Unit - 1

Finite Automata :-

Basic definitions :

Alphabets : It is a finite, non Empty set of symbols (Σ)

Eg $\Sigma = \{a, b\}$ – an alphabet of 2 symbols a & b.

$\Sigma = \{0, 1, 2\}$ - an alphabet of 3 symbols 0, 1, 2

Strings :-

Strings (or) Word over an alphabet set Σ.

It is a finite sequence of symbols from Σ

Eg: abab, abba, are strings over the $\Sigma = \{a, b\}$

a, aa, aaa, are strings over the alphabet $\Sigma = \{a\}$

01101 is string over binary alphabet $\Sigma = \{0, 1\}$

Empty string (or) Null string :-

If string consisting of Zero symbols, The length of a string is 0.

It is denoted by ε or λ

$|\varepsilon| = 0$

OPERATIONS ON STRINGS :-

1) **Length of a String** :

Let W be the string, the length of the string $|W|$, ie, no. of symbols composing the string

Eg
$W = abcd$

$|W| = 4$

$x = 01010101$

$|x| = 8$

$|\varepsilon| = 0$.

2.) Concatenation of a string :-

Eg: 2 strings W and V

appending symbols of V to the right of W

$W = a_1 a_2 a_3 \ldots a_m$

$V = b_1 b_2 b_3 \ldots b_n$

$WV = a_1 a_2 a_3 \ldots a_m \; b_1 b_2 b_3 \ldots b_n$

Eg: x = PAS y = CAL

xy = PASCAL

3.) Reverse of a string : symbols in reverse order

$W = a_1 a_2 \ldots a_m$

$W^R = a_m \ldots a_2 a_1$

$w = 0101011$

$w^r = 1101010$

Powers of Σ :-

Σ^* - denotes the set of all strings over the alphabet Σ

Σ^n - denotes the set of all strings over the alphabet Σ of length n.

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

Σ^1 = Set of all strings over Σ of length exactly 1

ie $\{a, b\}$

$\Sigma^2 = \Sigma \cdot \Sigma \Rightarrow \{a, b\}\{a, b\}$

$\Rightarrow \{aa, ab, ba, bb\}$

$$\Sigma^3 = \Sigma \cdot \Sigma \cdot \Sigma$$
$$= \{aa, ab, ba, bb\} \cdot \{a, b\}$$
$$= \{aaa, aba, baa, bba, aab, abb, bab, bbb\}$$

Σ^0 = set of strings of length '0'.

$\Sigma^0 = \{\varepsilon\} \rightarrow$ Epsilon.

<u>Kleene closure</u> (or) <u>star closure</u> :- set of strings of any length (including null string ε)

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

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

$$\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, aaa \ldots\ldots\}$$

$$\boxed{\Sigma^* = \Sigma^+ \cup \varepsilon}$$

$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \ldots\ldots$ (It consists of all string of any length excluding ε)

$\Sigma^+ = \{a, b, aa, ab, ba \ldots\ldots\}$

$\hookrightarrow$ Positive closure.

<u>Languages</u> :-

A set of strings all of which are chosen from Σ^*

if Σ is an alphabet then L is a language over Σ

$$L \subseteq \Sigma^*$$

language over Σ need not to include strings with all the symbols of Σ.

$\phi \rightarrow$ Empty language (not Even Empty strings)

$\varepsilon \rightarrow$ Null string language $\{\varepsilon\}$ contains only one Empty string.

$$\phi \neq \{\varepsilon\}$$

Language is subset of Σ^* ie $L \subseteq \Sigma^*$

<u>Set formers</u> :- A common way to define a language

$\{W \mid \text{something about } W\}$

above expression reads the set of word W such that whatever is said about W to the right of Vertical Bar

Eg :- $\{w \mid w \text{ consists of equal No. of } 0\text{'s} + 1\text{'s}\}$

$\downarrow$

Expression with parameters.

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

read set 0 to the n, 1 to the n, such that n is greater than or equal to 1.

language consists of

$\{01, 0011, 000111 \cdots\cdots\}$

We can raise a single symbol to a power n to represent 'n' copies of that symbol.

Operations of Languages :-

1) Concatenation of L_1 and L_2

$L_1L_2 = \{ xy \mid x \text{ is in } L_1 , y \text{ is in } L_2 \}$

$L_1 = \{10, 1\}$ $L_2 = \{011, 11\}$

$L_1L_2 = \{10011, 1011, 1011, 111\}$

2) Union :-

$L_1 \cup L_2 = \{ w \mid w \in L_1 \text{ or } w \in L_2 \}$

$L_1 = \{0, 11, 110\}$ $L_2 = \{\varepsilon, 0, 0\}$

$L_1 \cup L_2 = \{\varepsilon, 0, 0, 11, 110\}$

3) Intersection :-

$L_1 \cap L_2 = \{ w \mid w \in L_1 \text{ and } w \in L_2 \}$

$L_1 = \{0, 11, 110\}$ $L_2 = \{\varepsilon, 0, 0\}$

$L_1 \cap L_2 = \{0\}$.

TOC :-

Computation is executing an ~~an~~ algorithm , it involves taking some inputs and performing required operations on it to produce an o/p.

TOC suggests various abstract models of computation represented mathematically.

The computer which performs computations are not actually computers - they are abstract machines.

Abstract M/C

1) Finite Automata

2.) Turing Machine.

Appln of TOC:-

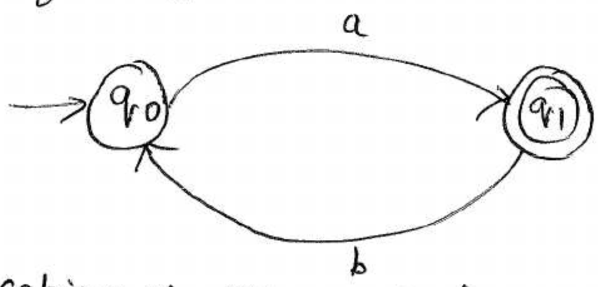
1) compiler Design

2.) Robotics.

3) A.I

Finite Automata (or) Automaton (or) Finite state Machine.

Finite Automata is a Mathematical Model of a System, with discrete inputs and outputs, finite no. of memory configuration called states, and set of transistions from state to state. that occur on i/p symbol from Σ.



Specification of FA:- It is represented by Machine M & specified by 5 tuples.. $M = (Q, \Sigma, \delta, S, F)$

$Q \Rightarrow$ finite, non Empty set of states.

$\Sigma \Rightarrow$ finite, non Empty set of i/p symbols.

$S \rightarrow$ start state or initial state $(S \in Q)$ $\longrightarrow\bigcirc$

$F \rightarrow$ set of final states $F \subseteq Q$ represented by $\circledcirc$

$\delta \rightarrow$ mapping function or transition function.

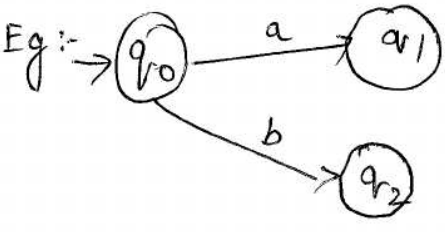
Types of finite Automata :-

DFA (Deterministic finite Automata)

NFA (or) NDFA (non - Deterministic finite Automata).

DFA :- F·A is DFA if there is only one path for a specific input from current state to next state.

Eg :-



State q_0 for i/p 'a', there is only one path going to q_1.

State q_0, for i/p 'b' there is only one path to q_2.

Specification of DFA :-

DFA M is defined by 5 tuples $M = (Q, \Sigma, \delta, S, F)$

$Q \rightarrow$ a finite, non Empty set of states.

$\Sigma \rightarrow$ a finite, non empty set of i/p Alphabets.

$S \rightarrow$ start state $(S \in Q)$

$F \rightarrow$ Set of final states $F \subseteq Q$.

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

Properties of transistion function :-

(i) $\delta = (q, \varepsilon) = q$ system state can be changed by only i/p symbol else remains in original state.

(ii) for i/p symbol 'a' and string w

$$\delta(q, aw) = \delta(\delta(q, a) w)$$

Language Accepted by DFA :-

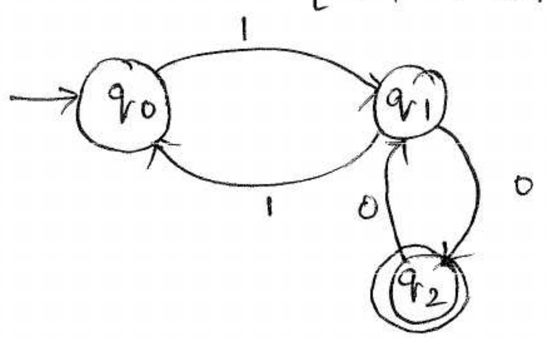
A string 'x' is accepted by DFA $M = (Q, \Sigma, \delta, S, F)$ only if $\delta(q_0, x) = P$ for some P in F

q_0 → Start State, x → String, P → Final state.

Language Accepted by DFA machine M is

$L(G) = \{ x \mid \delta(q_0, x) \in F \}$ where F is a set of final states

① Given $M = \{Q, \Sigma, \delta, q_0, F\}$



Check whether i/p string "1000" is accepted by DFA or not.

by $\delta(q_0, aW) = \delta(\delta(q_0, a)W)$

$\cdot\ \delta(q_0, 1000)$

$\delta(\underline{\delta(q_0, 1)}000)$

$= \delta(q_1, 000)$

$= \delta(\underline{\delta(q_1, 0)}00)$

$= \delta(q_2, 00)$

$= \delta(\underline{\delta(q_2, 0)}0)$

$= \delta(q_1, 0)$

$= \underline{\underline{q_2}}$ → It is a final state.

Hence string is accepted by M.

(2) Describe the language accepted by DFA

$M = (\{q_0, q_1, q_2\}, \{0,1\}, q_0, \{q_1\}, \delta)$ where δ is given by .

q_0 → Start state

$\{q_1\}$ → final state

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

$\delta(q_0, 0) = q_0$ → not accepted .

$\delta(q_0, 01) = \delta(\delta(q_0, 0).1)$

$= \delta(q_0, 1)$

$= q_1$ → [String is accepted by final state q_1].

$\delta(q_0, 011) = \delta(\delta(q_0, 0) 11$

$= \delta(q_0, 11)$

$= \delta(\delta(q_0, 1) 1)$

$= \delta(q_1, 1)$

$= q_2$ (q_2 - non-final state , string not accepted)

$\delta(q_0, 0111) = \delta(\delta(q_0, 0) 111)$

$= \delta(q_0, 111)$

$= \delta(\delta(q_0, 1) 11)$

$= \delta(q_1, 11)$

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

$= \delta(q_2, 1)$

$= q_1$ ⇒ [String 0111 accepted by final state q_1]

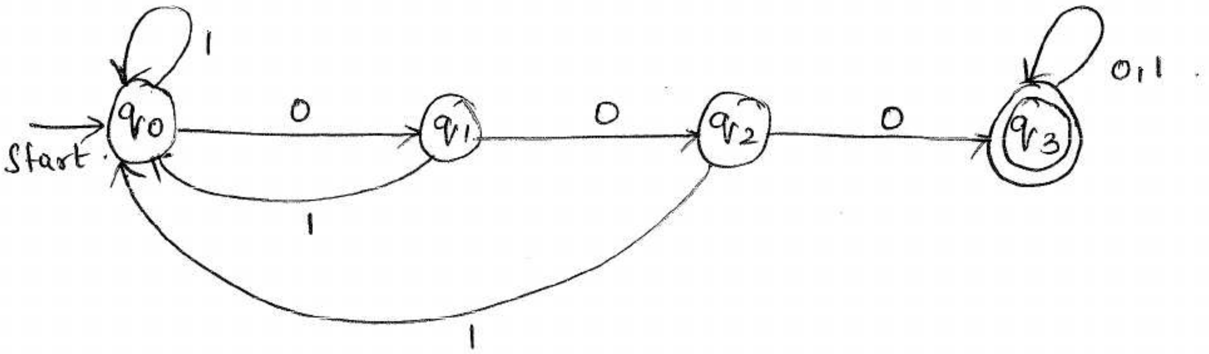
∴ DFA accepts

{01, 0111, 011111, ...} i.e. odd number of 1's at the end of the string.

Design of DFA :-

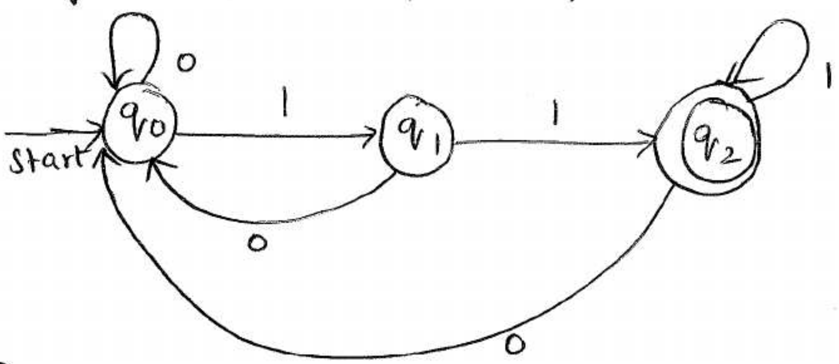
① Design a DFA for the language over $\Sigma = \{0,1\}^*$ such that it contains "000" (3 consecutive 0's)

$L = \{000, 1000, 01000, 1011000 \ldots\ldots\}$



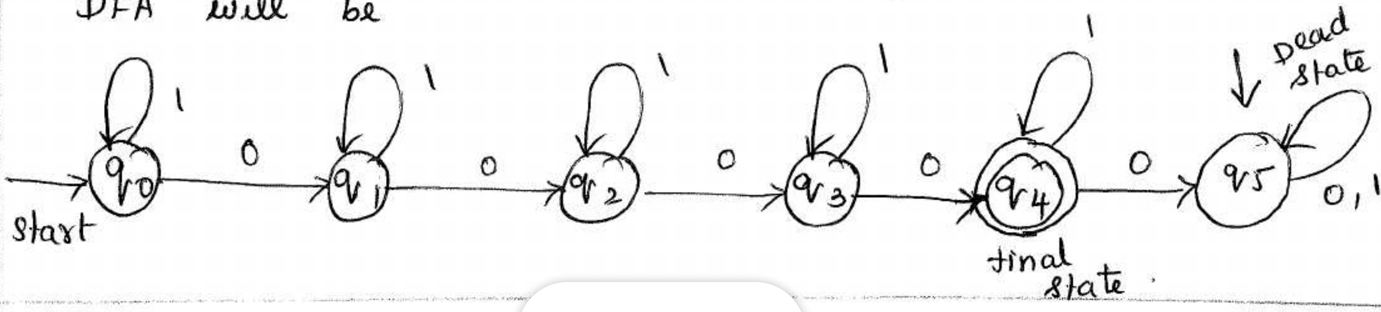
② DFA over {0,1}, that accepts string ends with "11"

Eg $L = \{11, 011, 0011, 1011 \ldots\}$



③ All strings that contain exactly 4 0's.

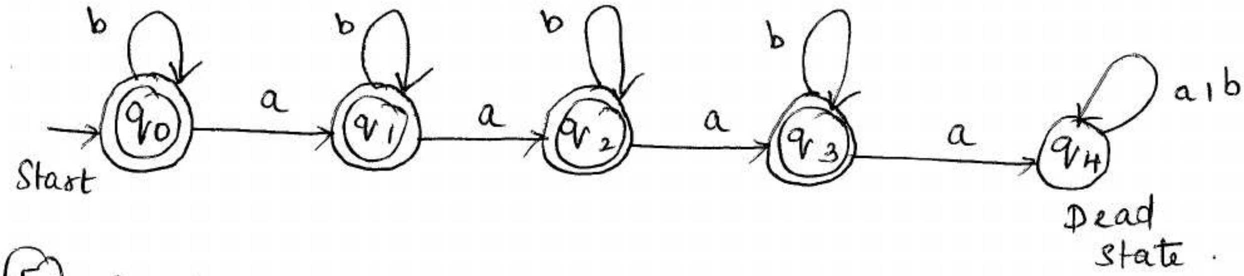
DFA will be



(4.) Construct a DFA over $\Sigma = \{a, b\}$ which produces not more than 3 a's.

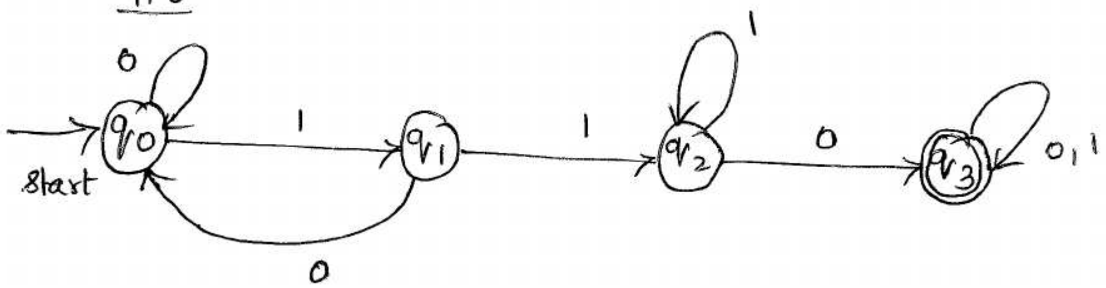
In the given language at most 3 a's are allowed. and there is no restriction on number of b's.

DFA will be

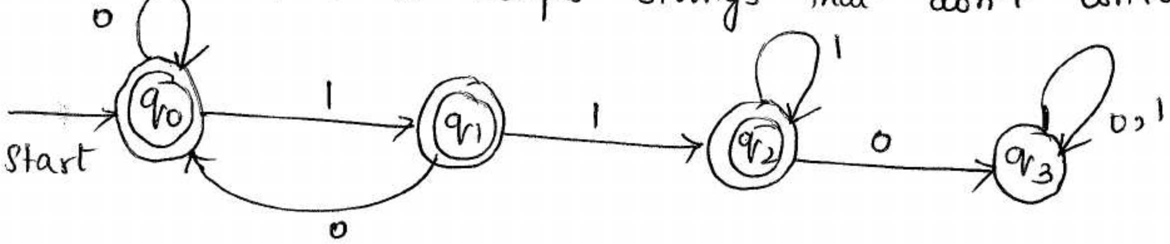


(5) Construct a DFA over Σ {0,1}, that accepts strings that don't contain substring "110"

Step 1 :- Create a DFA that contains the substring "110"

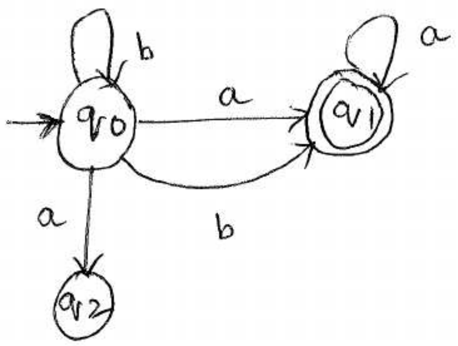


Step 2 :- Change non-final state to final state and final state to non-final state. So that it accepts strings that don't contain "110"



NFA (or) NDFA :-

F.A is NFA when there exists many paths for a specific input from current state to next state.



Finite Automata Accepts a String W if there is a path in the transition Diagram which begins at start state and ends at accepted state.

Transition function :-

(i) $\delta(q, \varepsilon) = q$.

$W = \underset{\substack{\downarrow \\ \text{string}}}{x} \quad \underset{\substack{\downarrow \\ \text{symbol.}}}{a}$

(ii) $\delta(q, \underline{\overset{W}{xa}}) = \delta(\delta(q, x)\, a)$

$= \delta(\{P_1, P_2 \cdots P_n\}, a)$

$= \delta(P_1, a) \cup \delta(P_2, a) \cup \delta(P_n, a)$

$$\delta(q, W) = \bigcup \delta(P_i, a).$$

Language Accepted by NFA :-

$$L(M) = \{ W \mid \delta(q_0, W) \cap F \neq \phi \}$$

Eg:- For the NFA shown, check whether the i/p string "0100" is accepted or not.

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

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

$$\delta(q_0, 01) = \delta(\underline{\delta(q_0, 0)}\, 1)$$
$$= \delta(\{q_0, q_1\}, 1)$$
$$= \delta(q_0, 1) \cup \delta(q_1, 1)$$
$$= \{q_2\} \cup \{q_1, q_2\}$$
$$= \underline{\{q_1, q_2\}}$$

$$\delta(q_0, 010) = \delta(\underline{\delta(q_0, 01)}\, 0)$$
$$= \delta(\{q_1, q_2\}, 0)$$
$$= \delta(q_1, 0) \cup \delta(q_2, 0)$$
$$= \phi \cup \{q_0, q_2\}$$
$$= \underline{\{q_0, q_2\}}$$

$$\delta(q_0, 0100) = \delta(\underline{\delta(q_0, 010)}\, 0)$$
$$= \delta(\{q_0, q_2\}, 0)$$
$$= \delta(q_0, 0) \cup \delta(q_2, 0)$$
$$= \{q_0, q_1\} \cup \{q_0, q_2\}$$
$$= \{q_0, q_1, q_2\}$$

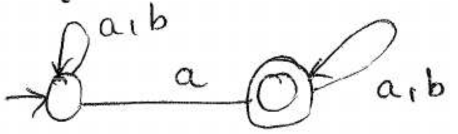
by $\quad \delta(q_0, w) \cap F \neq \phi$

$$\delta(q_0, 0100) \cap F$$
$$\{q_0, q_1, q_2\} \cap \{q_0\}$$
$$\{q_0\} \neq \phi$$

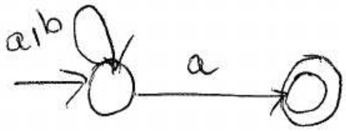
$\hookrightarrow$ final state, the string is accepted by NFA.

Design NFA :-

L_1 = {contains 'a'}

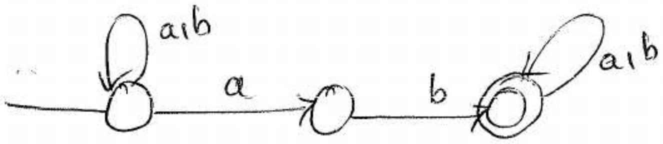


L_2 = {ends with 'a'}

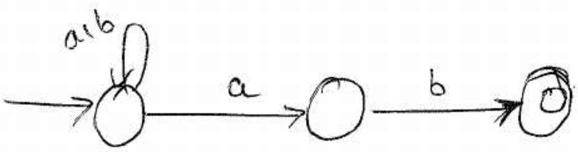


L_3 = {starts with 'ab'}

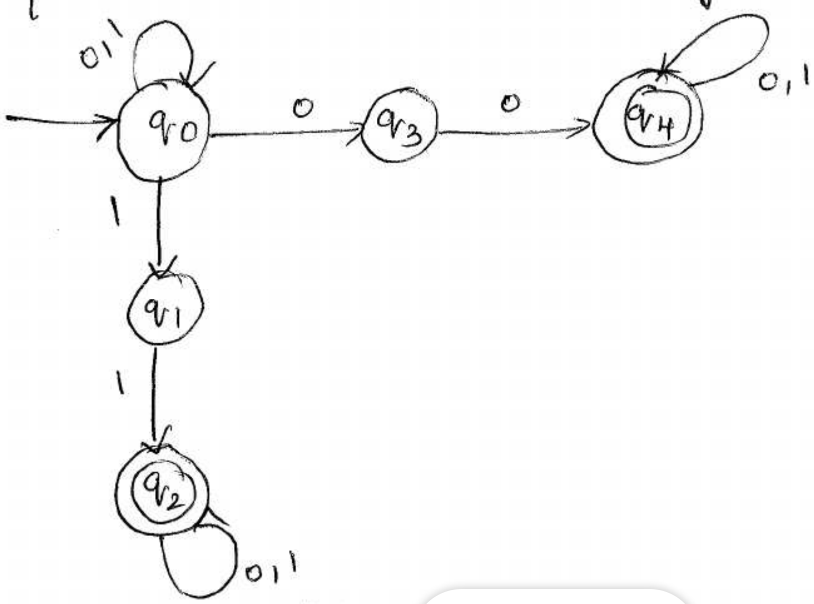
L_4 = {contains 'ab'} or {substring 'ab'}



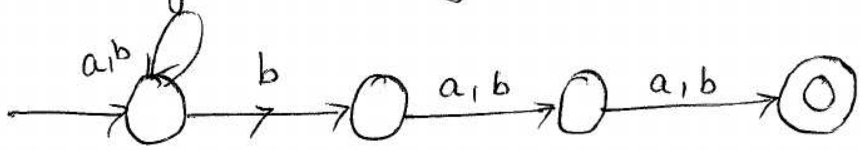
L_5 = {ends with 'ab'}



L_6 = {contains 00 or 11 as substring}



Design a NFA for $L = \{W \mid W \in (a,b)^*$ and third symbol from right(W) is b$\}$.



Finite Automata with ε-moves :-

It is extended form of NFA by introducing a ε-moves that allows us to make transition on Empty string. ε-transitions are used simply to change one state to other. Sometimes to reach to final state we do not get proper state from start state. In such a case we simply want to reach to certain state which leads to final state. Such a transition to that specific state should be without any input symbol. Hence we require some ε-moves by which proper state can be obtained for reaching to final state.

Acceptance of language :-

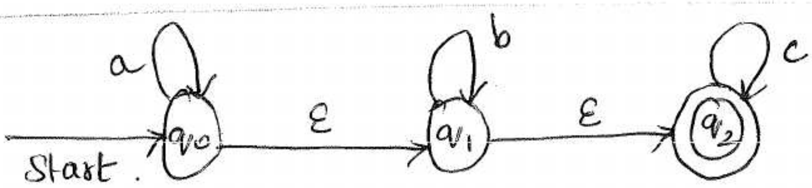
The Language L accepted by NFA-ε denoted by

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

where Σ = input $\cup \{\varepsilon\}$

$$\delta : Q \times \Sigma \cup \{\varepsilon\} \rightarrow 2^{Q}.$$

NFA with ε which accepts a language consisting the strings of any no. of a's followed by any no of b's followed by any no. of c's.



	a	b	c	ε
q_0	q_0	ϕ	ϕ	q_1
q_1	ϕ	q_1	ϕ	q_2
q_2	ϕ	ϕ	q_2	ϕ

Parse the String aabbcc as follows

$$\delta(\underline{q_0, a}abbcc) \vdash \delta(q_0, abbcc)$$
$$\vdash \delta(q_0, bbcc)$$
$$\vdash \delta(q_0, \varepsilon bbcc)$$
$$\vdash \delta(\underline{q_1, b}bcc)$$
$$\vdash \delta(\underline{q_1, b}cc)$$
$$\vdash \delta(q_1, cc)$$
$$\vdash \delta(q_1, \varepsilon cc)$$
$$\vdash \delta(q_2, cc)$$
$$\vdash \delta(q_2, c)$$
$$\vdash \delta(q_2, \varepsilon)$$

Thus we reach to accept State, after Scanning the Complete String

Definition of ε-closure :-

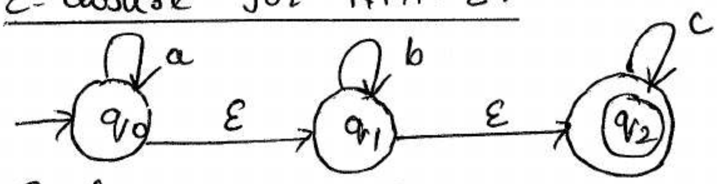
The ε-closure (P) is a set of all states which are reachable from state P on ε transitions such that

(i) ε-closure (P) = P where $P \in Q$.

(ii) If there Exists ε-closure (P) = $\{q\}$ and $\delta(q, \varepsilon) = r$ then ε-closure (P) = $\{q, r\}$

ε- closure for NFA$-\varepsilon$.

①

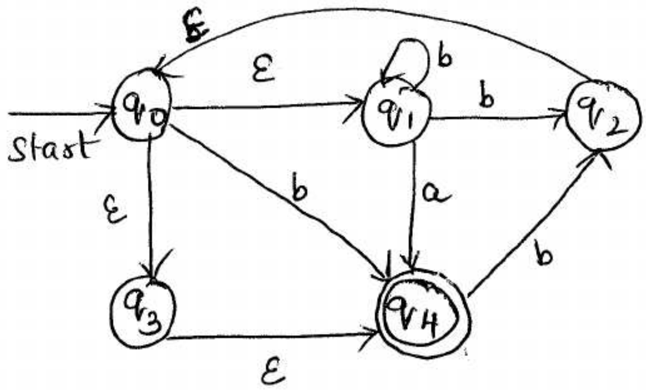


ε- closure $(q_0) = \{q_0, q_1, q_2\}$ means self state + ε- reachable states.

ε- closure $(q_1) = \{q_1, q_2\}$ means q_1 is a self state and q_2 is a state obtained from q_1 with ε input.

ε- closure $(q_2) = \{q_2\}$.

②



ε closure $(q_0) = \{q_0, q_1, q_3, q_4\}$

ε closure $(q_1) = \{q_1\}$

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

ε-closure$(q_3) = \{q_3, q_4\}$

ε-closure$(q_4) = \{q_4\}$

Transition function :-

(i) $\delta'(q_0, \varepsilon) = \varepsilon$- closure (q_0)

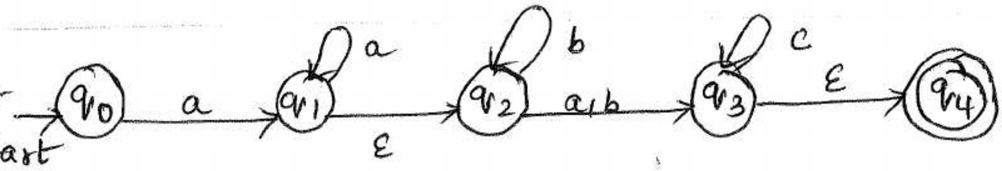
(ii) $\delta'(q_0, a) = \varepsilon$- closure $\delta(\delta'(q_0, \varepsilon), a)$

(iii) $\delta'(q_0, Wa) = \varepsilon$- closure $\delta(\delta'(q_0, W), a)$

Language accepted by NFA-ε :-

$L(M) = \{W \mid W \in \Sigma^*$ and δ transition for W from S reaches to $F\}$

Eg:-



update ε-closure of each state

$\varepsilon\text{-}cl(q_0) = \{q_0\}$ $\qquad$ $\varepsilon\text{-}cl(q_3) = \{q_3, q_4\}$

$(q_1) = \{q_1, q_2\}$ $\qquad$ $\varepsilon\text{-}cl(q_4) = \{q_4\}$

$(q_2) = \{q_2\}$

Find $\delta'(q_0, aba)$

$\delta'(q_0, a) = \varepsilon\text{-closure } \delta(\delta'(q_0, \varepsilon), a)$ [By transition fun (ii)]

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

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

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

$\delta'(q_0, ab) = \varepsilon\text{-closure } \delta(\underline{\delta'(q_0, a)}\ b)$

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

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

$= \varepsilon\text{-closure } (\phi) \cup (q_2, q_3)$

$= \varepsilon\text{-closure } (q_2, q_3)$

$= \varepsilon\text{-}cl(q_2) \cup \varepsilon\text{-}cl(q_3)$

$= \{q_2\} \cup \{q_3, q_4\} \Rightarrow \underline{\underline{\{q_2, q_3, q_4\}}}$

$\delta'(q_0, aba) = \varepsilon\text{ closure } \delta(\underline{\delta'(q_0, ab)}\ a)$

$= \varepsilon\text{ closure } \delta(\{q_2, q_3, q_4\}, a)$

$= \varepsilon\text{ closure } \delta(q_2, a) \cup \delta(q_3, a) \cup \delta(q_4, a)$

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

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

$= \underline{\underline{\{q_3, q_4\}}}$ Set has final state or final element, so Given String is accepted.

Equivalence of NFA and DFA:

Theorem:- Let L be a set accepted by NFA then then there exists a DFA that accept L.

Proof:-

Every DFA is a NFA. DFA can simulate NFA i.e., for every NFA we can construct an equivalent DFA.

All states in NFA are also in DFA. Language accepted by NFA also accepted by DFA.

Let $M = (Q, \Sigma, \delta, q_0, F)$ be NFA accepting L. Then define a DFA as $M' = (Q', \Sigma, \delta', q_0', F')$ where

$Q' = 2^Q$

$q_0' = [q_0]$

F' is the set of all subsets of Q containing an element of F $\delta'([q_1, q_2, \ldots q_i], a) = \delta(q_1, a) \cup \delta(q_2, a) \cup \ldots \delta(q_i, a)$

Equivalently

$$\delta'(\{q_1, q_2, \ldots q_i\}, a) = \{p_1, p_2 \ldots p_j\} \text{ iff}$$

$$\delta([q_1, q_2 \ldots q_i], a) = [p_1, p_2 \ldots p_j]$$

Before proving $L(M) = L(M')$

$$\delta'(q_0', x) = [q_1, q_2 \ldots q_i]$$

if and if only

$$\delta(q_0, x) = \{q_1, q_2, \ldots q_i\}$$

We prove by induction on $|x|$

when $|x| = 0$

$\delta(q_0, x) = \{q_0\}$ similarly $\delta'(q_0', x) = \{q_0\}$

Assume above true for all strings with $|y| \leq M$.

Let x be string of length $M+1$, we can write x as 'ya'. $|y| = M$ and $a \in$

Let

$\delta(q_0, y) = \{P_1, P_2 \ldots P_j\}$ and $\delta(q_0, ya) = \{r_1 \ldots r_j\}$

$\delta'(q_0', y') = [P_1, P_2 \ldots P_j]$ and

$$\delta'(q_0', ya) = \delta'(\delta'(q_0', y), a)$$
$$= \delta'([P_1, P_2 \ldots P_j], a)$$
$$= [r_1, r_2, \ldots r_j]$$

Now $x \in L(M)$ iff $\delta(q_0, x)$ contains a state of 'F' if and only if $\delta'(q_0', x)$ Contains a state of 'F''

Hence $x \in L(M')$.

$\therefore L(M) = L(M')$.

Theorem 2 :- Equivalence of NFA - ε to NFA.

If L is accepted by NFA with ε, then there exists L which is accepted by NFA without ε.

Proof :-

Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA - ε

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

$$F' = \begin{cases} F & \text{if } \varepsilon\text{closure}(q_0) \text{ doesn't contain } F \\ F \cup \{q_0\} & \text{if } \varepsilon\text{closure}(q_0) \text{ contains } F \end{cases}$$

Basis :- $|x| = 1$, Then x is a symbol a

$$\overset{NFA-\varepsilon}{\delta'(q_0, a)} = \overset{NFA}{\delta''(q_0, a)}$$

But this statement is not true for ε.

i.e., $\delta'(q_0, \varepsilon) = \{q_0\}$ $\delta''(q_0, \varepsilon) = \varepsilon\text{-closure}(q_0)$

Induction :- $|x| > 1$ Let $x = wa$

$$\delta'(q_0, w) = \delta''(q_0, w) = P.$$

show that $\delta'(P, a) = \delta'(q_0, wa)$

But $\delta'(P, a) = \bigcup\limits_{q \in P} \delta'(q, a) = \bigcup\limits_{q \in P} \delta''(q, a)$

As $P = \delta''(q_0, w)$

We have $\bigcup\limits_{q \in P} \delta''(q, a) = \delta''(q_0, wa)$

Thus by definition δ''

$$\boxed{\delta'(q_0, wa) = \delta''(q_0, wa)}$$

3 0

Eg Construct a DFA for the given NFA
$M = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$ where
δ is given by

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

Solns:-

DFA $M' = \{Q', \delta', \{a, b\}, q_0, F'\}$

$Q' = \{\phi, [q_0], [q_1], [q_2], [q_0, q_1] [q_0, q_2] \ldots\}$

$\delta'([q_0], 0) = [q_0, q_1] \rightarrow$ new state $(\because \delta(q_0, 0) = \{q_0, q_1\})$

DFA NFA

$\delta'([q_0], 1) = [q_2]$ $(\because \delta(q_0, 1) = \{q_2\}$

$$\begin{aligned}\delta'([q_0, q_1], 0) &= \delta(\{q_0, q_1\}, 0) \\ &= \delta(q_0, 0) \cup (q_1, 0) \\ &= \{q_0, q_1\} \cup \{q_0\} \\ &= [q_0, q_1]\end{aligned}$$

$$\begin{aligned}\delta'([q_0, q_1], 1) &= \delta(\{q_0, q_1\}, 1) \\ &= \delta(q_0, 1) \cup \delta(q_1, 1) \\ &= \{q_2\} \cup \{q_1\} \\ &= [q_1, q_2] \rightarrow \text{new state.}\end{aligned}$$

3 0

$\delta'([q_2], 0) = \phi \Rightarrow$ No state getting generated.

$\delta'([q_2], 1) = [q_0, q_1]$ $\quad (\because \delta(q_2, 1) = \{q_0, q_1\})$.

$$\begin{aligned}
\delta'([q_1, q_2], 0) &= \delta(\{q_1, q_2\}, 0) \\
&= \delta(q_1, 0) \cup \delta(q_2, 0) \\
&= \{q_0\} \cup \phi \\
&= [q_0]
\end{aligned}$$

$$\begin{aligned}
\delta'([q_1, q_2], 1) &= \delta(\{q_1, q_2\}, 1) \\
&= \delta(q_1, 1) \cup \delta(q_2, 1) \\
&= \{q_1\} \cup \{q_0, q_1\} \\
&= \{q_0, q_1\} \\
\text{ie} \quad & [q_0, q_1]
\end{aligned}$$

As now no new states are generating, The transition table for DFA using above δ' is

States	0	1
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$*[q_2]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_1, q_2]$
$*[q_1, q_2]$	$[q_0]$	$[q_0, q_1]$

Assignment :-

1) construct DFA equivalent to the given NFA.

	0	1
$\rightarrow p$	$\{p, q\}$	p
q	r	r
r	s	ϕ
$*s$	s	s

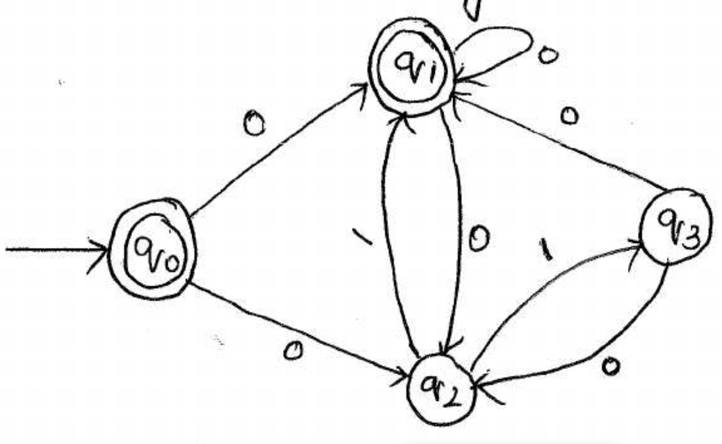
2) Convert the given NFA to DFA.

	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_1\}$
q_2	$\{q_3\}$	$\{q_3\}$
$*q_3$	$\{\phi\}$	$\{q_2\}$

3) convert the given NFA to DFA.

State	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_1\}$
$*q_1$	$\{\phi\}$	$\{q_0, q_1\}$.

4) Convert the following NFA into DFA



* First design transition table from transition Diagram.

i) Construct DFA equivalent to the NFA.

$M = (\{p, q, r\}, \{0, 1\}, \delta, P, \{q, s\})$ where δ is defined in the following table.

δ	0	1
$\rightarrow p$	$\{q, s\}$	$\{q\}$
q	$\{r\}$	$\{q, r\}$
r	$\{s\}$	$\{p\}$
s	–	$\{p\}$.

Solution:- To construct DFA.

$\delta\{p, 0\} = \{q, s\}$ Since we have new state we have to

$\delta\{p, 1\} = \{q\}$ find transistions.

$\delta\{q, 0\} = \{r\}$

$\delta\{q, 1\} = \{q, r\}$ new state.

$\delta\{r, 0\} = \{s\}$

$\delta\{r, 1\} = \{p\}$

$\delta\{s, 0\} = -$

$\delta\{s, 1\} = \{p\}$

$\delta\{\{q, s\}, 0\} = \{r\}$

$\delta\{\{q, s,\}, 1\} = \{p, q, r\}$ — new state.

$\delta\{\{q, r\}, 0\} = \{r, s\}$ — new state

$\delta\{\{q, r\}, 1\} = \{p, q, r\}$

$\delta\{\{p, q, r\}, 0\} = \{q, r, s\}$ – new state.

$\delta\{\{p, q, r\}, 1\} = \{p, q, r\}$

$\delta\{\{r, s\}, 0\} = \{s\}$

$\delta\{\{r, s\}, 1\} = \{p\}$

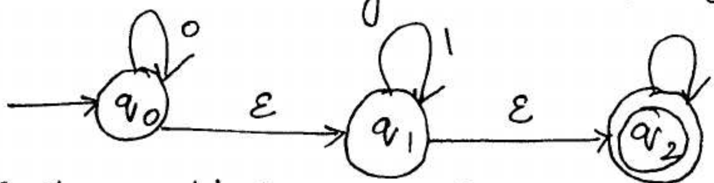
$\delta\{\{q, r, s\}, 0\} = \{r, s\}$

$\delta\{\{q, r, s\}, 1\} = \{p, q, r\}$.

The transition table :-

δ	0	1
$\rightarrow p$	$\{q, s\}$	$\{q\}$
q	$\{r\}$	$\{q, r\}$
r	$\{s\}$	$\{p\}$
s	–	$\{p\}$
$\{q, s\}$	$\{r\}$	$\{p, q, r\}$
$\{q, r\}$	$\{r, s\}$	$\{p, q, r\}$
$\{p, q, r\}$	$\{q, r, s\}$	$\{p, q, r\}$
$\{r, s\}$	$\{s\}$	$\{p\}$
$\{q, r, s\}$	$\{r, s\}$	$\{p, q, r\}$

(ii) Convert the given NFA - ε to NFA without ε.



Step-1 :- obtain ε-closure of each state.

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

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

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

Step-2 :- obtain δ' transitions for each state on each i/p symbol.

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

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

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

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

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

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

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

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

$= \varepsilon\text{-cl}\,(\phi) = \phi$

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

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

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

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

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

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

$= \{q_1,q_2\}$

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

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

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

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

$= \phi$

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

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

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

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

$= \phi$

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

$= \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_0), 2))$

$= \varepsilon\text{-closure}(\delta(q_0, q_1, q_2), 2)$

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

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

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

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

$= \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q_1), 2))$

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

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

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

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

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

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

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

$= \{q_2\}$

From this Write the transition table as.

State \ i/p	0	1	2
q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	ϕ	$\{q_1, q_2\}$	$\{q_2\}$
q_2	ϕ	ϕ	$\{q_2\}$