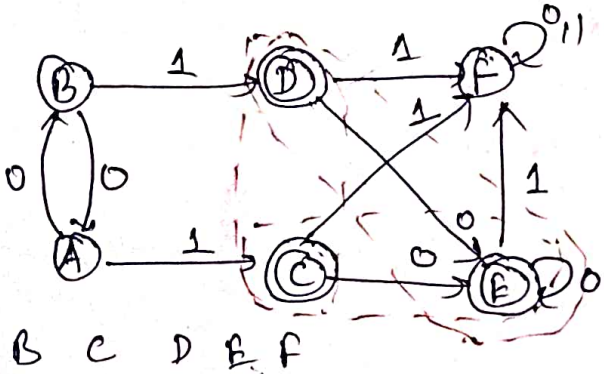
Minimization of DFA - Table Filling Method (Myhill-Nerode Theorem)

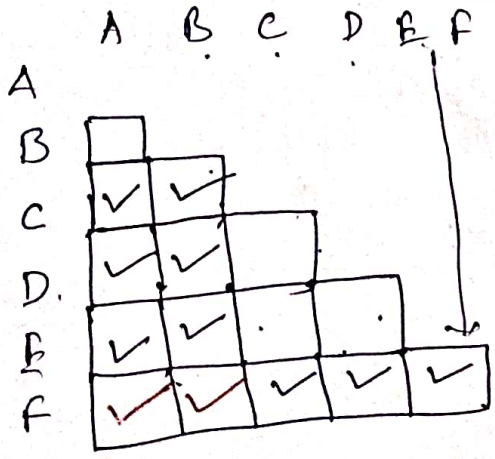
Steps:

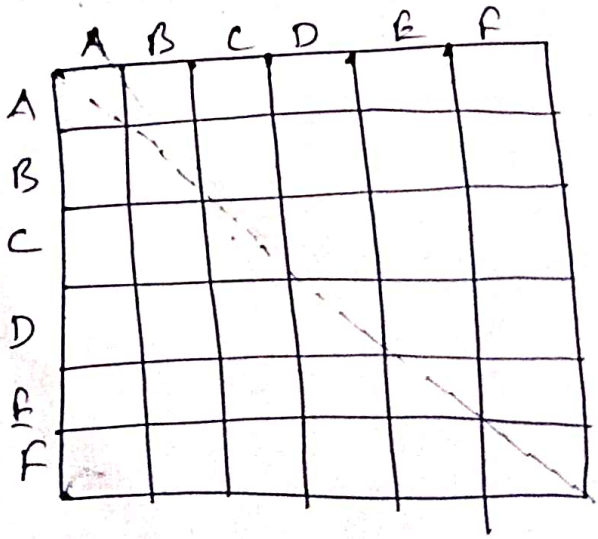
1. Draw a table for all pairs of states (P, Q)
2. Marks all pairs where $P \in F$ and $Q \notin F$
3. if there are any unmarked pairs (P, Q) such that $[\delta(P, x), \delta(Q, x)]$ is Marked, then mark $[P, Q]$

   Where x is an I/P symbol.

   Repeat this until No more markings can be made
4. Combine all the unmarked pairs and make them a single state in the minimized DFA







$(B, A) = \{\delta(B,0) \vdash A, \delta(A,0) \vdash B\}$ unmarked ; $\{\delta(B,1) \vdash D, \delta(A,1) \vdash C\}$ unmarked

$(D, C) = \{\delta(D,0) \vdash E, \delta(C,0) \vdash E\}$ unmarked ; $\{\delta(D,1) \vdash F, \delta(C,1) \vdash F\}$ unmarked

$(E, C) = \{\delta(E,0) \vdash E, \delta(C,0) \vdash E\}$ unmarked ; $\{\delta(E,1) \vdash F, \delta(C,1) \vdash F\}$ unmarked

(E,D) = $\delta(E,0) \vdash B$, $\delta(D,0) \vdash E$ } unmarked ; $\delta(E,1) \vdash F$, $\delta(D,1) \vdash F$ } unmarked

(F,A) = $\delta(F,0) \vdash F$, $\delta(A,0) \vdash B$ } unmarked ; $\delta(F,1) \vdash F$, $\delta(A,1) \vdash C$ } marked

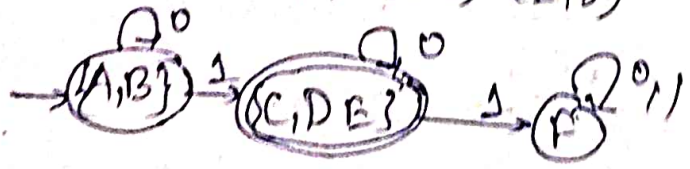
(F,C) is marked so (F,A) become marked

(F,B) = $\delta(F,0) \vdash F$, $\delta(B,0) \vdash A$ } marked

(F,A) is marked so (F,B) become marked.

Make a set of unmarked pair

(A,B) (D,C) (E,C) (E,D)



⇒ Transformation of NFA to DFA:

The DFA equivalent of NFA is to simulate the moves in parallel state of a DFA will be combination of one or more states of NFA, hence every state of a DFA will be represented by some subset of set of states of NFA. and therefore the transition of NFA to DFA is normally called a subset Constructions.

⇒ Equivalence of DFA and NFA:

We are going to prove following " if NDFA accepts a language 'L', then there exist a DFA that also accept 'L'.

Proof:- Let M be any given NFA, which accept L.

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

Now Let us Construct a DFA $M' = \{ Q', \Sigma, \delta', q_0', q_f' \}$ where.

(i) $Q' = 2^Q$, i.e Q' contains the subset of Q (any state in Q' is denoted by $\{q_1, q_2, \cdots q_n\}$ where $q_1, q_2, \cdots q_n \in Q$ }

(ii) $q_0' = \{q_0\}$

(iii) q_f' is the set of all subsets of Q containing an element of q_f

(iv) Transition function δ' is defined as follows

$$\delta'([q_1, q_2, \cdots q_n], a) = \delta(q_1, a) \cup \delta(q_2, a) \cup \delta(q_3, a) \cdots \delta(q_n, a)$$
$$= P_1 \cup P_2 \cup \cdots \cup P_k$$

if and only if $\delta'(q_1, q_2, \cdots q_n), a) = \{P_1, P_2, \cdots P_k\}$

⇒ Procedure for converting NFA to Equivalent DFA:

Let M be an NFA denoted by $\{Q, \Sigma, \delta, q_0, q_f\}$ which accepts L. to obtain a equivalent DFA $M' = (Q', \Sigma, \delta', q_0, q_f)$ which accept the same language as given NFA $M = (Q, \Sigma, \delta, q_0, q_f)$ above, we may proceed as follows:

Step 1 - Initially $Q' = \phi$

Step 2 - Put $[q_0]$ into Q'. $[q_0]$ is the initial state of DFA M'.

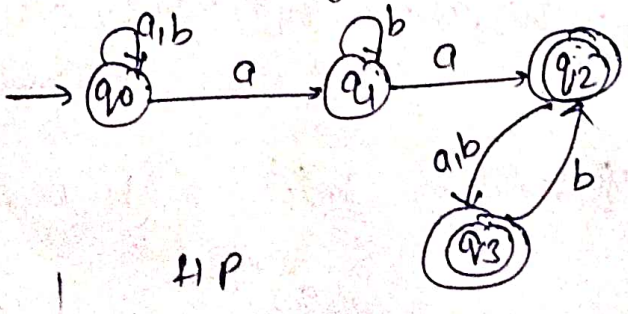
Step 3. Add every new state q to Q'

Where $\delta'(q, a) = \bigcup_{p \in q} \delta(p, a)$, δ on the right hand side is that of NFA 'M'.

Step 4. Repeat step 3 till new states are there to add in Q', if there is no further new state found to add in Q', The process terminated. All states in Q' that contain final state of 'M' are accepting state of M'

Note: The states which are not reached from the initial state should not be included in Q'. Thus the set of states (Q') is not necessarily equal to 2^Q.

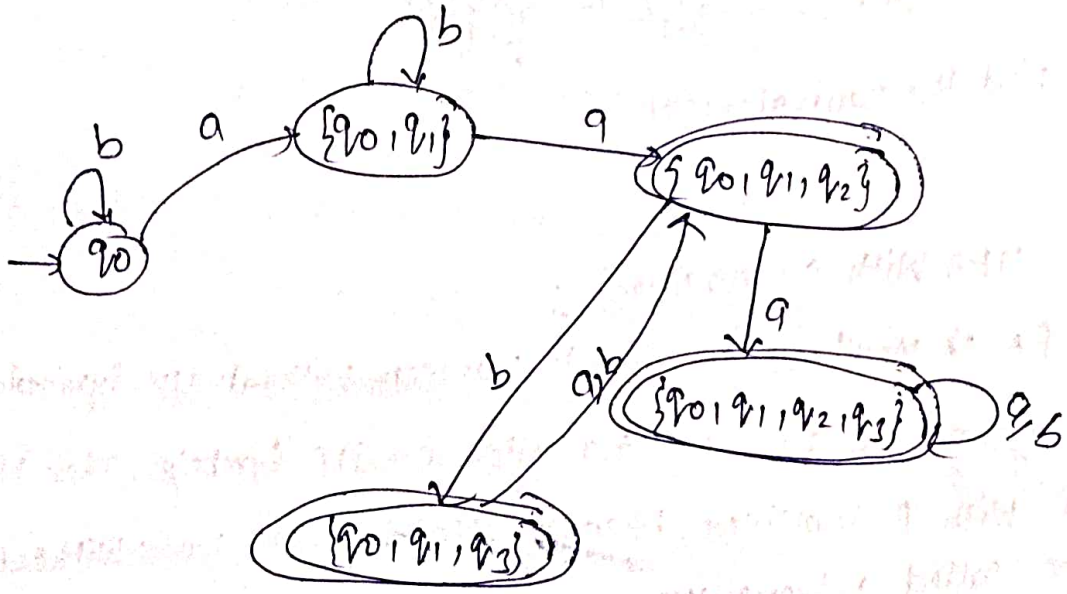
Q. Convert the following NFA into DFA



Soln:

State	I/P a	b
→ q_0	$\{q_0, q_1\}$	q_0
q_1	q_2	q_1
q_2	q_3	q_3
q_3	ϵ	q_2

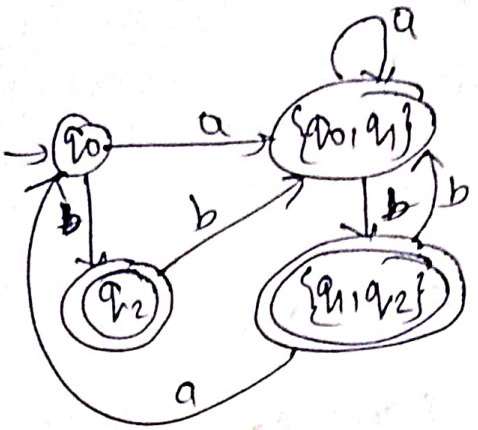
Q	a	b
$\to q_0$	$\{q_0, q_1\}$	q_0
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$
$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_3\}$
$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3\}$
$\{q_0, q_1, q_3\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$



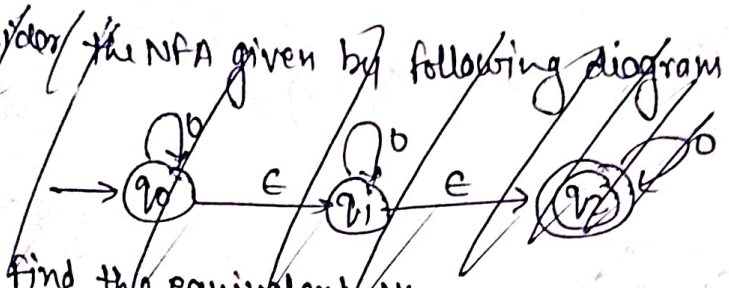
Q. Convert the following NFA into DFA

Q	a	b
$\to q_0$	$\{q_0, q_1\}$	$\{q_2\}$
q_1	$\{q_0\}$	$\{q_1\}$
(q_2)	ϕ	$\{q_0, q_1\}$

Q	a	b
$\to q_0$	$\{q_0, q_1\}$	$\{q_2\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_1, q_2\}$
q_2	ϕ	$\{q_0, q_1\}$
$\{q_1, q_2\}$	$\{q_0\}$	$\{q_0, q_1\}$



Q.1 Consider the NFA given by following diagram



Find the equivalent N.

NFA-ϵ (NFA With ϵ-transition) :

If a FA is Modified to Permit transition without I/P symbole, along With zero, one or more transition on I/P symbole, then we get a NFA With ϵ-transitions, because the transition mode without symboles are called ϵ-transition.

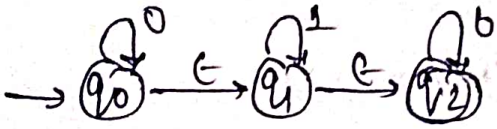


fig.1

fig.1 represent the NFA With ϵ-transition because it is possible to make transition from state q_0 to q_2 and q_1 to q_2 without Consuming any of the I/P symboles.

5-tuple $(Q, \Sigma, \delta, q_0, q_f)$

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

Note: For representing ϵ transition need to know ϵ-closure

ϵ-closure(q) = set of all those states which Can be reached from q on the path labeled by ϵ

$$\hat{\delta}(\epsilon\text{-closure}(q, aw)) \vdash \epsilon\text{-closure}(\delta(\hat{\delta}((\epsilon\text{-closure } q), a), w))$$

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

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

Q. Convert the given NFA With ϵ to NFA Without ϵ

ϵ-closure$(q_0) = \{q_0, q_1, q_2\}$

ϵ-closure$(q_1) = \{q_1, q_2\}$

ϵ-closure$(q_2) = \{q_2\}$

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

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

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

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

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

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

$= \{q_0, q_1 q_2\}$

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

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

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

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

$= \epsilon\text{-Closure}(q_1, q_2\}$

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

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

$= \{q_1, q_2\}$

NFA (Non-Deterministic finite ...

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

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

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

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

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

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

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

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

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

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

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

$= \{q_1,q_2\}$

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

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

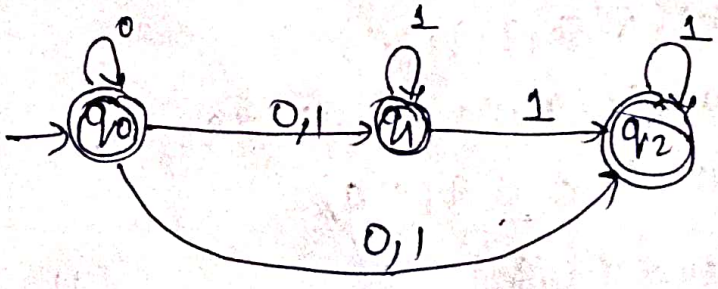
$= \epsilon\text{-closure}(\phi) = \phi$

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

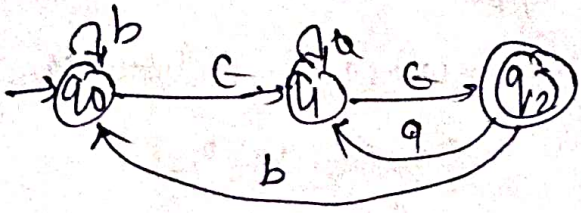
$= \epsilon\text{-closure}(\delta(q_2,1))$

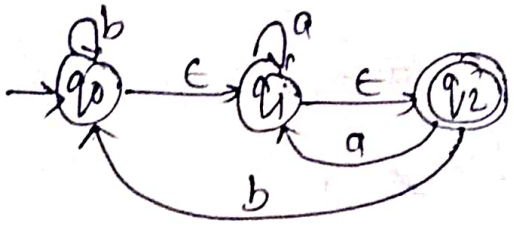
$= \epsilon\text{-closure}(q_2)$

$= q_2$



Q.





Convert the given NFA with ε to NFA without ε.

ε-closure(q_0) → $\{q_0, q_1, q_2\}$

ε-closure(q_1) → $\{q_1, q_2\}$

ε-closure(q_2) → $\{q_2\}$

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

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

= ε-closure($\delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a)$)

= ε-closure($\epsilon \cup q_1 \cup q_1$)

= ε-closure(q_1)

= $\{q_1, q_2\}$

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

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

= ε-closure($\delta(q_0, b) \cup \delta(q_1, b) \cup \delta(q_1, b)$)

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

= ε-closure(q_0)

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

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

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

= ε-closure($\delta(q_1, a) \cup \delta(q_2, a)$)

= ε-closure($\phi \cup q_1$)

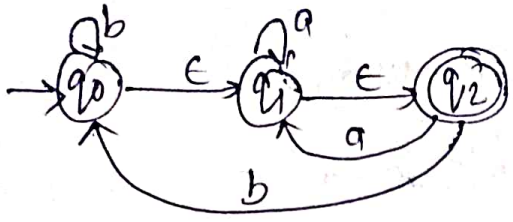
= $\{q_1, q_2\}$

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

= ε-closure($\delta(q_1, q_2), b$)

= ε-closure($\delta(q_1, b) \cup \delta(q_2, b)$)

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



Convert the given NFA with ε to NFA without ε.

ε-closure(q_0) → $\{q_0, q_1, q_2\}$

ε-closure(q_1) → $\{q_1, q_2\}$

ε-closure(q_2) → $\{q_2\}$

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

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

= ε-closure($\delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a)$)

= ε-closure($\epsilon \cup q_1 \cup q_1$)

= ε-closure($\{q_1\}$)

= $\{q_1, q_2\}$

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

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

= ε-closure($\delta(q_0, b) \cup \delta(q_1, b) \cup \delta(q_1, b)$)

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

= ε-closure(q_0)

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

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

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

= ε-closure($\delta(q_1, a) \cup \delta(q_2, a)$)

= ε-closure($\phi \cup q_1$)

= $\{q_1, q_2\}$

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

= ε-closure($\delta(q_1, q_2), b$)

= ε-closure($\delta(q_1, b) \cup \delta(q_2, b)$)

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

~~$\hat{\delta}(q_2, a) =$ ϵ-closure $\{\delta(\delta($ ϵ-closure,~~

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

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

$= \epsilon$-closure (q_1)

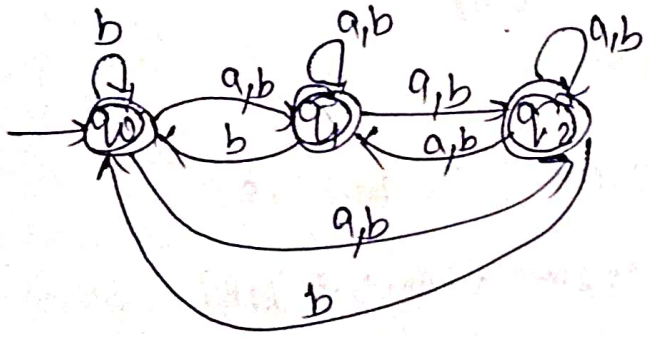
$= \{q_1, q_2\}$

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

$= \epsilon$-closure $(\delta(q_2, b))$

$= \epsilon$-closure (q_0)

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



NFA · With out ϵ

NFA (Non-Deterministic finite Automata)

5-tuple ($Q, \Sigma, \delta, q_0, q_f$)

Q → Non Emply set of states

Σ → " " " " I/P Alphabate

δ → Transition function shown the mapping b/w ~~Q~~ $Q \times (\Sigma \cup \{\in\}) \to 2^Q$

$q_0 \in Q$ is the initial state

$q_f \subseteq Q$ is the final state

Note!. NFA make diffrence with DFA, In NFA it's Contain all Possible Combination some may Contain true or may not be.

* it's used for backtracking

Q. Draw the NFA for $\Sigma = \{a, b\}$

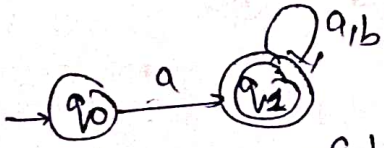
(i) L_1 = { Start With a }

(ii) L_2 = { Containing a }

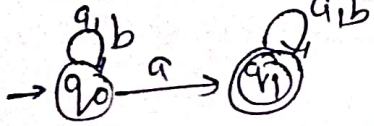
(iii) L_3 = { end wit a }

(iv) L_4 = { Stat With ab }

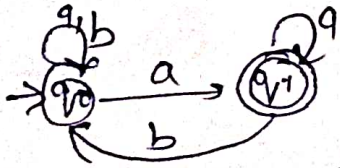
(i)



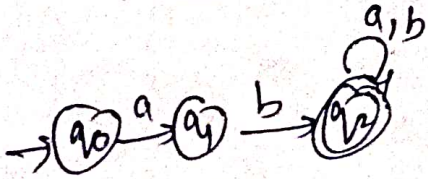
(ii)



(iii)



(iv)

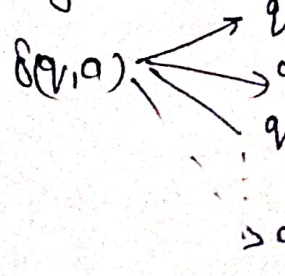


Note!. In Non-deterministic accepter, the range of δ is the power set of 2^Q, so it's value is not a single element of Q, but a subset of it.

The subset defines the set of possible states that can be reached by transition.

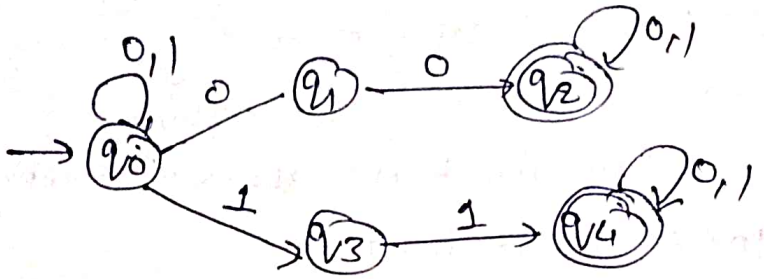
Let q be an state of Q and $a \in \Sigma$ then.

We can ~~transi~~ transit from a state q on same input a to diffrent states $q_0, q_1 \cdots q_n$ in Q this not possible in the case of DFA

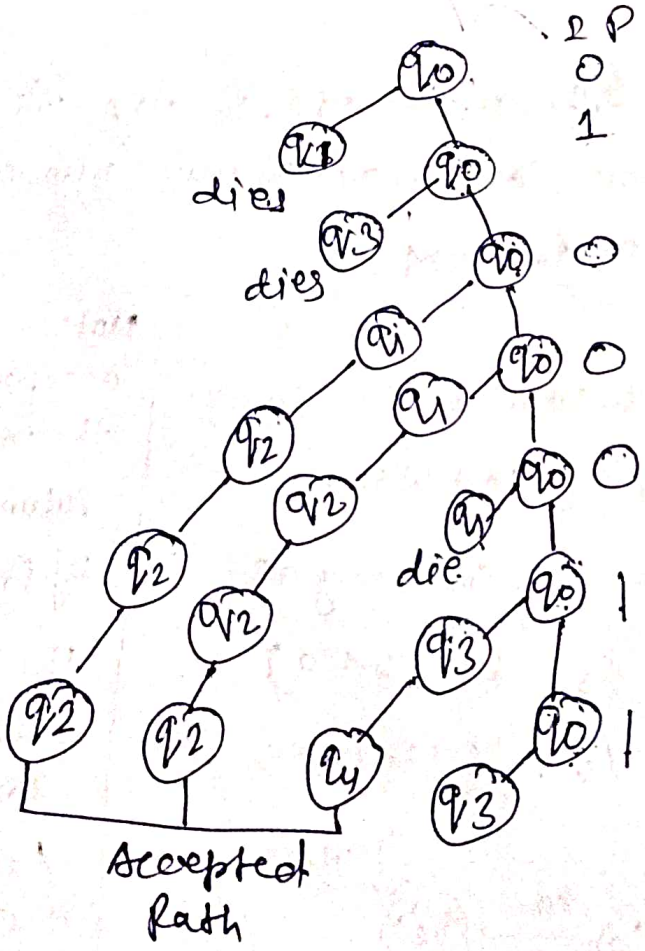


Q. Design a NFA for a language L = all string over {0,1} that has at least two consecutive 0's or 1's

Sol:

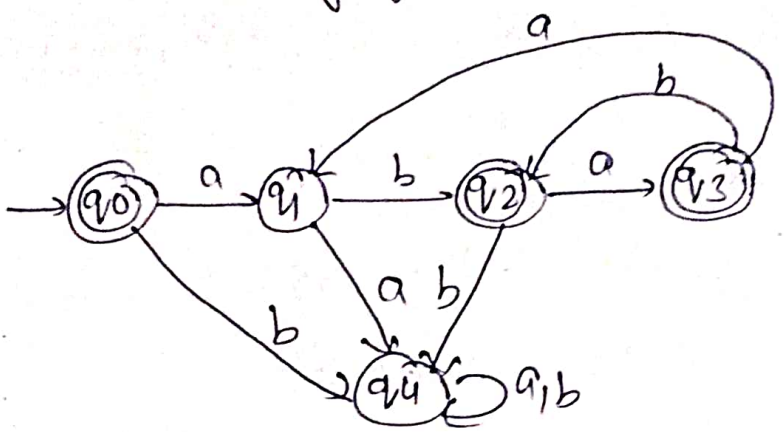


The tree of states this NFA is in for the I/P 0100011

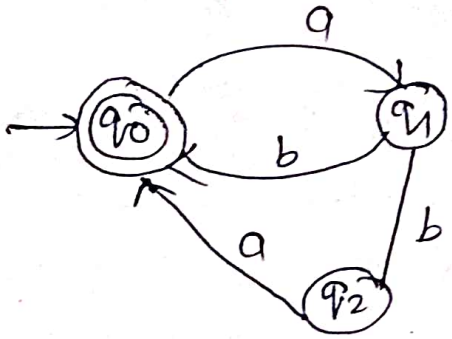


Design a NFA for the language $L = (ab \cup aba)^*$.

Ans



DFA



NFA

Q. Draw the state diagram for NFA accepting string.

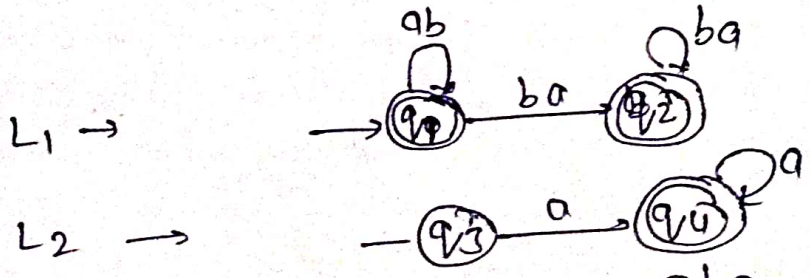
$$L = (ab)^*(ba)^* \cup aa^*$$

Ans: We construct NFA for the language L in two paths i.e

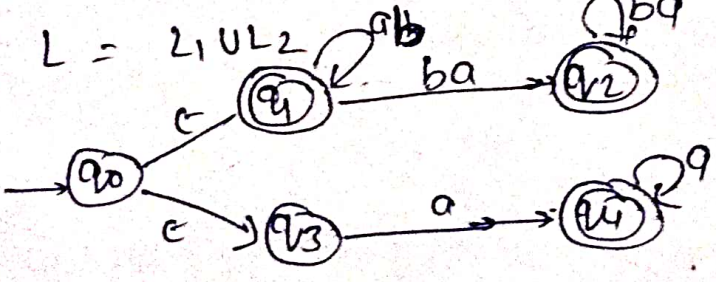
$$L = L_1 \cup L_2$$

$$L_1 = (ab)^*(ba)^*$$

$$L_2 = aa^*$$



$L = L_1 \cup L_2$



Q. Find NFA with four state for the language.

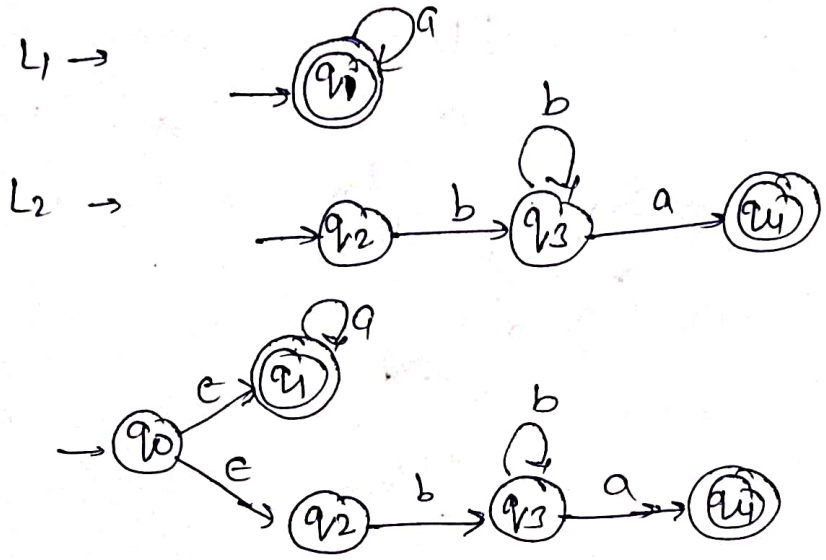
$$L = \{(a^n : n \geq 0) \cup (b^n a : n \geq 1)\}$$

Ans:

$$L = L_1 \cup L_2$$

$$L_1 = a^n : n \geq 0$$

$$L_2 = b^n a : n \geq 1$$

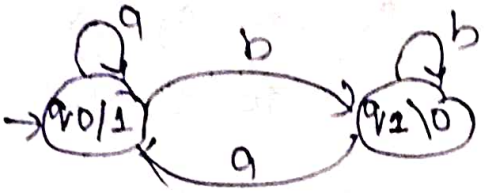


FA With output

Moore machine

$\lambda : Q \to \Delta$

$\lambda(t) = \lambda q(t)$



6-tuple $(Q, \Sigma, \delta, \Delta, \lambda, q_0)$

$Q \to$ Finite set of state

$\Sigma \to$ Non Empty set of I/P alphabate

$\delta \to$ Transition function $Q \times \Sigma \to Q$

$q_0 \in Q$ Initial state

~~$F \subseteq Q$ Final state~~

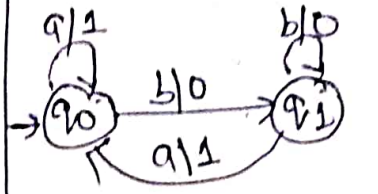
$\Delta \to$ Output alphabate

$\lambda \to$ output function

Mealy machine

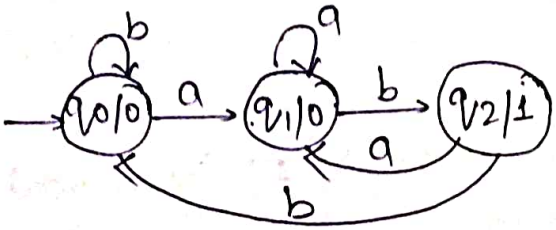
$\lambda : Q \times \Sigma \to \Delta$

$\lambda(t) = \lambda(q(t) \cdot x(t))$

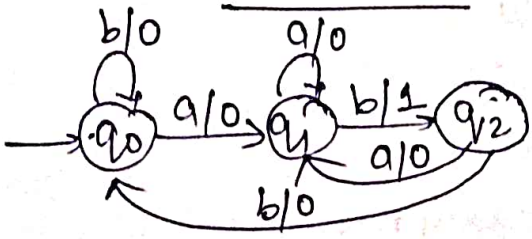


Q: Construct a Moore machine that takes set of all string over $\Sigma = \{a, b\}$ as i/p and print '1' as o/p for every occurance of 'ab' as a substring

Moore.



Moore M/C



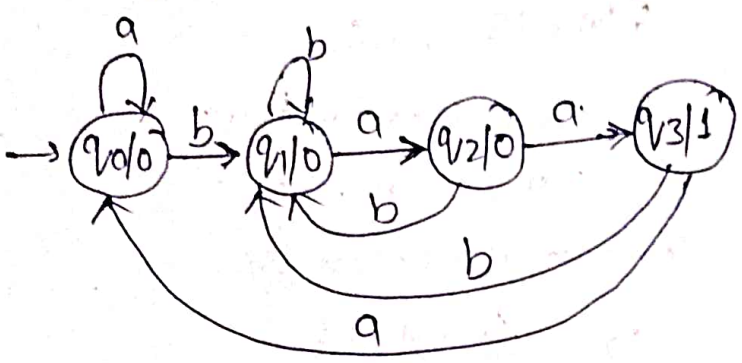
Mealy M/C

⇒ Transition Table

PS (Q)	Next state a	Next state b	Output Δ
q_0	q_1	q_0	0
q_1	q_1	q_2	0
q_2	q_1	q_0	1

P.T.O

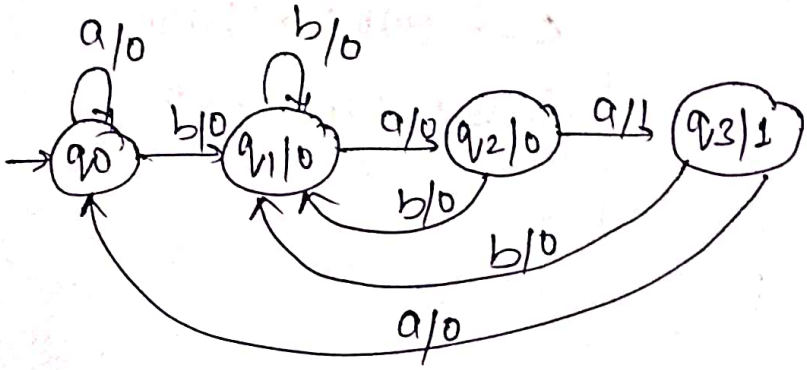
Q. Construct a Moore M/c that takes set of all string over {a,b} and counts no. of occurance of substring 'baa'



Moore M/c

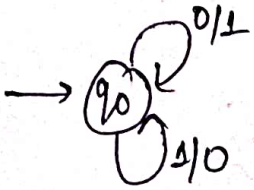
T.F. P.S.	Next State a	Next State b	O/P
→ q_0	q_0	q_1	0
q_1	q_2	q_1	0
q_2	q_3	q_1	0
q_3	q_0	q_1	1

Q.

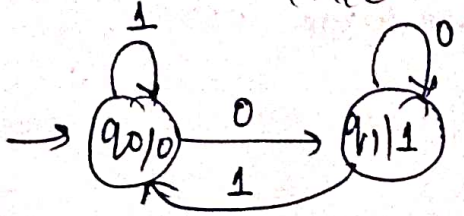


Mealy M/c

Q. Design a Mealy M/c which prints 1's Complement of input string over alphabate $\Sigma = \{0,1\}$

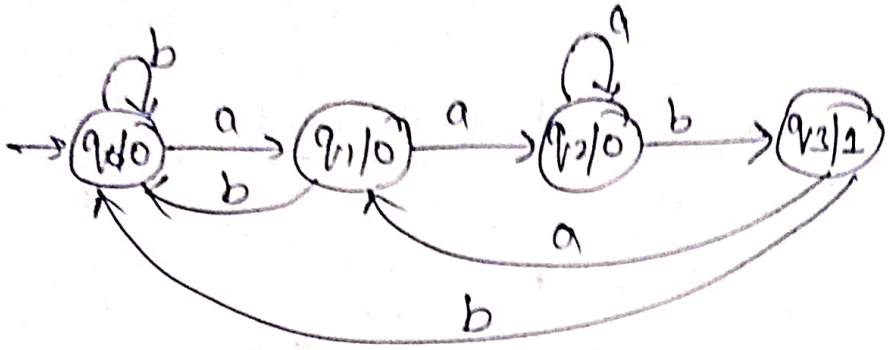


Mealy M/c

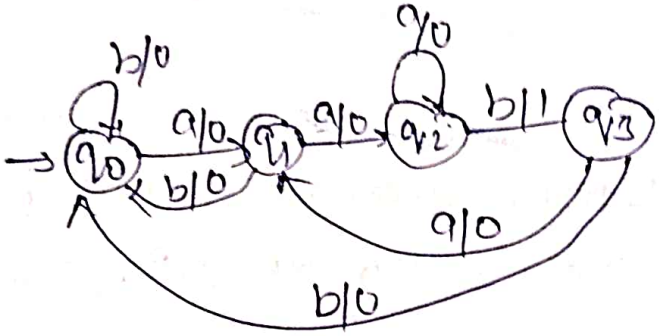


Moore M/c

Design a Moore MIC Which Count the accurance of substring aab in I/P string

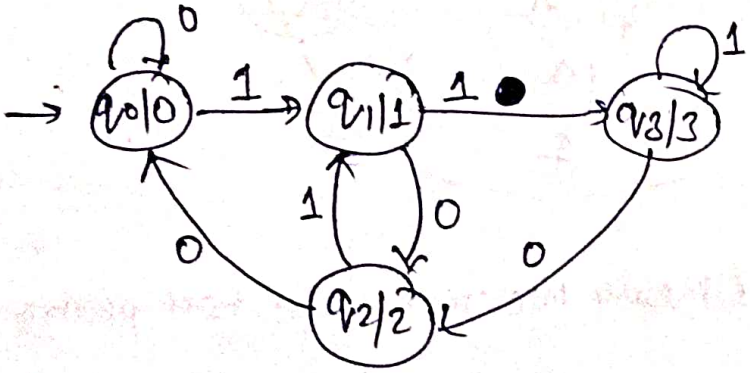


Moore MIC



Mealy MIC

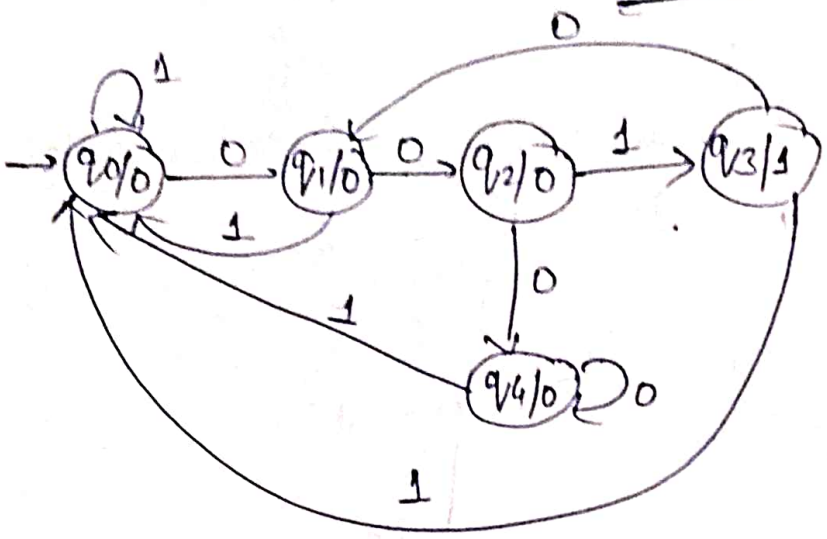
Q. Construct a Moore MIC Which Calculate residue mode 4 for each binary string treated as binary integer.



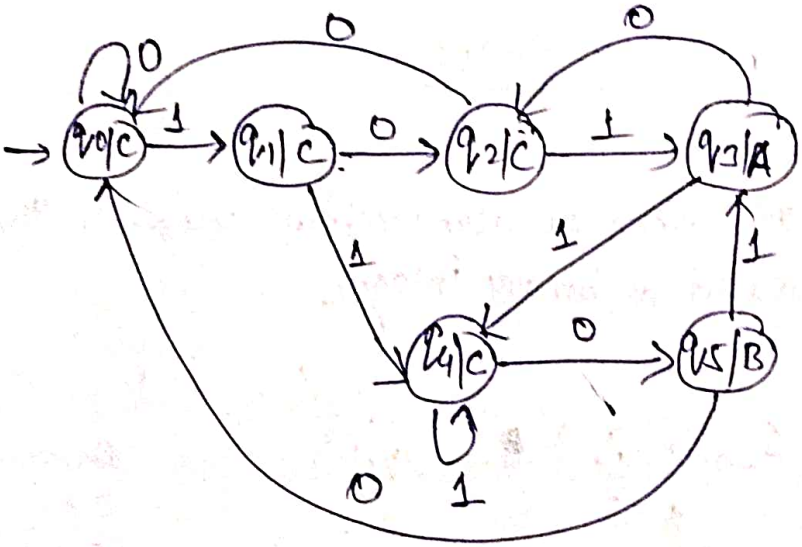
000 — 0
001 — 1
010 — 2
011 — 3
100 — 4
101 — 5
110 — 6
111 — 7

P.T.O

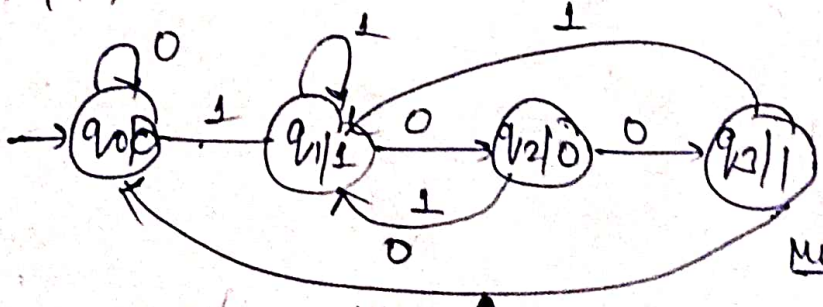
Q. Design a Moore M/c that gives o/p 1 if input of bin. sequence a '1' is preceded by exactly two zero's



Q. Design a Moore M/c for a binary I/p sequence if it ends in 101 o/p is A, if it ends in 110 o/p is B.

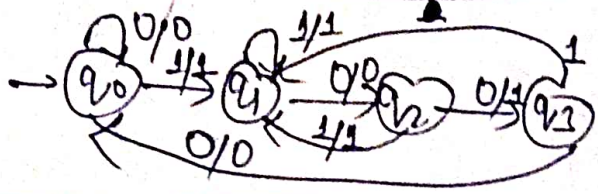


Q. Design a Moore & Mealy M/c to Convert each accurance of Substring 100 by 101



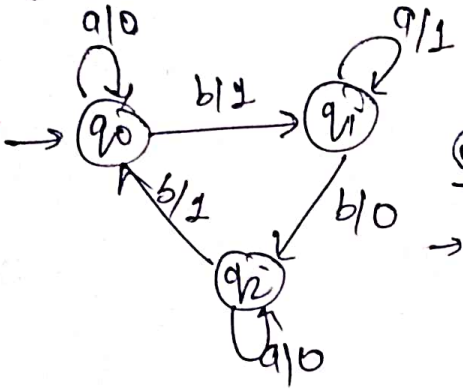
Moore M/c

Mealy M/c →



Conversion of Mealy to Moore

Q. Convert the given Mealy M/c into to Moore M/c



Mealy M/c Transition table

P.S	NS a/p(x)		NS O/P(λ)
→ q_0	q_0	0	q_1 1
q_1	q_1	1	q_2 0
q_2	q_2	0	q_0 1

Note:

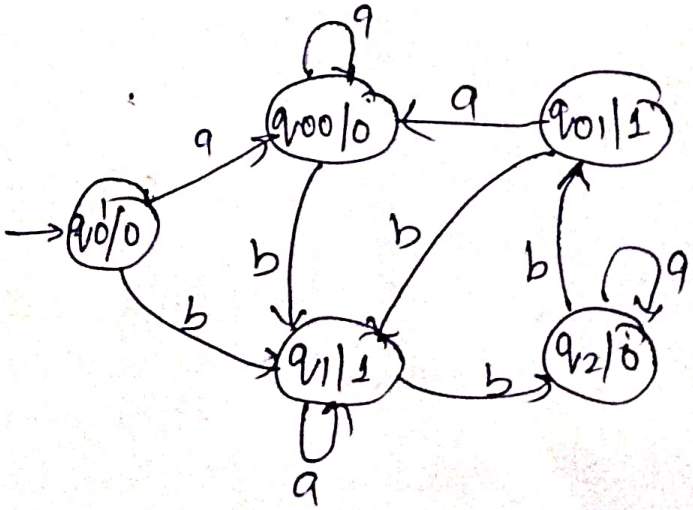
In the transition table of Mealy M/c we found q_0 generated two different types of O/P, now we reconstruct Mealy M/c transition table.

Updated Mealy M/c transition table:

Present state	a NS	a O/P	b NS	b O/P
→ q_{00}	q_{00}	0	q_1	1
q_{01}	q_{00}	0	q_1	1
q_1	q_1	1	q_2	0
q_2	q_2	0	q_{01}	1

Moore M/c transition table:

P.S	NS a	NS b	O/P
→ q_{00}	q_{00}	q_1	0
q_{00}	q_{00}	q_1	0
q_{01}	q_{00}	q_1	1
q_1	q_1	q_2	1
q_2	q_2	q_{01}	0



Moore M/c