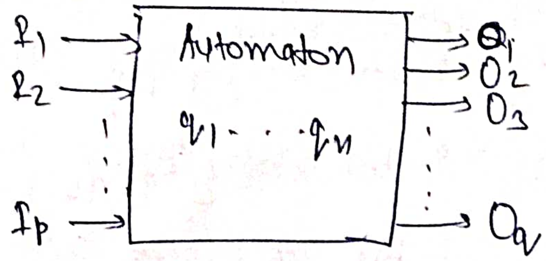
Automata

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.

Ex: Automatic machine tools, Automatic packing machines & Automatic photo printing machine etc.



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

(ii) **O/P :-** $O_1, O_2 \cdots O_q$ 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 $q_1, q_2, \cdots q_n$

(iv) **State Relation:** The next state of the automaton at any ~~stat~~ 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, bbb}
$= 2^3 = 8$

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

Language :. An English Language is collection of words, but the m/c 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, ... }

(Lecture : 02)

Deterministic Finite Automata:

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

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

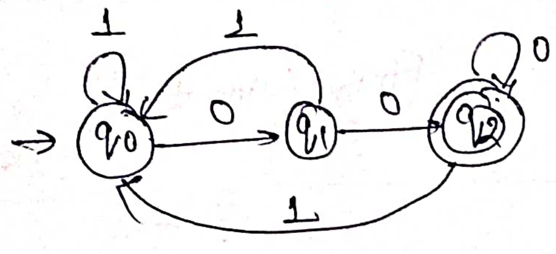
$\Sigma \rightarrow$ is a finite non-empty set of I/P alphabate

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

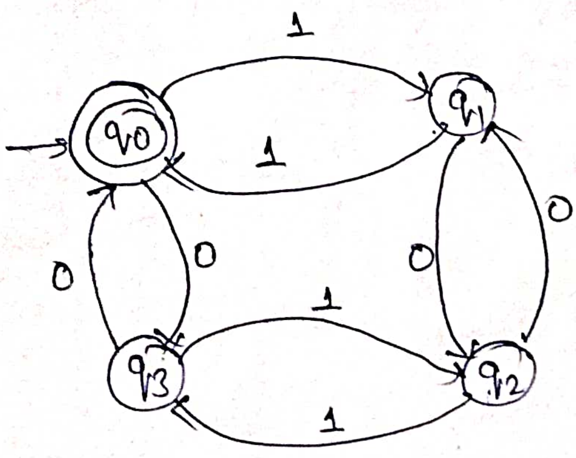
$q_0 \in Q$ is the initial state

$q_f \subseteq Q$ 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 $\Sigma = \{0,1\}$.



Powers of Σ (Sigma) :

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

Σ^0 = ~~[illegible]~~ 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 2 $\{aa, ab, ba, bb\}$ = $\Sigma \cdot \Sigma$ = $\{a,b\} \cdot \{a,b\}$

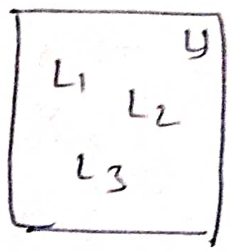
Σ^3 = for length 3

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

$\Sigma \cdot \Sigma \cdot \Sigma$ =

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

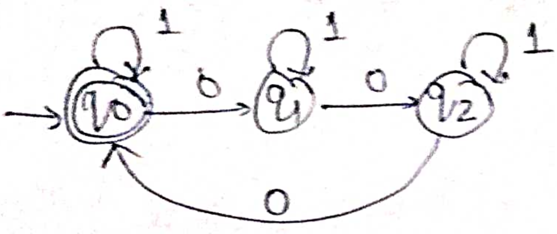
= Universal set



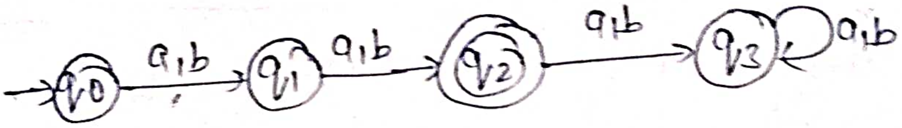
Finite Automata

- Finite Automata With output
  - Moore Machine
  - Mealy Machine
- Finite Automata Without output
  - DFA
  - NFA
  - ϵ-NFA

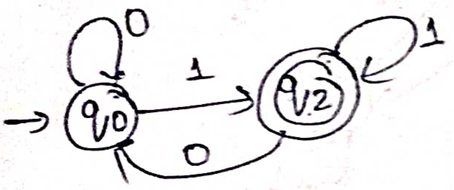
Q. Construct A DFA that accept set of string where the no. of 0's in every string is multiple of 3 over alphabet $\Sigma = \{0,1\}$



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



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



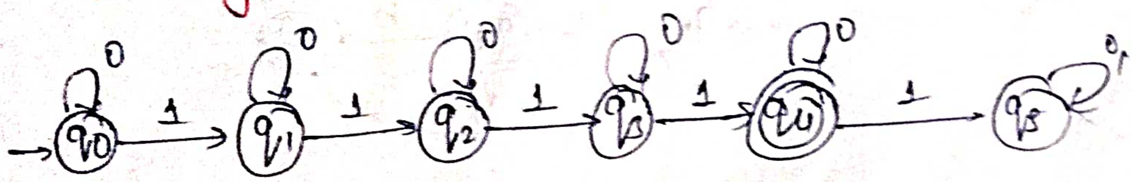
Q Construct a DFA which accept set of string containing (i) exactly four 1's in every string over alphabate $\Sigma = \{0,1\}$

At least

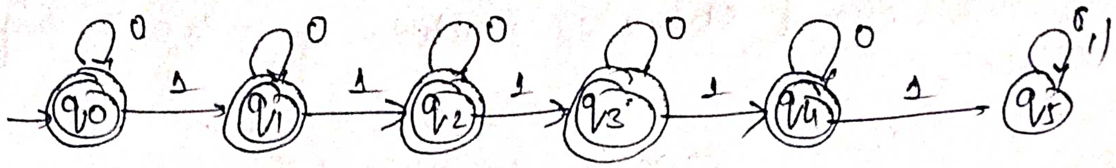
At Most

(ii) At Most four 1's

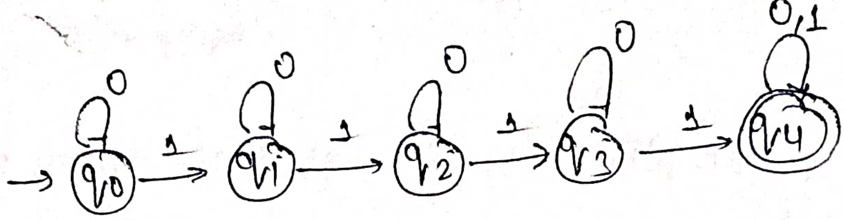
(iii) At lest four 1's



(ii) At Most four 1's



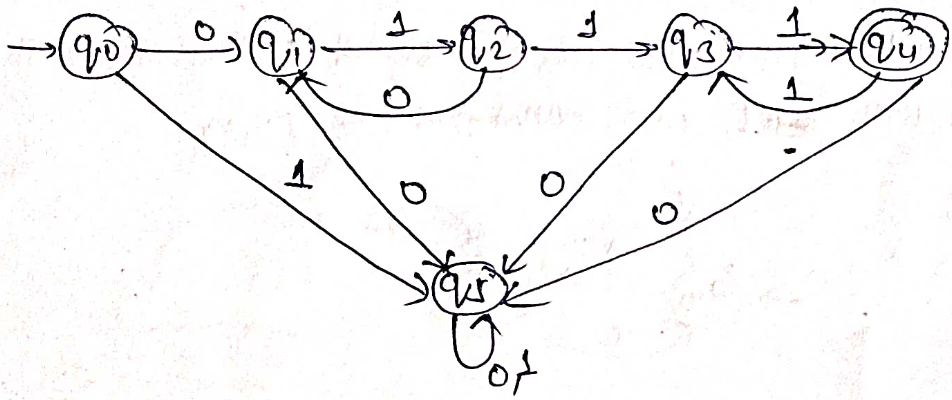
(iii) At lest four 1's



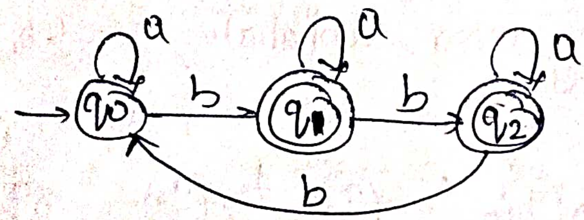
(Lecture-03)

Q. Design a FA for the language.

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

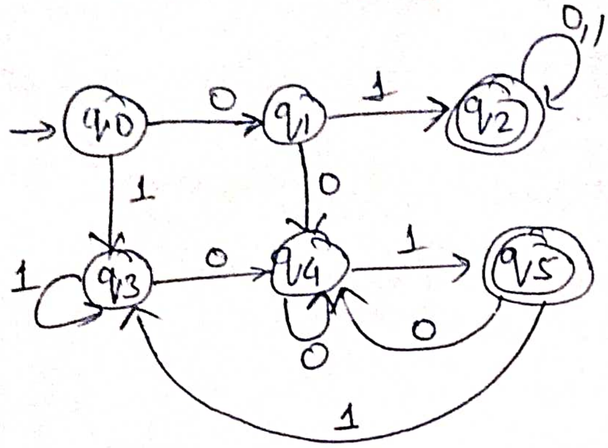


Q. Design a FA for the language $L = \{ w \in (a,b)^* \mid n_b(w) \bmod 3 > 0 \}$

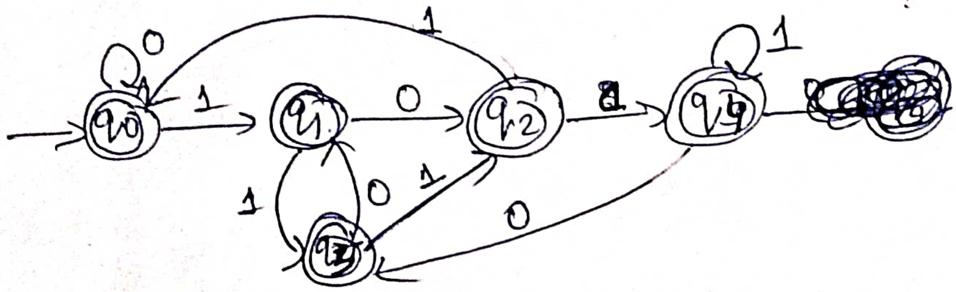


(4)

Q. 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 | W ∈ (0,1)* }



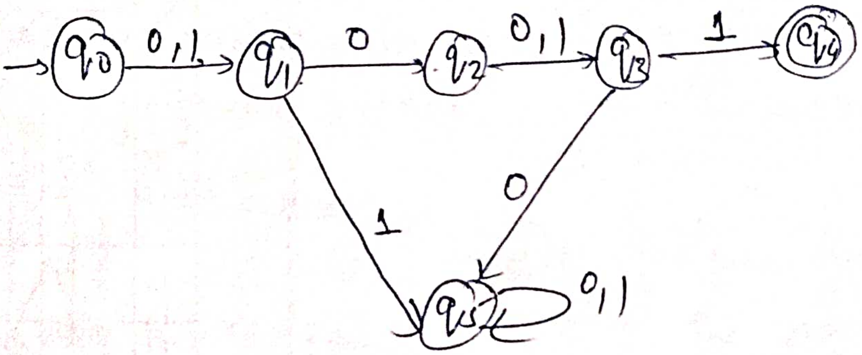
0000 → 0
0001 → 1
0010 – 2
0011 – 3
0100 – 4
0101 – 5
0110 – 6
0111 – 7
1000 – 8
1001 – 9
1010 – 10
1011 – 11
1100 – 12
1101 – 13
1110 – 14
1111 – 15

Automata

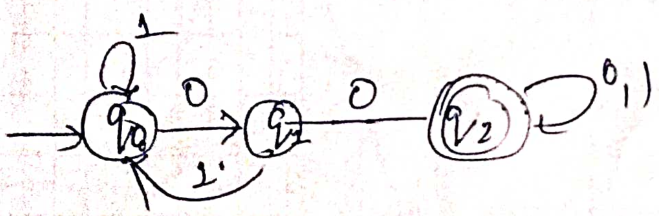
Q Design a FA Which accepts the language.

$L = \{ w \in (0,1)^* \mid$ 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 $\}$

NFA (Non-Deterministic finite Automata)

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

$Q \rightarrow$ Non Empty set of states

$\Sigma \rightarrow$ " " " " I/P Alphabate

$\delta \rightarrow$ Transition function shown the mapping btw ~~$Q \times \Sigma \rightarrow 2^Q$~~ $Q \times (\Sigma \cup \{\epsilon\}) \rightarrow 2^Q$

$q_0 \in Q$ 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 $\Sigma = \{a, b\}$

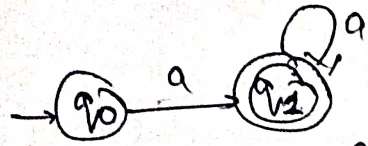
(i) $L_1 = \{$ Start With a $\}$

(ii) $L_2 = \{$ Containing a $\}$

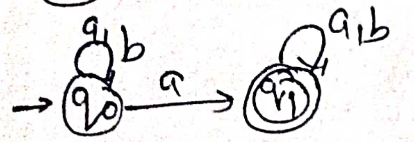
(iii) $L_3 = \{$ end with a $\}$

(IV) $L_4 = \{$ Start With ab $\}$

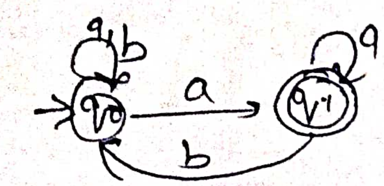
(i)



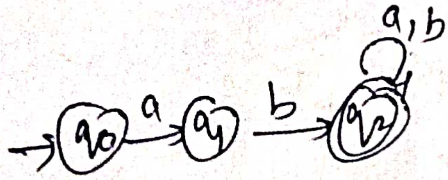
(ii)



(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.

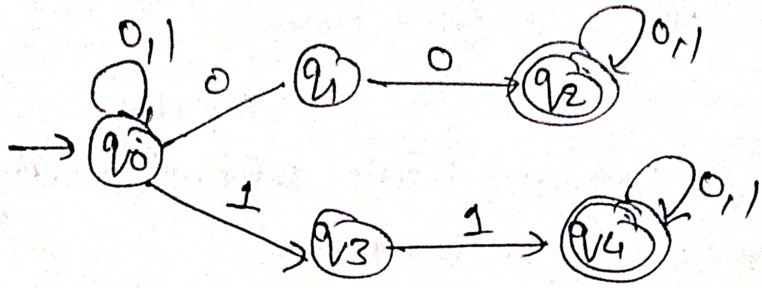
Let q be an state of Q and $a \in \Sigma$ then.

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

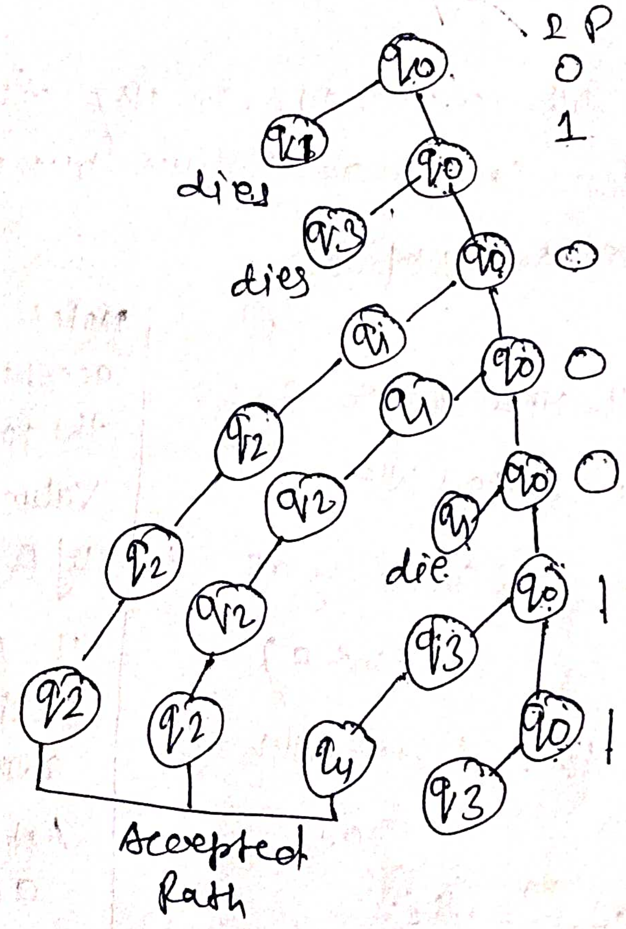
$$\delta(q, a) \rightarrow q_0,\ q_1,\ q_2,\ \ldots,\ q_n$$

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

Ans:

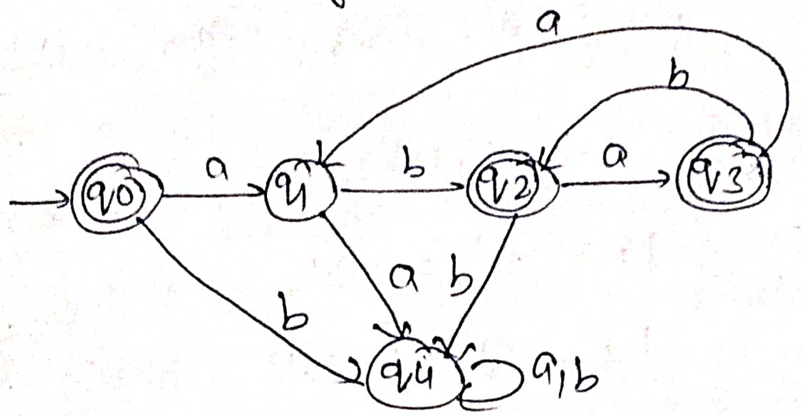


The tree of states this NFA is in for the I/P 0100011

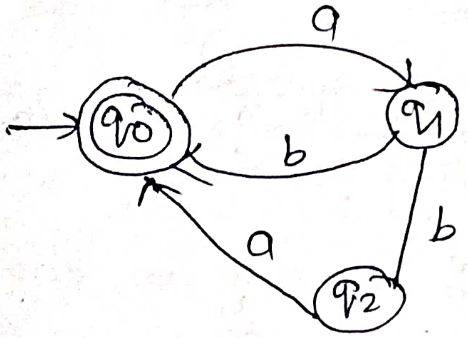


Q. Design a NFA for the language $L = (ab \cup aba)^*$.

Ans



DFA



NFA

Q. Draw the state diagram for NFA accepting string.

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

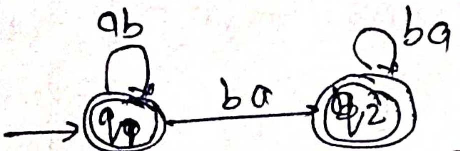
Ans: We construct NFA for the language L in two paths i.e

$L = L_1 \cup L_2$

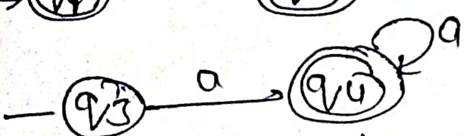
$L_1 = (ab)^*(ba)^*$

$L_2 = aa^*$

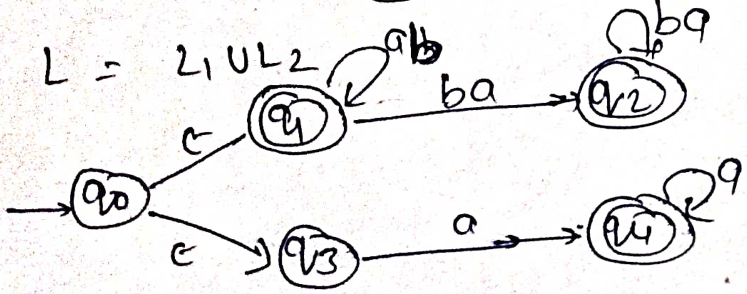
$L_1 \rightarrow$



$L_2 \rightarrow$



$L = L_1 \cup L_2$



Q. Find NFA with four state for the language.

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

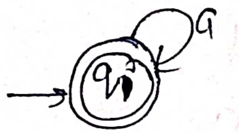
Ans:

$$L = L_1 \cup L_2$$

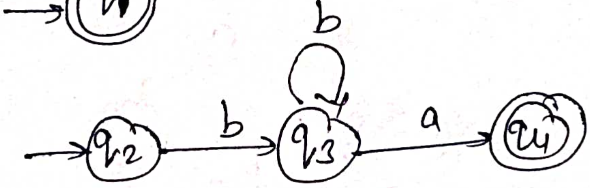
$$L_1 = a^n : n \geq 0$$

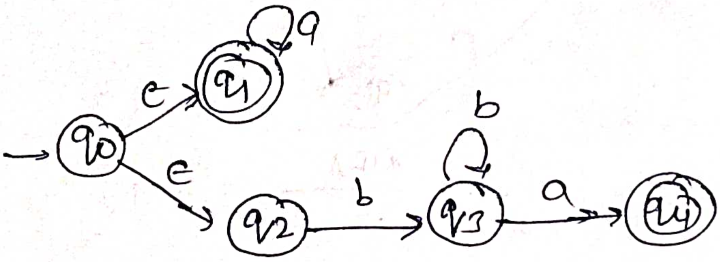
$$L_2 = b^n a : n \geq 1$$

$L_1 \rightarrow$



$L_2 \rightarrow$





⇒ 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 $\{q_1, q_2, \dots q_n\}$ where $q_1, q_2, \dots q_n \in Q$ }

(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

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

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

(Lecture-06)

⇒ 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 $M = (Q, \Sigma, \delta, q_0, q_f)$ does, We may proceed as follows:

Step 1 - Initially $Q' = \phi$

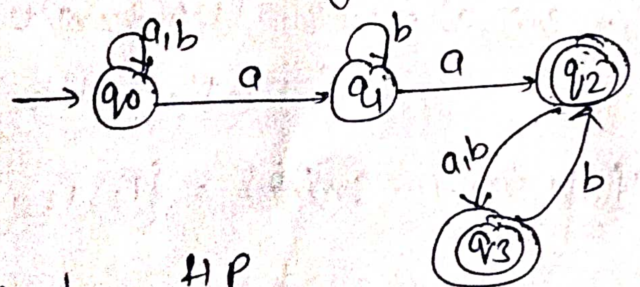
Step 2 - Put $[q_0]$ into Q'. $[q_0]$ is the initial state of DFA M'.

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'

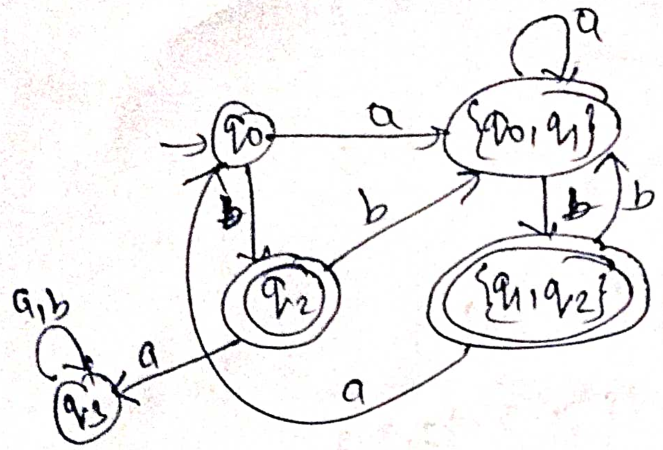
Note: The states Which are not reached from the initial state should not be included in Q'. Thus the set of states (Q') is not necessarily equal to 2^Q.

Q. Convert the following NFA into DFA

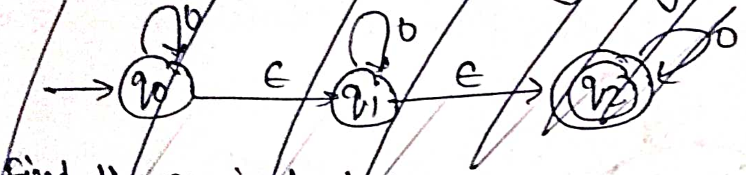


Ans:

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.1 Consider the NFA given by following diagram~~



~~find the equivalent N.~~

(Lecture-06)

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 made without symboles are called ϵ-transition.

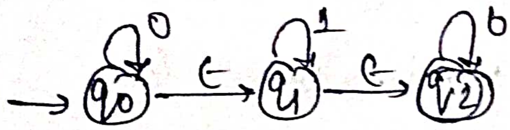


fig.1

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.

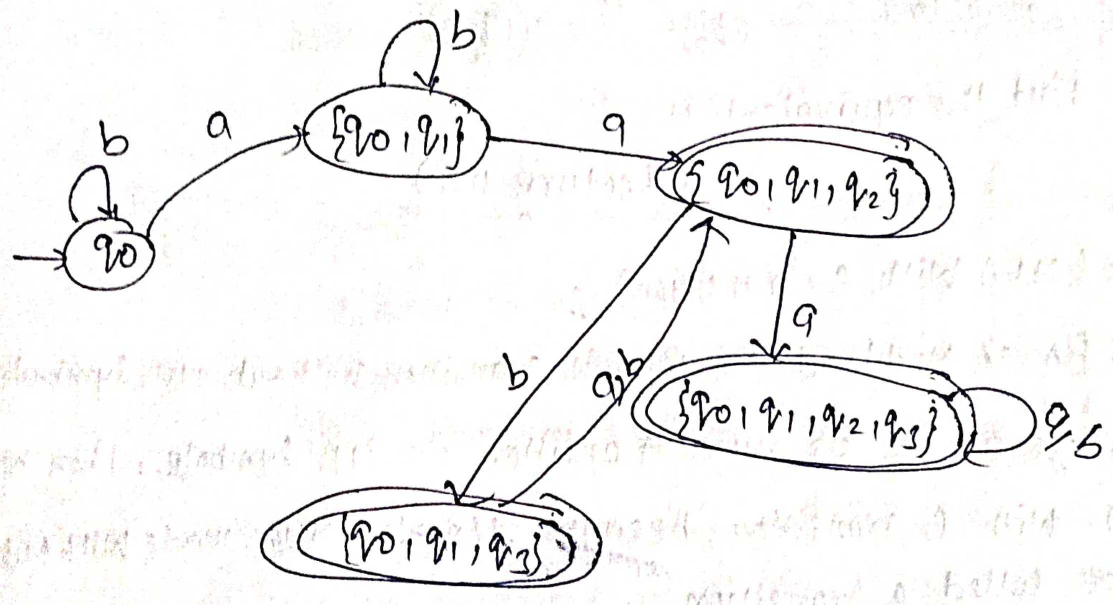
ϵ-tuple $(Q, \Sigma, \delta, q_0, q_f)$

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

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

Q	I/P a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	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
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_2\}$
q_1	$\{q_0\}$	$\{q_1\}$
q_2	ϕ	$\{q_0, q_1\}$

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

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

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

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

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$(q_2) = \{q_2\}$

$$\begin{aligned}
\delta'(q_0, 0) &= \epsilon\text{-closure}(\delta(\delta(q_0, \epsilon), 0)) \\
&= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_0), 0) \\
&= \epsilon\text{-closure}(\delta(q_0, q_1, q_2), 0) \\
&= \epsilon\text{-closure}(\delta(q_0, 0) \cup \delta(q_1, 0) \cup \cup \delta(q_2, 0)) \\
&= \epsilon\text{-closure}(q_0 \cup \phi \cup \phi) \\
&= \epsilon\text{-closure}(q_0) \\
&= \{q_0, q_1, q_2\}
\end{aligned}$$

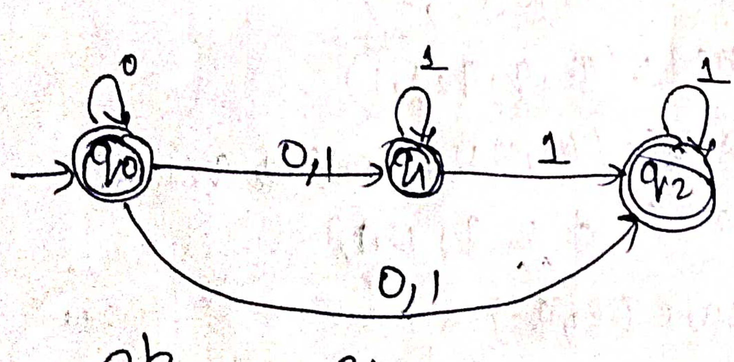
$$\begin{aligned}
\hat{\delta}(q_0, 1) &= \epsilon\text{-closure}(\delta(\delta(q_0, \epsilon), 1) \\
&= \epsilon\text{-closure}(\delta(q_0, q_1, q_2), 1) \\
&= \epsilon\text{-closure}(\delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1)) \\
&= \epsilon\text{-closure}(\phi \cup q_1 \cup q_2) \\
&= \epsilon\text{-closure}(q_1, q_2\} \\
&= \cancel{\{q_1, q_2\}}\ \epsilon\text{-closure}(q_1) \cup \epsilon\text{-closure}(q_2) \\
&= \{q_1, q_2\} \cup \{q_2\} \\
&= \{q_1, q_2\}
\end{aligned}$$

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

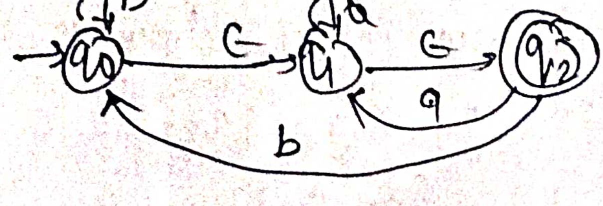
$\hat{\delta}(q_1,1) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1,\epsilon),1))$
$= \epsilon\text{-closure}(\delta(q_1,q_2),1)$
$= \epsilon\text{-closure}(\delta(q_1,1) \cup \delta(q_2,1))$
$= \epsilon\text{-closure}(q_1,q_2)$
$= \epsilon\text{-closure}\{q_1\} \cup \epsilon\text{-closure}(q_2)$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

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

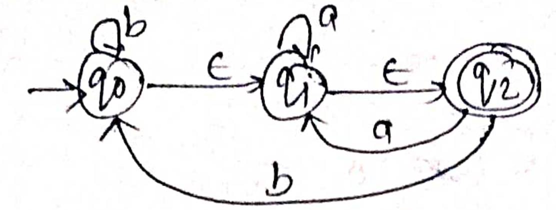
$\hat{\delta}(q_2,1) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_2,\epsilon),1))$
$= \epsilon\text{-closure}(\delta(q_2,1))$
$= \epsilon\text{-closure}(q_2)$
$= q_2$



Q.



Q. Convert the given NFA with ϵ to NFA without ϵ. (6)



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

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

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

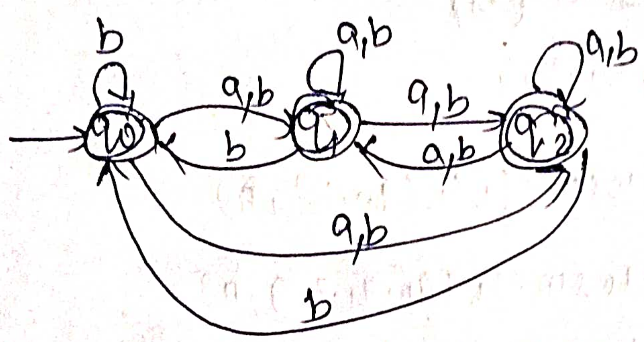
$\hat{\delta}(q_1, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, \epsilon), a))$
$= \epsilon\text{-closure}(\delta(\{q_1, q_2\}, a))$
$= \epsilon\text{-closure}(\delta(q_1, a) \cup \delta(q_2, a))$
$= \epsilon\text{-closure}(\phi \cup q_1)$
$= \{q_1, q_2\}$

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

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

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

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



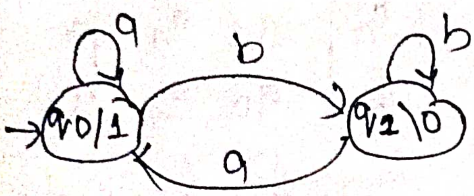
NFA - With out ε

FA With output

Moore machine

$\lambda : Q \rightarrow \Delta$

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



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

$Q \rightarrow$ Finite set of state

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

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

$q_0 \in Q$ initial state

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

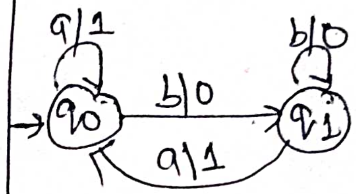
$\Delta \rightarrow$ Output alphabate

$\lambda \rightarrow$ output function

Mealy Machine

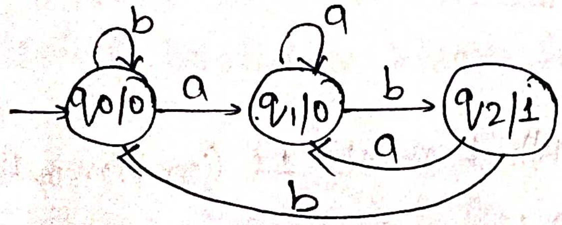
$\lambda : Q \times \Sigma \rightarrow \Delta$

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



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

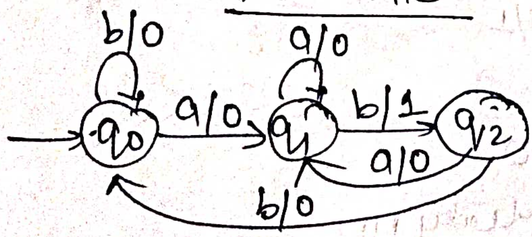
~~Moore~~.



Moore M/C

$\Rightarrow$ 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



Mealy M/C

Q.