FORMAL LANGUAGE AND AUTOMATA THEORY

Formal languages are defined by two sets of rules.

- Syntax : precise rules that tell you the symbols you are allowed to use and how to put them together into legal expression.
- semantics :- precise rules that tell you the meaning of the symbols and legal expressions.

Automata theory :-

→ is the study of abstract computing devices of "machines" as well as the computational problems that can be solved using them.

→ automata – Greek word → self-acting.

Finite automata → Devices with a small amount of memory. These are very simple machines.

Push-down automata → Devices with unbounded memory that can be accessed in a restricted way. Used to parse grammars.

Turing machines → Devices with unbounded memory These are actual computers.

Time-bounded Turing Machines:

→ Devices with unbounded memory but bounded running time.

→ These are computers that run fast.

Computational Models:-

1. Finite Automation (FA)
   → recognizes regular languages.
2. Push Down Automation (PDA)
   → recognizes Context Free languages.
3. Turing Machines (TM)
   → recognizes Computable languages

Deterministic Finite automata:-(DFA)

DFA is a five-tuple

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

Q → finite set of states

Σ → finite input alphabet.

q_0 → intial/starting state

q_0 is in Q.

$F \rightarrow$ set of final state
which is subset of Q.

$\delta \rightarrow$ A transition function

$\delta : (Q \times \Sigma) \rightarrow Q$

δ is defined for any q in Q and s in Σ

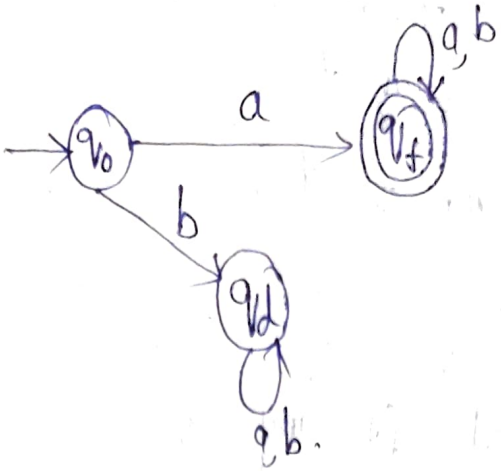
$\delta(q, s) = q'$, q' is another state in Q.

Transition diagram:-

A transition diagram for a DFA $A = (Q, \Sigma, \delta, q_0, F)$ is a graph defined as follows:

- → for each state in Q there is a node.
- → For each state q in Q and each input symbol a in Σ, Let $\delta(q, a) = p$ Then the transition diagram has an arc from node q to node p labeled a.
- → There is an arrow into state q_0 Labeled start.
- → Final state(F) are marked as double circles.

Eg:-



Transition table:- (TT)

. A TT is a conventional tabular representation of a function like δ that takes two arguments and returns a value.

	0	1
$\rightarrow q_0$	q_f	D
$*q_f$	q_f	q_f
D	D	D.

Design of DFA:-

Step 1:- List out the possible strings in the given language.

Step 2:- Decide the no of states by identifying the length of minimum possible string.

* If all string starting with 'n' length substring → (n+2) states
* If all string ending with 'n' length substring → minimum (n+1) states.

Step 3:- Draw the state and while constructing a DFA:

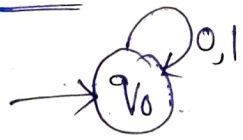
- Always prefer to use existing path.
- Create a new path only when there exists no path to go with.

→ For the strings starting with 'n' length substring send all the left possible combinations to the dead state.

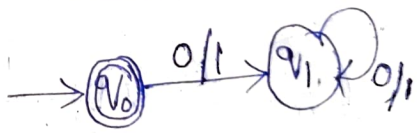
→ For the strings that ends with 'n' length substring send all the left possible combinations to the starting state.

* Let $\Sigma = \{0, 1\}$ Give DFA's for $\{\}$, $\{\varepsilon\}$, Σ^* and Σ^+

For $\{\}$:-

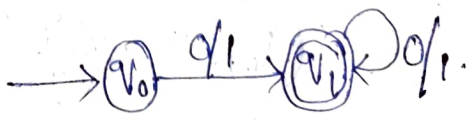


For $\{\varepsilon\}$:-



For Σ^*

For Σ^+

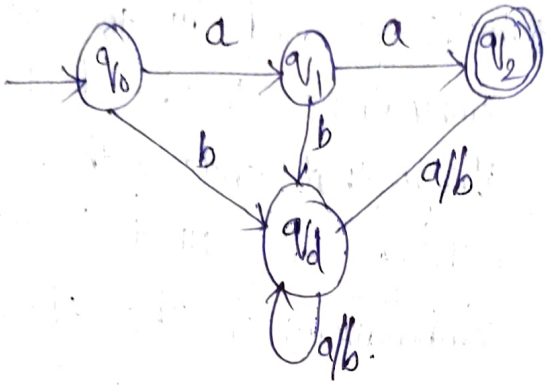


Examples for Construction of DFA:-

1. $\Sigma = \{a, b\}$ ⇒ possible strings

   $L = \{aa\}$ ⇒ Language.

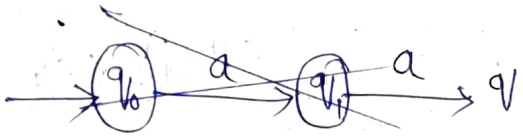
   Minimum states = 2+1 = 3.

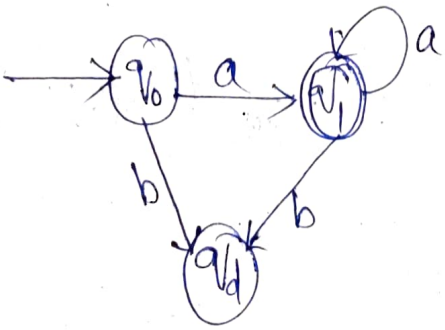


2. $\Sigma = \{a, b\}$

   $L = \{a, aa, aaa, aaaa, \ldots\}$

   Minimum states = 1+1 = 2

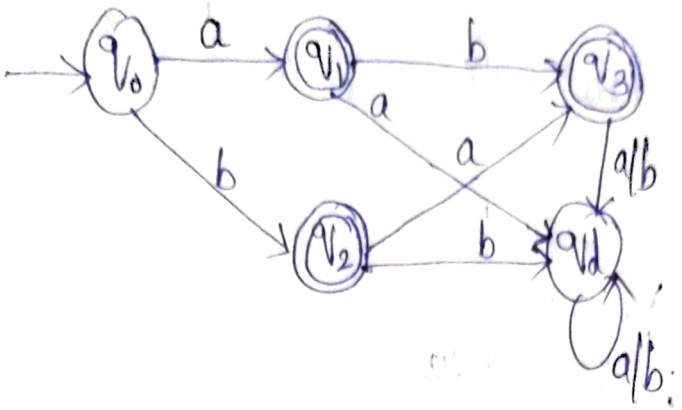




3. $\Sigma = \{a, b\}$

$L = \{ab, ba, a, b\}$

Minimum states = 2



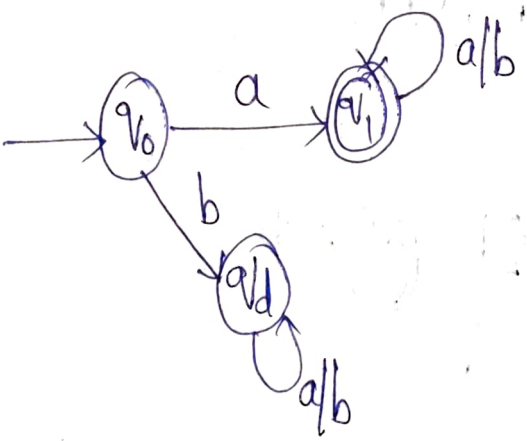
4. $\Sigma = \{a, b\}$

$L = \{aw \mid w \in (a+b)^*\}$.

$L = \{a, aa, ab, aab, aaa, \ldots\}$.

Minimum states = $n+1$

$= 1+1 = 2$.



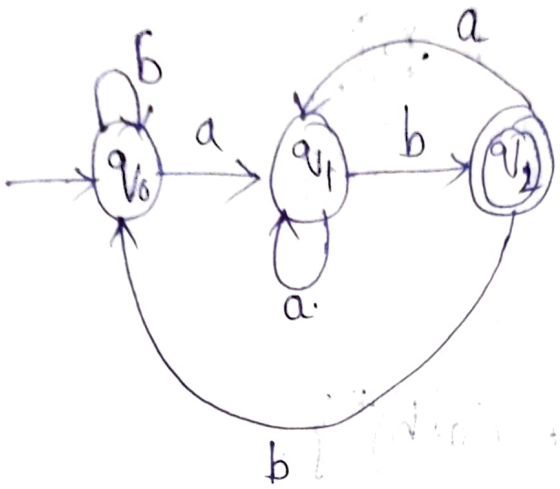
(5) $L = \{ w\,ab \mid w \in (a+b)^* \}$

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

Minimum states $= n+1$

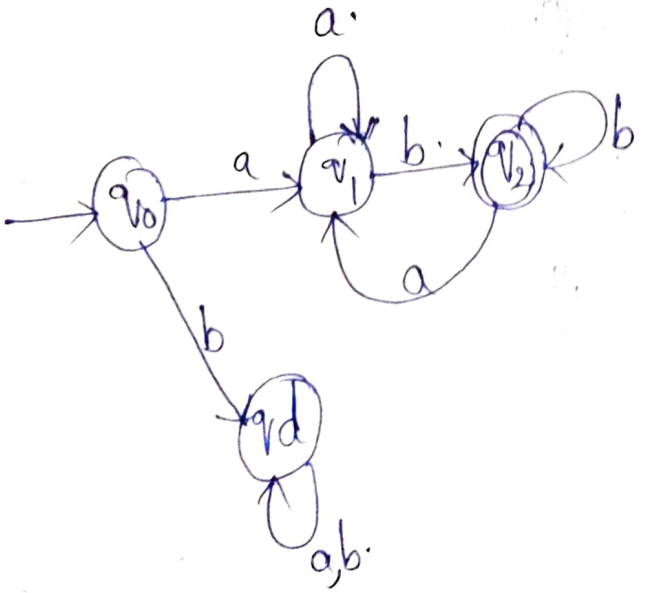
$= 2+1 = 3$.

$L = \{ab, aab, bab, \ldots\}$



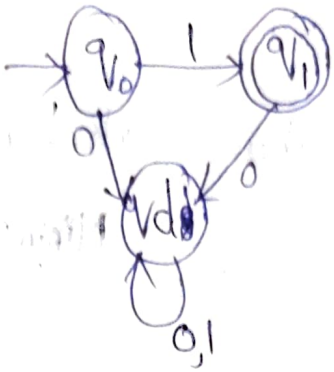
(6) $L = \{ awb \mid w \in (a+b)^* \}$.

$L = \{ab, aab, abb, aabb, abab, \ldots\}$.

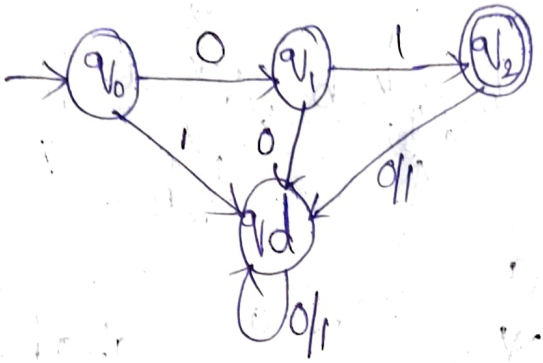


7) $\Sigma = \{0, 1\}$.

1) $L = \{1\}$

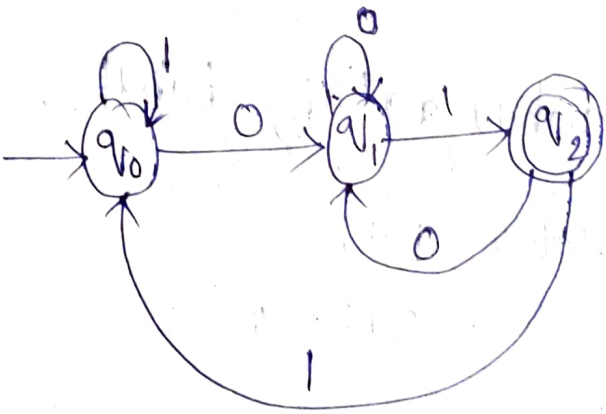


2) $L = \{0, 1\}$.



3) $L = \{ w/w$ is a string of 0s & 1s ending with the string 01 $\}$.

$L = \{01, 001, 101 \ldots\}$.



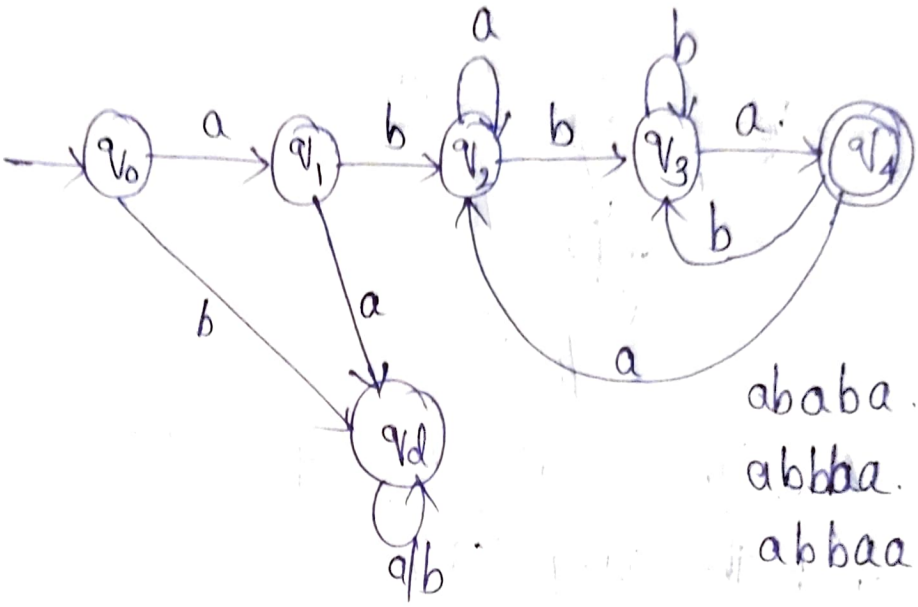
(8) $L = \{ab\,w\,ba \mid (a+b)^*\}$.

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

$L = \{abba, ababa, abbba, \ldots\}$

Minimum states $= n+1$

$= 4+1 = 5$



ababa.
abbba.
abbaa

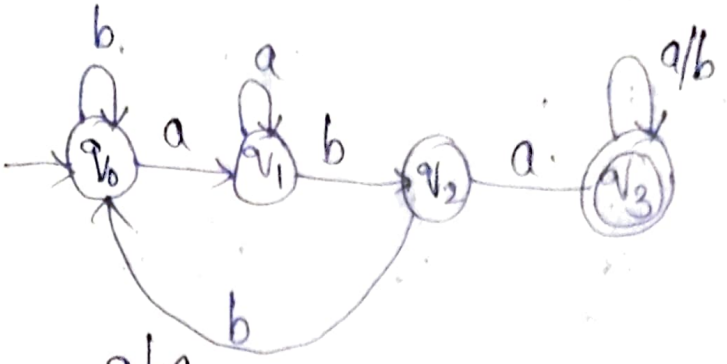
(9) $L = \{ w\,aba\,w \in (a+b)^* \}$.

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

$L = \{aba, aabaa, aabab, babab, \ldots\}$

Minimum states $= n+1$

$= 3+1 = 4$.

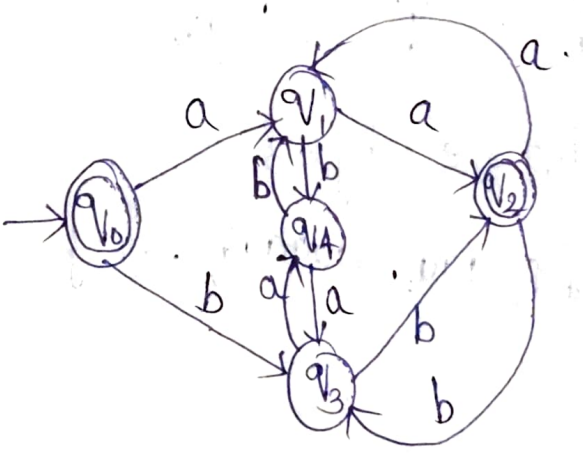


aba
baba
aaba.
abbaba.
abaa
ababa.

(10.) Even number of a's and ~~odd~~ even number of b's

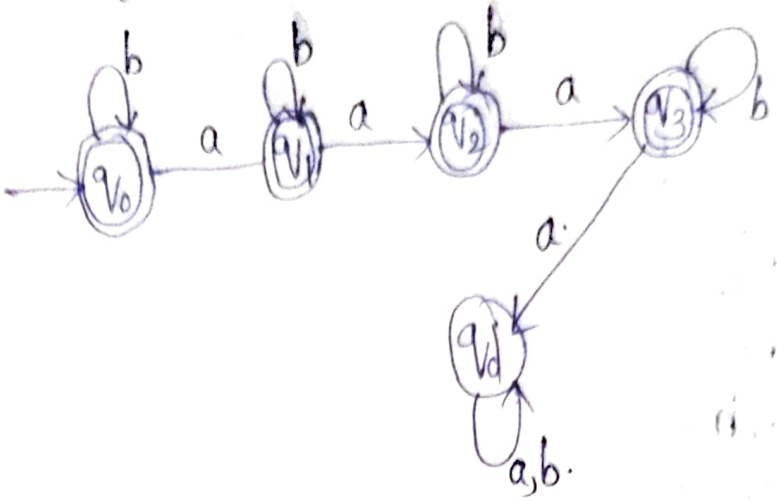
Minimum states = n+1
= 0+1
= 1

{ϕ, aabb, aaaabb, ...}



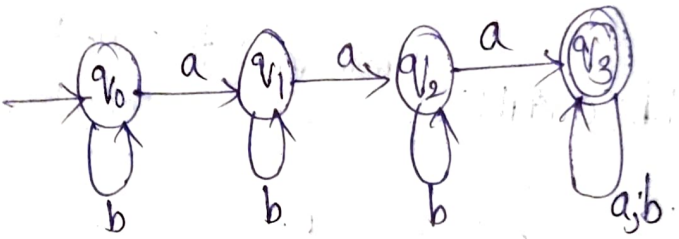
11. Atmost 3 a's.

~~$L = \{aaa, aaaa, b, \ldots\}$~~ $\{\phi, a, aa, aaa, \}$



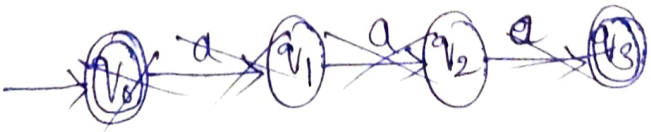
12. Atleast 3 a's

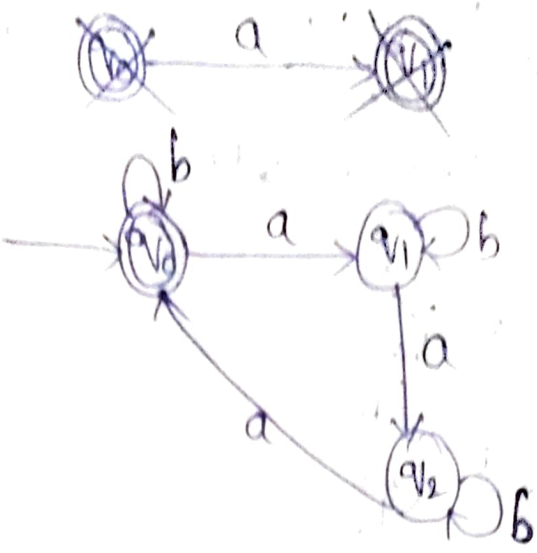
$\{aaa, aaab, \ldots\}$



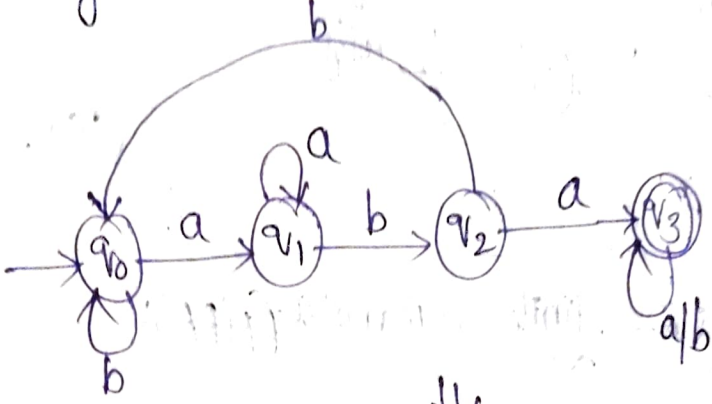
13. No of a's divisible by 3.

$\{\phi, aaa, aaaaaa, \ldots\}$





(14) string doesnot contain aba as substring.

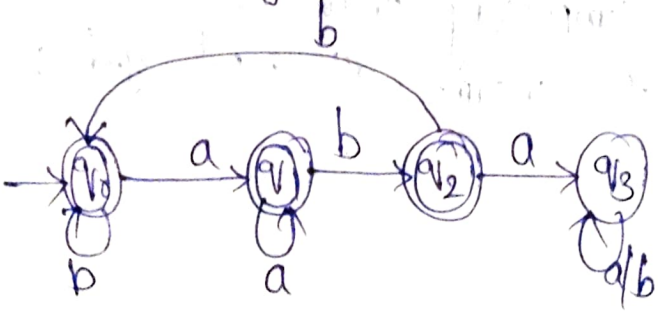


baabbaba.
abaa

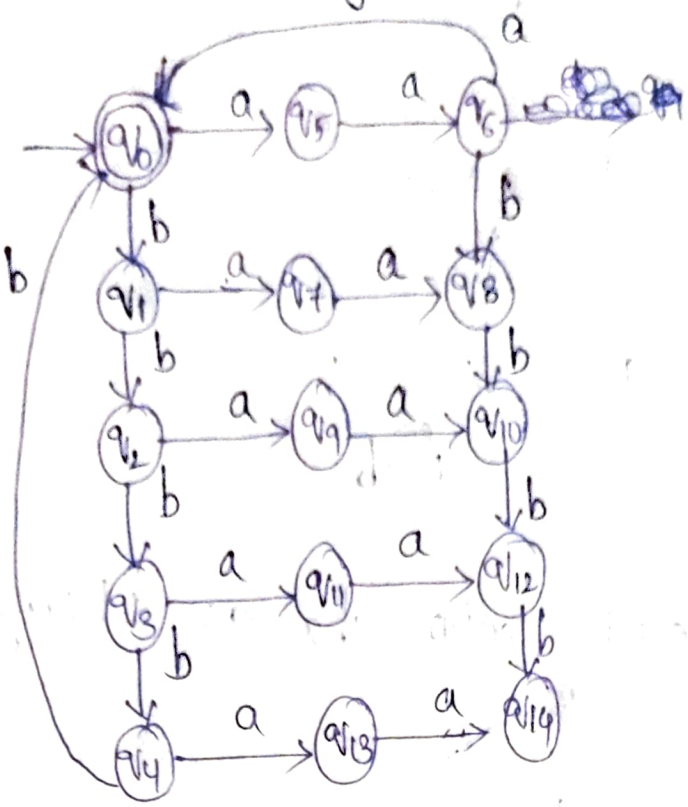
⇓

string contain "aba" as substring.

for string not containing "aba" substring.



(15) As divisible by 3 & Bs divisible by 3



(X).

31/08/20

Non-deterministic finite automata (NFA):-

→ In NFA, one input symbol causes to move more than one state or a state may contain more the one transistion from same input symbol.

→ It is not compulsory that all the states have to consume all input symbols in Σ.

→ A Non deterministic finite automata is represented by a 5 tuple $M = (Q, \Sigma, \delta, q_0, F)$

Q is finite set of states

Σ is finite input alphabet

δ is transition mapping function

q_0 initial state

F set of final states.

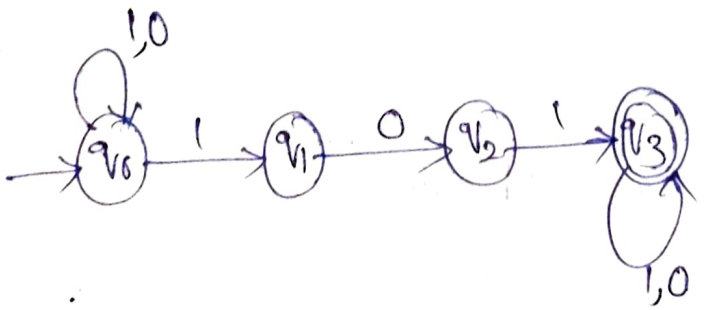
* An NFA accepts a string "w" if it is possible to make any sequence of choices of next state, while reading the charachters of w and go from start state to any accepting states..

① Construct NFA for

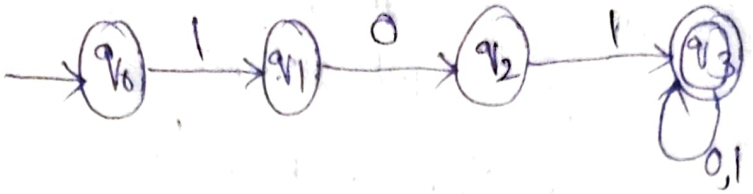
$L = \{ (1+0)^* \; 101 \; (1+0)^* \}$

Minimum possible string = 101

No of states = n+1 = 3+1 = 4

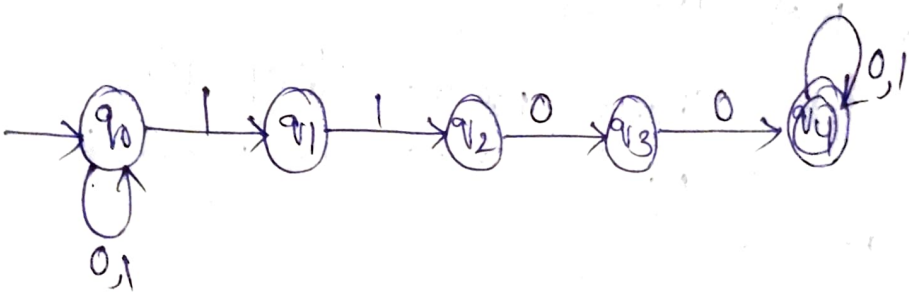


(2) $L = \{ 101 (1+0)^* \}$

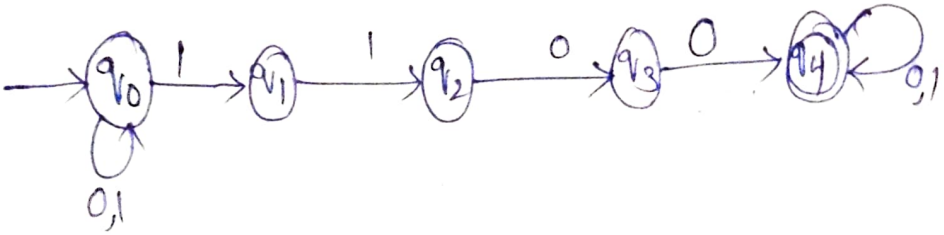


(3) $L = \{ 1100 \}$

(4) $L = \{ (0+1)^* 1100 (0+1)^* \}$



(5) $L = 1100$ as substring.



Equivalance between NFA & DFA:-

Step 1:-

Convert the given transition system into transition table where each state corresponds to a row & each input symbol corresponds to a column.

Step 2:-

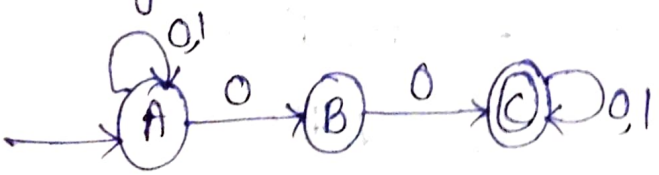
Construct the succession table which lists subsets of states reachable from the set of initial states.

Step 3:-

The transistion graph given by the sucessor table is the required deterministic system.

Eg:

1. Construct DFA equivalent to the following NFA.



Transition table :- (NFA)

state	0	1
→ A	A,B	A
B	C	ϕ
* C	C	C

Transition table :- (DFA) :-

	0	1
→ A	{A,B}	A
{A,B}	{A,B,C}	A
*{A,B,C}	{A,B,C}	{A,C}
*{A,C}	{A,B,C}	{A,C}

$$\delta(\{A,B\},0) = \delta(A,0) \cup \delta(B,0)$$
$$= A \cup B \cup C$$
$$= ABC$$

$$\delta(\{A,B\},1) = \delta(A,1) \cup \delta(B,1)$$
$$= A \cup \phi$$
$$= A$$

$$\delta(\{A,B,C\},0) = \delta(A,0) \cup \delta(B,0) \cup \delta(C,0)$$
$$= \{A,B\} \cup \{B\} \cup \{C\}$$
$$= \{A,B,C\}$$

$$\delta(\{A,B,C\},1) = \delta(A,1) \cup \delta(B,1) \cup \delta(C,1)$$
$$= A \cup \phi \cup C$$
$$= \{A,C\}$$

$$\delta(\{A,C\},0) = \delta(A,0) \cup \delta(C,0)$$
$$= \{A,B\} \cup \{C\}$$
$$= \{A,B,C\}$$

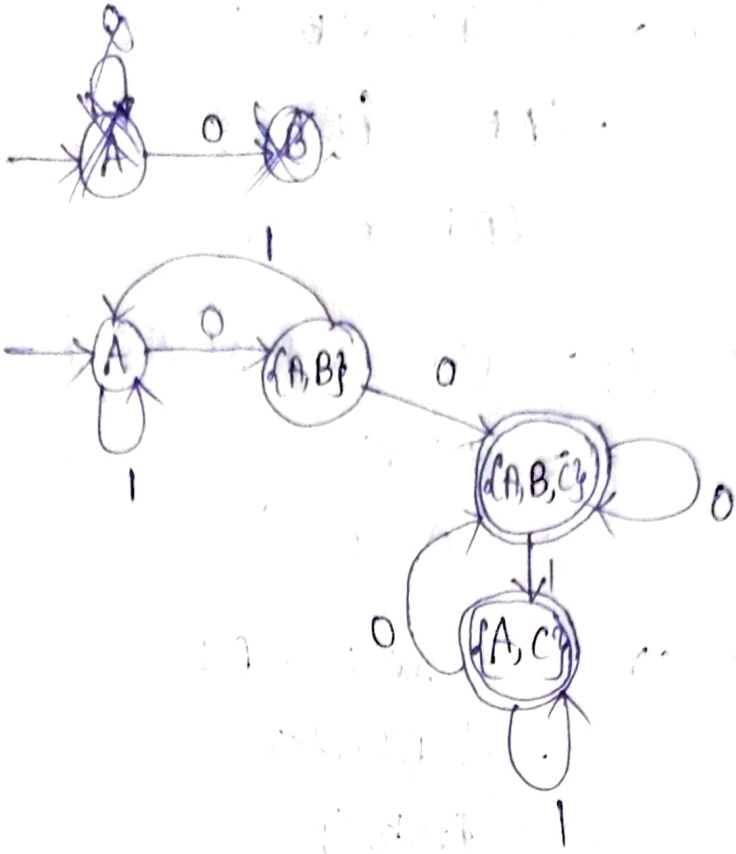
$$\delta(\{A,C\},1) = \delta(A,1) \cup \delta(C,1)$$
$$= A \cup C$$
$$= \{A,C\}$$

→ The process is to be repeated untill we didnot get any new states.

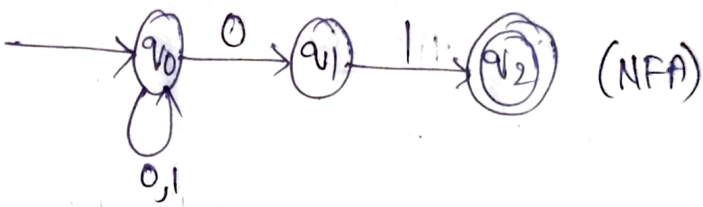
→ And the final state in NFA, is the final state in DFA.

(or).

If they contain single letter also they are considered as final states.



Q.



(NFA)

Transistion table :- (NFA)

		0	1
→	q_0	$\{q_0, q_1\}$	q_0
	q_1	ϕ	q_2
*	q_2	ϕ	ϕ

Transistion table:- (DFA)

	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	q_0
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
* $\{q_0, q_2\}$	$\{q_0, q_1\}$	q_0

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

$= \{q_0, q_1\} \cup \phi$

$= \{q_0, q_1\}.$

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

$= q_0 \cup q_2$

$= \{q_0, q_2\}$

$\rightarrow \delta(\{q_0, q_2\}, 0) = \delta(q_0, 0) \cup \delta(q_2, 0)$

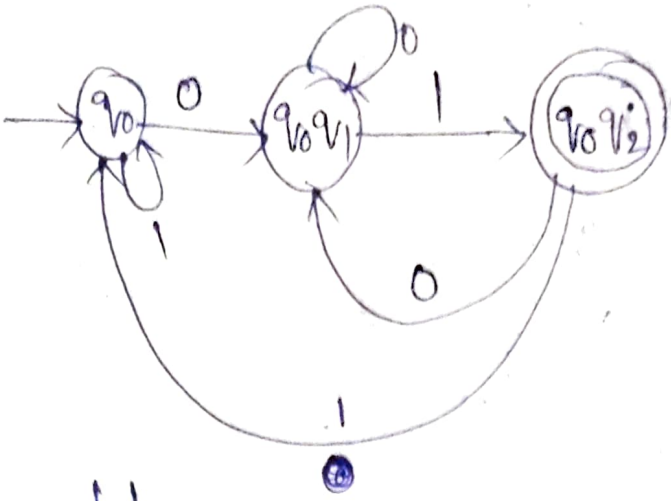
$= \{q_0, q_1\} \cup \phi$

$= \{q_0, q_1\}$

$\rightarrow \delta(\{q_0, q_2\}, 1) = \delta(q_0, 1) \cup \delta(q_2, 1)$

$= q_0 \cup \phi$

$= q_0$



03/09/20

NFA with epilson:-

* having transactions without reading inputs

elimination of ε :-

1. Suppose we want to replace 'ε' move from vertex V_1 to vertex V_2 then

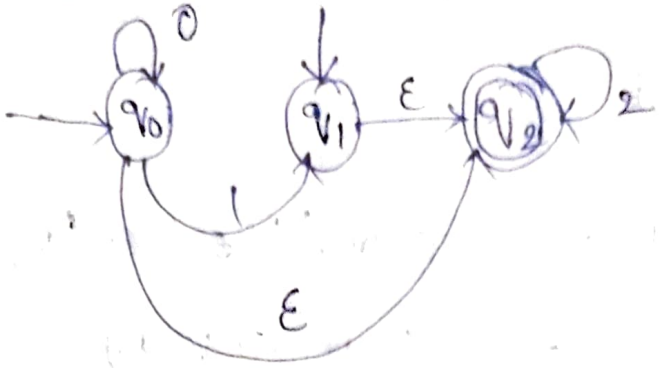
step 1:- Find all the edges strating from V_2

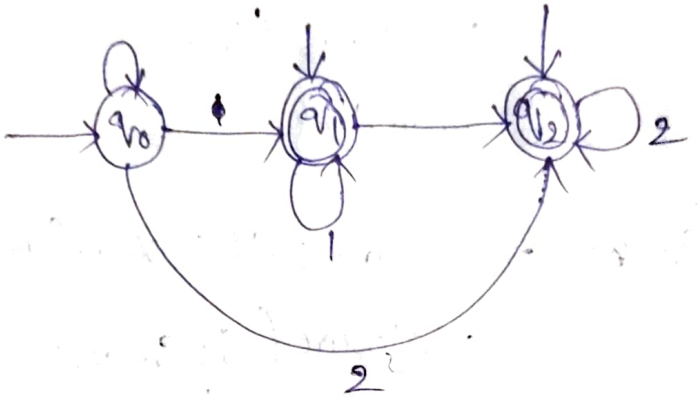
step 2:- Duplicate all these edges starting from V_1 without changing the edge labels.

step 3:- If V_1 is intial state make V_2 also as final state.

step-4:- If V_2 is final state make V_1 also as final state.

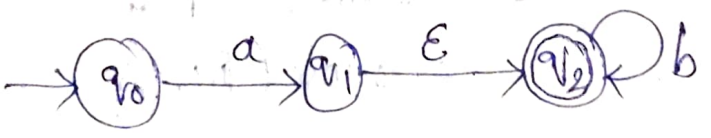
Step-1:-





Elimination of Epsilon:-

(Pb) Construct the following NFA with 'ε' to NFA without 'ε'



Sol:- Finding ε- closure

ε-closure $(q_0) = \{q_0\}$

ε-closure $(q_1) = \{q_1, q_2\}$

ε-closure $(q_2) = \{q_2\}$

Now find δ'

1. $\delta'(q_0, a) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_0), a))$
$= \varepsilon\text{-closure}(\delta(q_0, a))$
$= \varepsilon\text{-closure}(q_1)$
$= \{q_1, q_2\}$.

2. $\delta'(q_0, b) = \varepsilon\text{-clourse}(\delta(\varepsilon\text{-closure}(q_0), b))$
$= \varepsilon\text{-closure}(\delta(q_0, b))$
$= \varepsilon\text{-closure}(\phi)$
$= \phi$

3. $\delta'(q_1, a) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_1), a))$
$= \varepsilon\text{-closure}(\delta(q_1, q_2), a)$
$= \varepsilon\text{-closure}(\delta(q_1, a) \cup \delta(q_2, a))$
$= \varepsilon\text{-closure}(\phi \cup \phi)$
$= \phi$

4. $\delta'(q_1, b) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_1), b))$

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

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

$= \varepsilon\text{-closure}(\phi \cup q_2)$

$= q_2$

5. $\delta'(q_2, a) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_2), a)$.

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

$= \phi$

6. $\delta'(q_2, b) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_2), b)$

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

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

$= \{q_2\}$

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

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

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

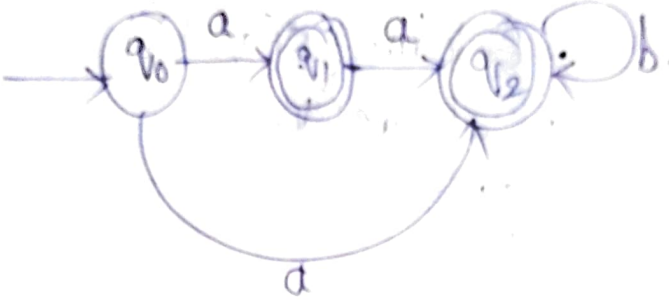
$\delta'(q_1, b) = \{q_2\}$

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

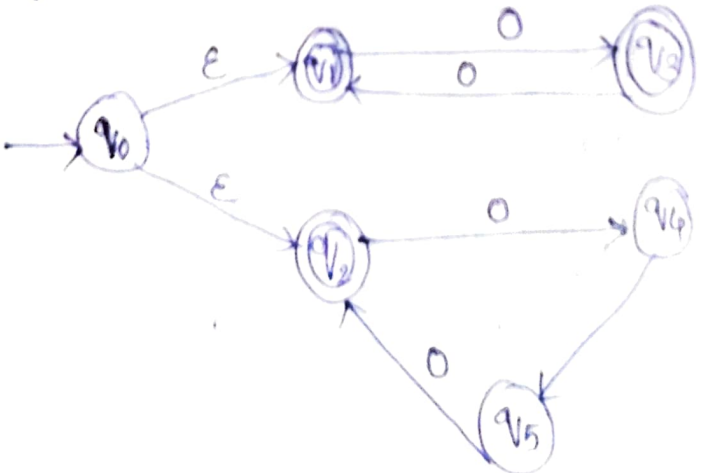
$\delta'(q_2, b) = \{q_2\}$

States	a	b
q_0	$\{q_0, q_1\}$	ϕ
* q_1	ϕ	$\{q_2\}$
* q_2	ϕ	$\{q_2\}$

q_1, q_2 are final states as ε-closures of q_1 & q_2 contain the final state q_2



(2) Construct the NFA without ε for the given NFA with ε:



sol:- Finding ε-closures:-

$\varepsilon\text{-closure}(q_0) = \{q_0, q_1, q_2\}$

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

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

$\varepsilon\text{-closure}(q_3) = \{q_3\}$

$\varepsilon\text{-closure}(q_4) = \{q_4\}$

$\varepsilon\text{-closure}(q_5) = \{q_5\}$.

~~ε-clo~~

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

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

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

$= \varepsilon\text{-closure}(q_3 \cup q_4)$

$= \varepsilon\text{-closure}(q_3) \cup \varepsilon\text{-closure}(q_4)$

$= \{q_3, q_4\}$.

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

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

$= \varepsilon\text{-closure}(q_3)$

$= q_3$.

$$\delta'(q_2, 0) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_2), 0))$$
$$= \varepsilon\text{-closure}(q_4)$$
$$= \{q_4\}$$

$$\delta'(q_3, 0) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_3), 0))$$
$$= \varepsilon\text{-closure}(q_1)$$
$$= \{q_1\}$$

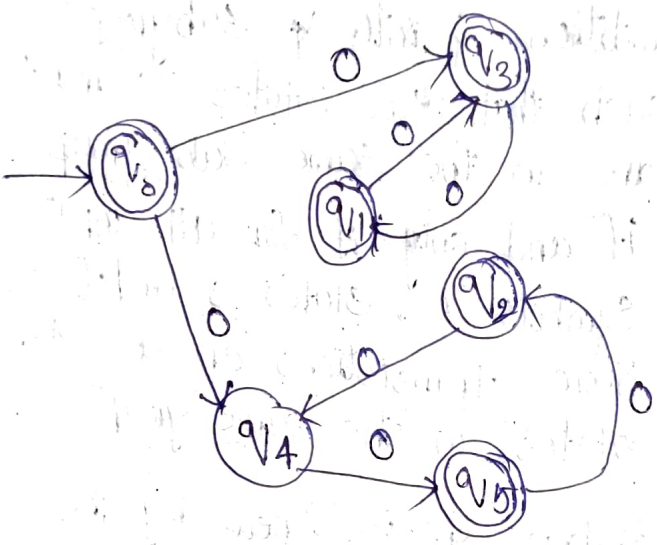
$$\delta'(q_4, 0) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_4), 0))$$
$$= \varepsilon\text{-closure}(q_5)$$
$$= \{q_5\}$$

$$\delta'(q_5, 0) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_5), 0))$$
$$= \varepsilon\text{-closure}(q_2)$$
$$= \{q_2\}.$$

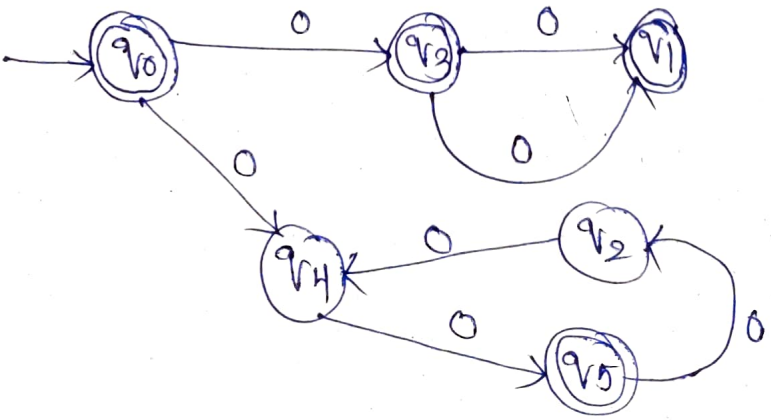
Here the input '1' is not there. So,

$$\delta'(q_0, \phi) = \phi$$
$$\delta'(q_1, \phi) = \phi$$
$$\delta(q_2, 1) = \phi$$
$$\delta(q_3, 1) = \phi$$
$$\delta(q_4, 1) = \phi$$
$$\delta(q_5, 1) = \phi$$

States	0	1
→ * q_0	$\{q_3, q_4\}$	ϕ
* q_1	$\{q_3\}$	ϕ
q_2	$\{q_4\}$	ϕ
* q_3	$\{q_1\}$	ϕ
q_4	$\{q_5\}$	ϕ
* q_5	$\{q_2\}$	ϕ



NFA with out ~~alkpse~~ epsillons.



Minimization of DFA:-

04/09/20

Given DFA:

→ start with at least two groups:
S and S-F.

→ Repeat the following algorithm until no more progress can be made.

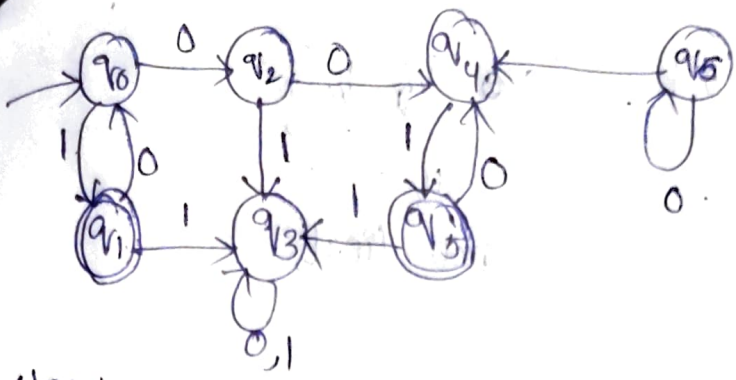
* initially, let $\Pi_{new} = \Pi$

for (each Group G of Π)

{ partition G into subgroups such that two states s and t are in the same sub group if and only if for all input symbols a, states s and t have transitions on a to states in the same group.

Replace G is Π_{new} by the set of all subgroups formed;

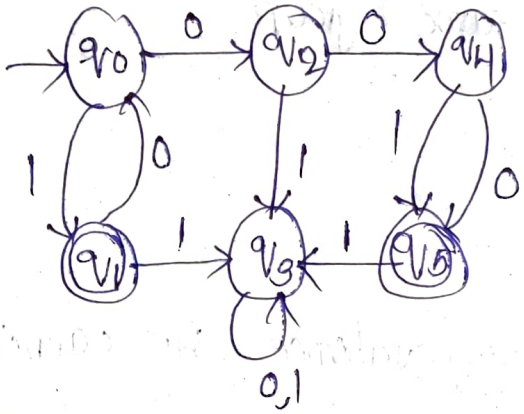
}



Step-1

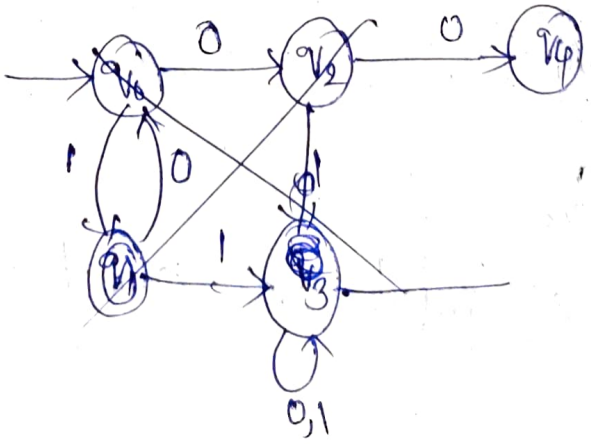
* Here q_6 is unreachable state (As we cannot reach it from source code q_0).

So, we have to remove unreachable state and associated transictions.



Step2:-

* Identify the dead state and remove it and associated transictions.



$\Pi_0 = \{q_1, q_5\}, \{q_0, q_2, q_4\}$

↓ final. ↓ non final

$\Pi_1 = ?$

Find the transistion for above groups for each input

$\delta(q_1, 0) = q_0$

$\delta(q_5, 0) = q_4$

* Based on '0' input we cannot devide as they are in same group.

$\delta(q_1, 1) = \phi$

$\delta(q_5, 1) = \phi$

* Based on '1' equivalence we cannot further divide q_1 & q_5.

$\Rightarrow$ $\delta(q_0, 0) = q_2$

$\delta(q_2, 0) = q_4$

$\delta(q_4, 0) = q_2$

* Based on '0' input we cannot further divide.

$\delta(q_0, 1) = q_1$

$\delta(q_2, 1) = \phi$

$\delta(q_4, 1) = q_5$

$\pi_1 = \{q_1\ q_5\}, \{q_0, q_4\}, \{q_2\}$

$\delta(q_1, 0) = q_0$ | $\delta(q_1, 1) = \phi$

$\delta(q_5, 0) = q_4$ | $\delta(q_5, 1) = \phi$

we cannot divide $\{q_1, q_5\}$ based on 1 and 0 equivalance.

$\delta(q_0, 0) = q_2$

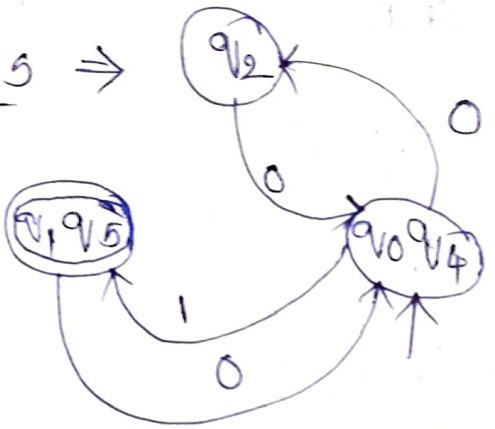
$\delta(q_4, 0) = q_2$

$\delta(q_0, 1) = q_1$

$\delta(q_4, 1) = q_5$

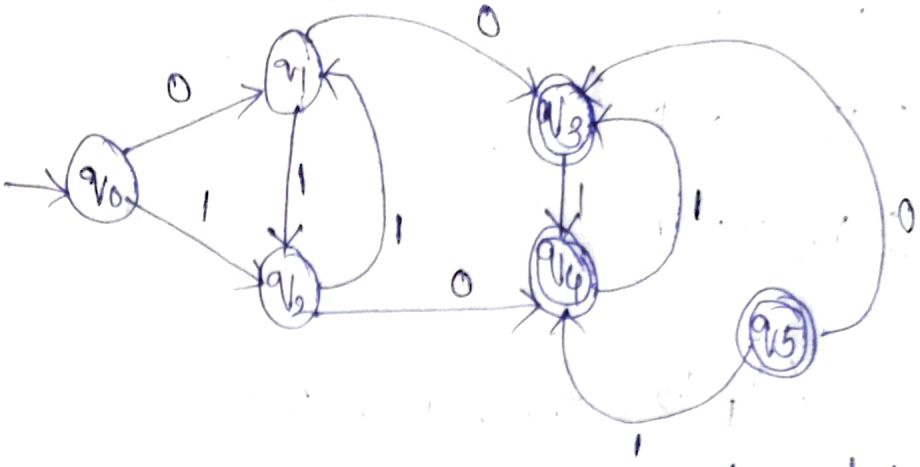
$\pi_2 = \{q_1, q_5\}, \{q_0, q_4\}, \{q_2\}$.

Final states $\Rightarrow$

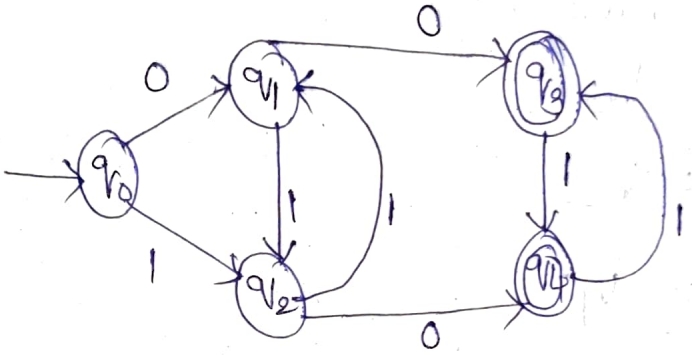


Keep the final states and intial states from the given diagram.

Pb



Sol:- From the given diagram it is clear that 'q_5' is unreachable state. Hence remove it and states associated with it.



$\pi_0 = \{q_0, q_1, q_2\}$ & $\pi_1 = \{q_3, q_4\}$

$\{q_0, q_1, q_2\}$ — non-final

$\{q_3, q_4\}$ — final.

$\delta(q_0, 0) = q_1$.

$\delta(q_1, 0) = q_3$.

$\delta(q_2, 0) = q_4$.

Based on the '0' equivalance we can divide them as.

$\pi_1 = \{q_0\}, \{q_1, q_2\}, \{q_3, q_4\}$.

$\delta(q_1, 0) = q_3$
$\delta(q_2, 0) = q_4$.
} Belongs to same group so we cannot divide them.

$\delta(q_1, 1) = q_2$
$\delta(q_2, 1) = q_1$
} Belongs to same group so we cannot divide them.

$\delta(q_3, 0) = \phi$
$\delta(q_4, 0) = \phi$
} Belongs to same group so we cannot divide them

$\delta(q_3, 1) = q_4$
$\delta(q_4, 1) = q_3$
} Belongs to same group so we cannot divide them.

$\pi_2 = \{q_0\}, \{q_1, q_2\}; \{q_3, q_4\}$

Final diagram:-