24/01/23

UNIT - 1

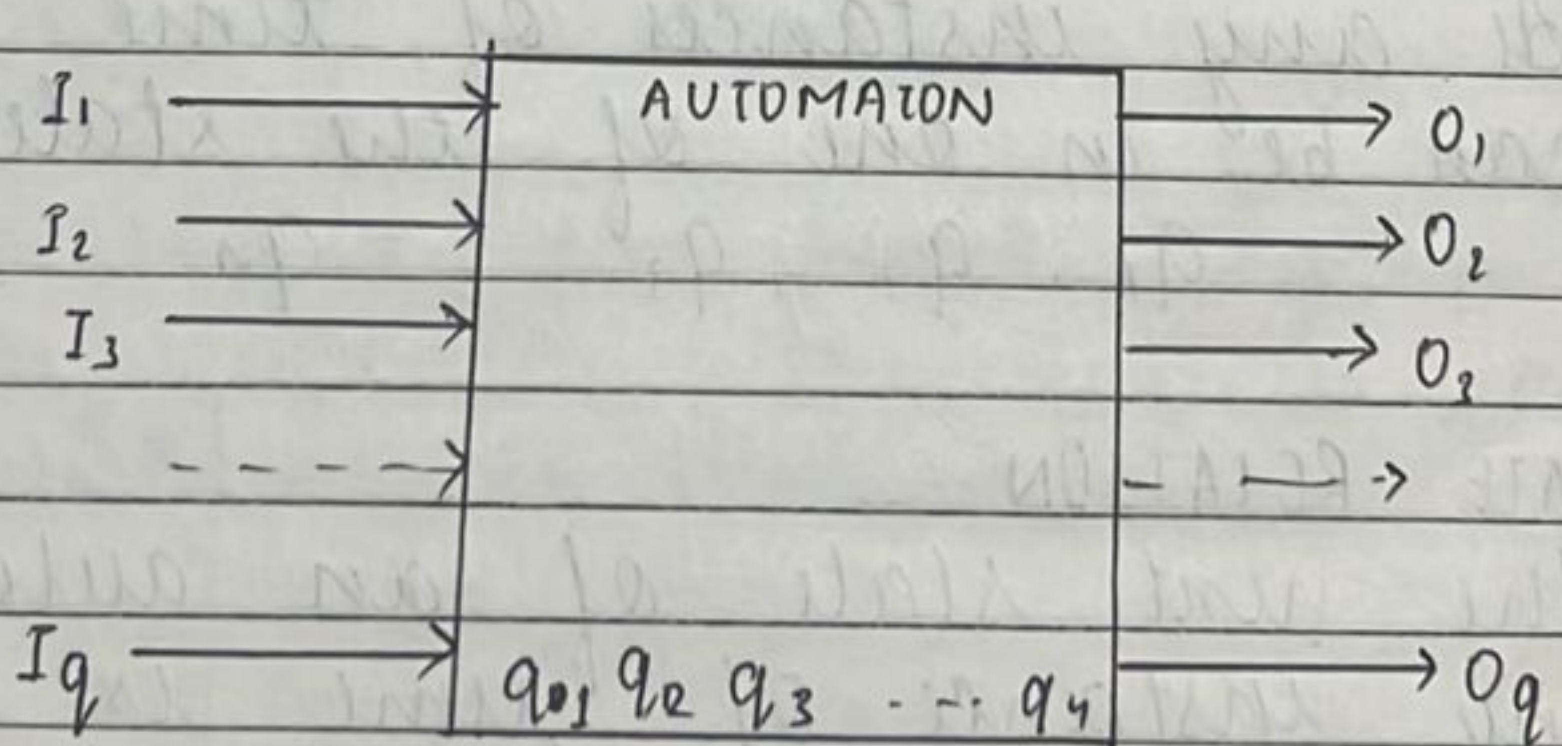
THE THEORY OF AUTOMATA

DEFINITION OF AUTOMATON

An automaton is defined as the system where energy, materials & information are transformed, transmitted and used for performing some functions without direct participation of man

Ex. - Automatic machine tools
Automatic packaging machine
Automatic photo printing machine

In computer science the term automaton means discrete automaton & is defined in a more abstract way as shown in figure



MODEL OF DISCRETE AUTOMATON

The characteristics of discrete automaton are now described :

1. INPUT

At each of the discrete instance of time $t_1, t_2, t_3 \ldots t_n$, the input values $I_1, I_2, I_3 \ldots I_p$ each of which can take a finite Numbers of fixed values from the input alphabets.

$t_1 \quad t_2 \quad t_3 \ldots t_n$
$I_1 \quad I_2 \quad I_3 \ldots I_n$

2. OUTPUT

$O_1, O_2, O_3 \ldots O_q$ are the outputs of the model each ~~of~~ ~~sar~~ of which can take a finite number of fixed

3. STATES

At any instances of time the automaton can be in one of the states,

$q_1, q_2, q_3 \ldots q_n$

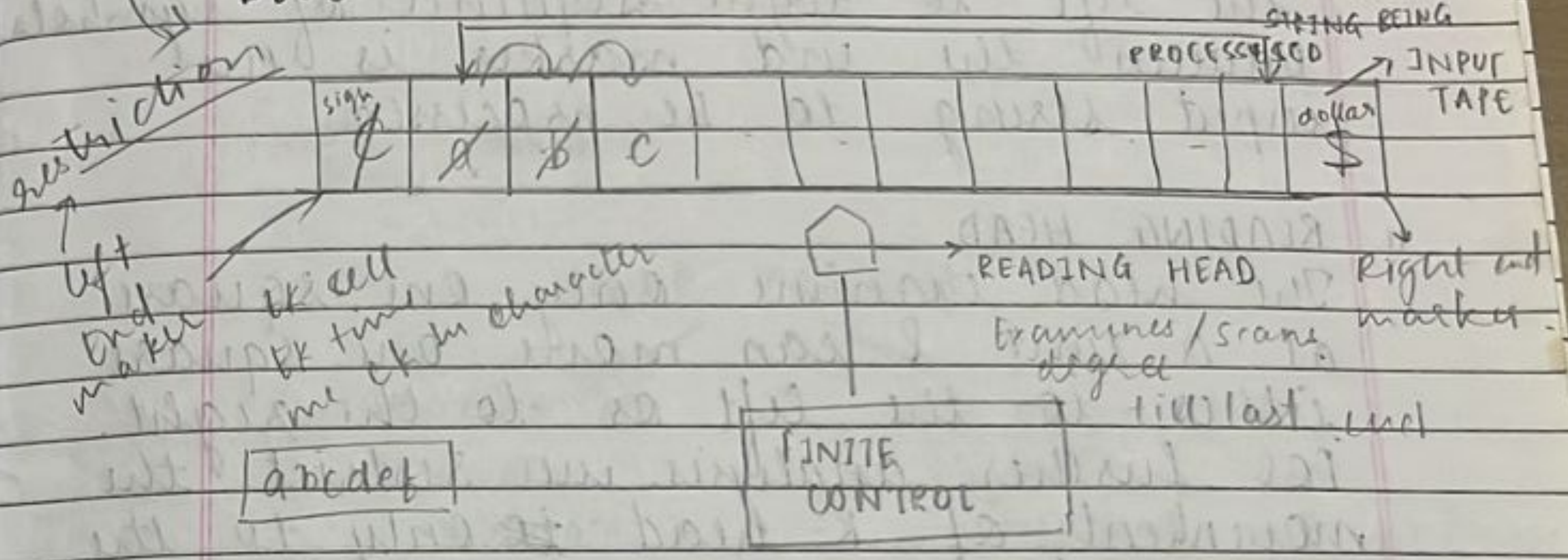
4. STATE RELATION

The next state of an automaton at any instance of time is determine by the present state and present input.

OUTPUT RELATION

The output is related to either state only or to both the input and the state

BLOCK DIAGRAM OF A FINITE AUTOMATON



BLOCK DIAGRAM OF ~~A~~ A FINITE AUTOMATA

Input tape → ek cell → ek alphabet
ek time

Read karna

Figure shows the block diagram of finite automata. The various component is shown as follow.

1. INPUT TAPE

The input tape is divided into squares each square containing a single symbol from the input alphabet ~~is~~ Σ.

The n square of ~~constant~~ ~~xxx~~ detail contain ¢ sign at the left end and

$ at the right end.
Absence of $ markers indicates that the tape is of infinite length.
The left to right sequence of symbols between the end marker is the input string to be processed

ii READING HEAD
The head examine only one square at a time & can move one square either to the left or to the right. For further analysis we restrict the movement of R-head ~~to~~ only to the right side.

iii FINITE CONTROL
The input to the finite control will be usually : symbol under the R-head say a or the present state of the machine say q to give the following output.

(A) · A motion of R-head along the tape to the next square (In same a Null move i.e R-head remaining the same square is permitted).

(B) : The next of the finite state machine given by $\delta(q, a)$ — CURRENT I/P, PRESENT

$\rightarrow q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2$

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

FINITE AUTOMATA

FINITE AUTOMATA
- DETERMINISTIC FINITE AUTOMATA (DFA)
- NON-DETERMINISTIC FINITE AUTOMATA (NDFA or NFA)

DETERMINISTIC
It is a mathematical model that has finite number of state which discrete input ~~or~~ & outputs.

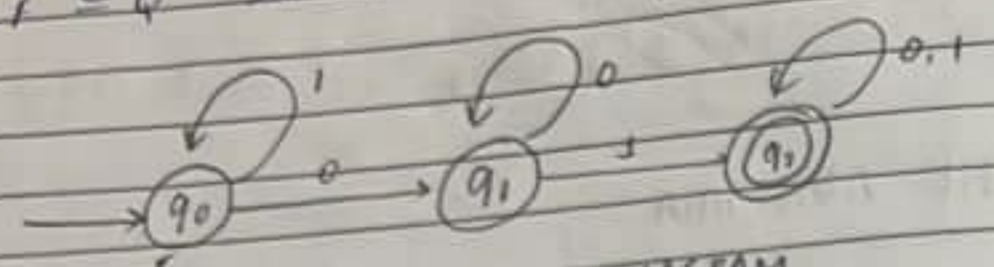
FORMAL DEFINITION OF FINITE AUTOMATA :
Finite automata can be represented by a five tuple

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

where
Q = finite ~~state of~~ set of state
Σ = finite set of input alphabet
δ = Transition function which maps
$$\delta : Q \times \Sigma \rightarrow Q$$

$F \subseteq Q$ subset

$q_0 \rightarrow$ Initial state or start symbols
$F \subseteq Q$ is the set of final state



INITIAL STATE   TRANSITION DIAGRAM

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

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

TRANSITION FUNCTION
$\delta(q_0, 0) = q_1$
$\delta(q_0, 1) = q_0$
$\delta(q_1, 0) = q_1$
$\delta(q_1, 1) = q_2$
$\delta(q_2, 0) = q_2$
$\delta(q_2, 1) = q_2$

Transition table is used to represent the transition function

STATE	INPUT ALPHABET 0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
q_2	q_2	q_2

$\rightarrow q_0$ - By default deterministic
Jab final me nahi hota toh aap woh string accept nahi karega

Applications: $F = \{q_2\}$
object recognisation

ACCEPTABILITY OF A STRING BY A FINITE AUTOMATA

pattern matching, face recognition

A string ~~x~~ x is accepted by the finite automata after reading x if it reaches the final state

$L(M) = \{x / \delta(q_0, x)$ is in $F\}$

language deterministic finite automata

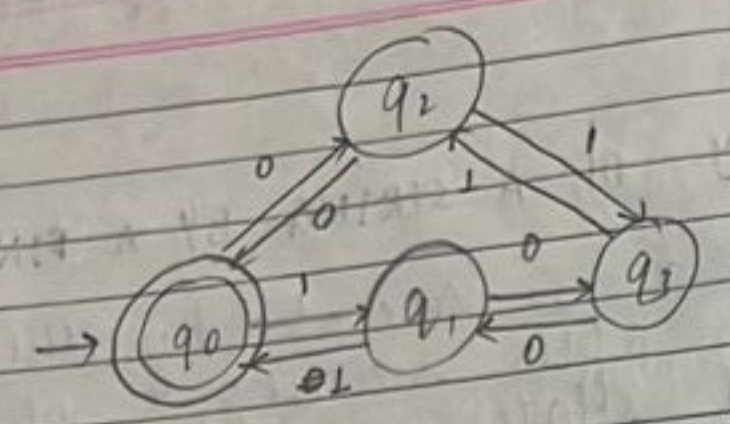
$$\boxed{L(M) = \{x / \delta(q_0, x) \in F\}}$$

CENTRAL / MAIN THEOREM

Consider the finite state machine where transition function δ is given in the form of transition table, here
$Q = \{q_0, q_1, q_2, q_3\}$
$\Sigma = \{0, 1\}$
$F = \{q_0\}$

Give the entire sequence of states for the input string 110101 & transition table

STATE	INPUT ALPHABET 0	1
$\rightarrow q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2



$\cdot L(M) = \{ x / \delta(q_0, x) \in F \}$

$\delta(q_0, \cancel{1}10101) = \delta(q_1, \cancel{1}0101)$

$= \delta(q_0, \cancel{0}101)$

$= \delta(q_2, \cancel{1}01)$

$= \delta(q_3, \cancel{0}1)$

$= \delta(q_1, \cancel{1})$

$= \delta(q_0, \epsilon) = q_0 \in F$

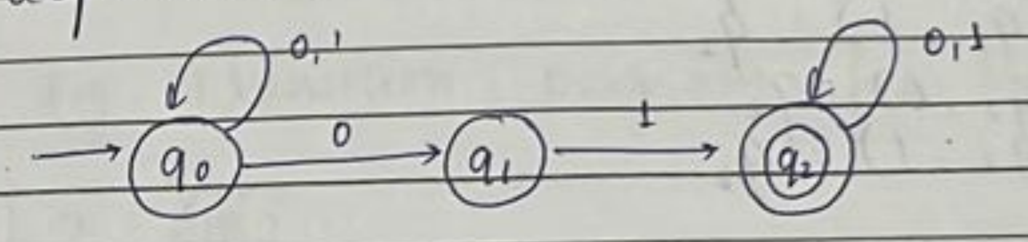
After reading $x = 110101$ it reaches to the state q_0 and this q_0 belongs to the final state so we can say that ~~x~~ input string $x = 110101$ is excepted by finite automata.

$q_0 \xrightarrow{1} q_1 \xrightarrow{\cancel{0}1} q_0 \xrightarrow{0} q_2 \xrightarrow{1} q_3 \xrightarrow{0} q_1 \xrightarrow{1} q_0$

↓ the symbol indicates represents current input symbol being processed by the machine

NON- DETERMINISTIC FINITE AUTOMATA.

- It is known as NDFA or NFA
- The NFA allow more than one transition from same state on the same input alphabets.



$\delta(q_0, 0) = q_{0,1}$
$\delta(q_0, 0) = q_0$

FORMAL DEFINITION OF NDFA

~~A~~ NDFA or NFA is a 5- Tuple

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

where

Q = Finite set of state

Σ = Finite set of input alphabets

δ = transition function which maps

~~q_0~~ ~~Initi~~ ~~$\delta(Q \times$~~

$$\boxed{\delta : Q \times \Sigma \Rightarrow 2^Q}$$

q_0 = Initial state or start symbol

$F \subseteq Q$ ~~set~~ set of final state.

δ

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

$\Sigma = \{$ set of symbols $\}$

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

$\delta(q_0, 0) = (q_0, q_1)$
$\delta(q_0, 1) = q_0$
$\delta(q_1, 0) = \phi$
$\delta(q_1, 1) = q_2$
$\delta(q_2, 0) = q_2$
$\delta(q_2, 1) = q_2$

$q_0 \rightarrow$ Initial state
$F = \{q_2\}$

STATE	0	1
$\rightarrow q_0$	q_0, q_1	q_0
q_1	ϕ	q_2
q_2	q_2	q_2

ACCEPTIBILITY OF STRING BY A NON DETERMINISTIC FINITE AUTOMATA

This fig. represents Transition Diagram for NFA

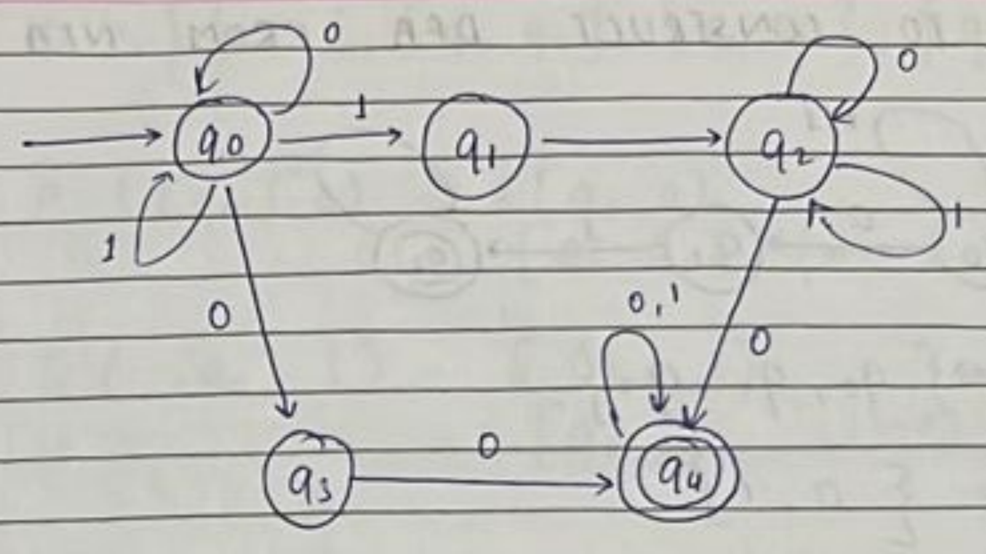


Fig. Transition Diagram for NFA

$x = 0100$

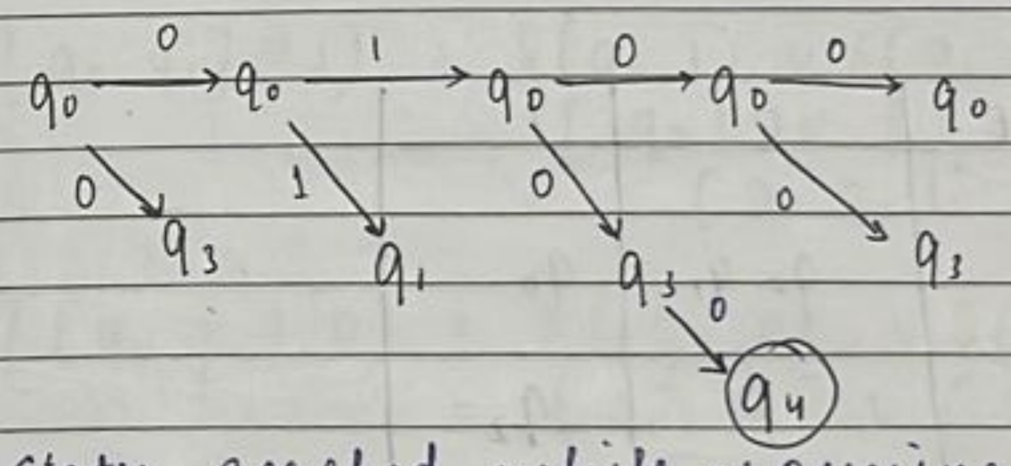


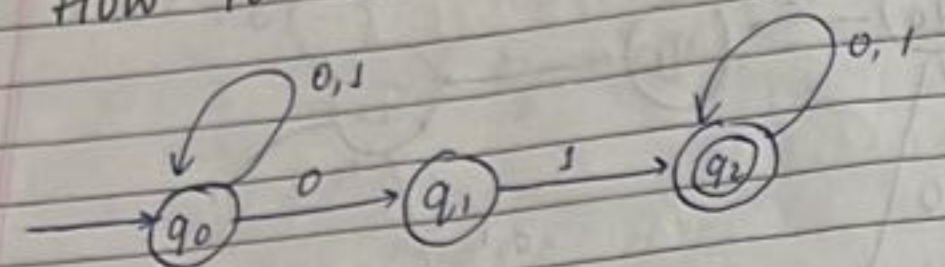
FIG: States reached while processing 0100

Atleast one path reaches the final state then the string is accepted

Q. Differentiate between NFA & DFA

SNO.	PROPERTIES	NFA	DFA
1.	definition	-	-
2.	Ex.	-	-
3.	Transition Function	$\delta = Q \times \Sigma \rightarrow 2^{\lvert Q \rvert}$	$\delta = Q \times \Sigma \rightarrow Q$

HOW TO CONSTRUCT DFA FROM NFA



$Q \rightarrow \{q_0, q_1, q_2\}$

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

q_0 = Initial state

$F = \{q_2\}$

TRANSITION TABLE

STATE	0	1
$\rightarrow q_0$	q_0, q_1	q_0
q_1	ϕ	q_2
q_2	q_2	q_2

$Q' = 2^{|Q|}$
$= 2^3 = 8$

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

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

$\delta'([q_0], 0) = \delta(q_0, 0)$
$= [q_0, q_1]$ (new)

$\delta'([q_0], 1) = \delta(q_0, 1)$
$= [q_0]$ (old)

$\delta'([q_0, q_1], 0) = \delta(q_0, 0) \cup \delta(q_1, 0)$
$= [q_0, q_1] \cup \phi$
$= [q_0, q_1]$

$\delta'([q_0, q_1], 1) = \delta(q_0, 1) \cup \delta(q_1, 1)$
$= [q_0] \cup [q_2]$
$= [q_0, q_2]$ (new)

$\delta'([q_0, q_2], 0) = \delta(q_0, 0) \cup \delta(q_2, 0)$
$= [q_0, q_1] \cup [q_2]$
$= [q_0, q_1, q_2]$ (new)

$\delta'([q_0, q_2], 1) = \delta(q_0, 1) \cup \delta(q_2, 1)$
$= [q_0] \cup [q_2]$
$= [q_0, q_2]$

$\delta'([q_0, q_1, q_2], 0) = \delta(q_0, 0) \cup \delta(q_1, 0) \cup \delta(q_2, 0)$
$= [q_0, q_1] \cup \phi \cup q_2$
$= [q_0, q_1, q_2]$

$$\delta'([q_0, q_1, q_2], 1) = \delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1)$$
$$= q_0 \cup q_2 \cup q_2$$
$$= [q_0, q_2]$$

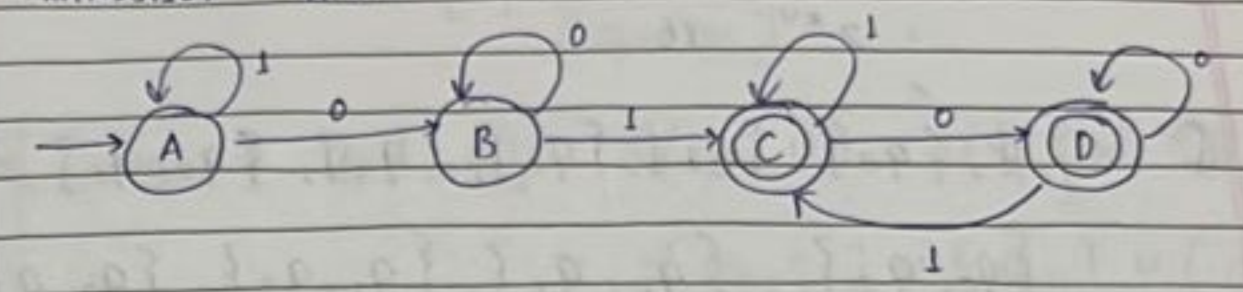
TRANSITION ~~STATE~~ FUNCTION δ'

	STATE	0	1
A	$\rightarrow [q_0]$	$[q_0, q_1]$	$[q_0]$
B	$[q_0, q_1]$	$[q_0, q_1]$	$[q_0, q_2]$
C	$[q_0, q_2]$	$[q_0, q_1, q_2]$	$[q_0, q_2]$
D	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$	$[q_0, q_2]$

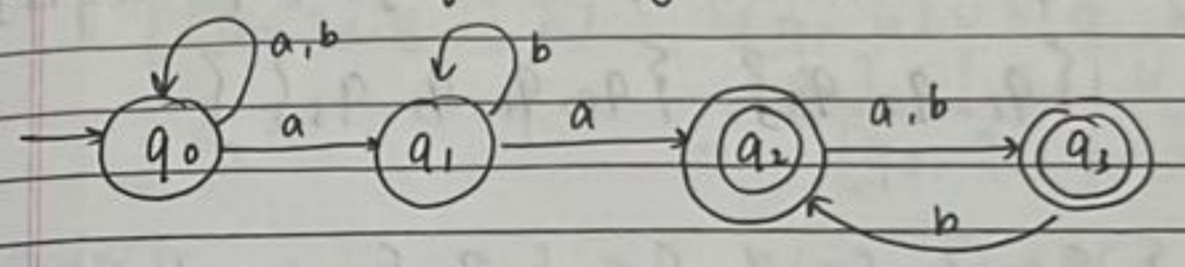
MODIFIED TRANSITION TABLE

STATE	0	1
$\rightarrow$ A	B	A
B	B	C
C	D	C
D	D	C

TRANSITION GRAPH



Q. Convert the following NFA to DFA



$Q = \{q_0, q_1, q_2, q_3\}$
$\Sigma = \{a, b\}$
q_0 = inital state
$F = \{q_2, q_3\}$

TRANSITION TABLE

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

$Q' = 2^{|Q|}$

$= 2^4 = 16$

$Q' = \{ \phi, \{q_0\}, \{q_1\}, \{q_2\}, \{q_3\}, \{q_0, q_1\}, \{q_0, q_2\},$

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

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

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

$F' = \{ \{q_2\}, \{q_3\}, \{q_0, q_2\}, \{q_0, q_3\}, \{q_1, q_2\},$

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

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

$\delta'([q_0], a) = \delta(q_0, a)$

$= [q_0, q_1]$ new

$\delta'([q_0], b) = \delta(q_0, b)$

$= [q_0]$

$\delta'([q_0, q_1], a) = \delta([q_0, a]) \cup \delta(q_1, a)$

$= [q_0, q_1] \cup [q_1]$

$= [q_0, q_1, q_2]$ new

$\delta'([q_0, q_1], b) = \delta(q_0, b) \cup \delta(q_1, b)$

$= [q_0] \cup [q_1]$

$= [q_0, q_1]$

$\delta'([q_0, q_1, q_2], a) = \delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a)$

$= [q_0, q_1] \cup [q_2] \cup [q_3]$

$= [q_0, q_1, q_2, q_3]$ new

$\delta'[(q_0, q_1, q_2), b] = \delta(q_0, b) \cup \delta(q_1, b) \cup \delta(q_2, b)$

$= [q_0] \cup [q_1] \cup [q_3]$

$= [q_0, q_1, q_3]$ new

$\delta'([q_0, q_1, q_2, q_3], a) = \delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a) \cup \delta(q_3, a)$

$= [q_0, q_1] \cup [q_2] \cup [q_3] \cup \phi$

$= [q_0, q_1, q_2, q_3]$

$\delta'([q_0, q_1, q_2, q_3], b) = \delta(q_0, b) \cup \delta(q_1, b) \cup \delta(q_2, b) \cup \delta(q_3, b)$

$= q_0 \cup q_1 \cup q_3 \cup q_2$

$= [q_0, q_1, q_2, q_3]$

$\delta'([q_0, q_1, q_3], a) = \delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_3, a)$

$= [q_0, q_1] \cup [q_2] \cup \phi$

$= [q_0, q_1, q_2]$

$\delta'([q_0, q_1, q_3], b) = \delta(q_0, b) \cup \delta(q_1, b) \cup \delta(q_3, b)$

$= [q_0] \cup [q_1] \cup [q_2]$

$= [q_0, q_1, q_2]$

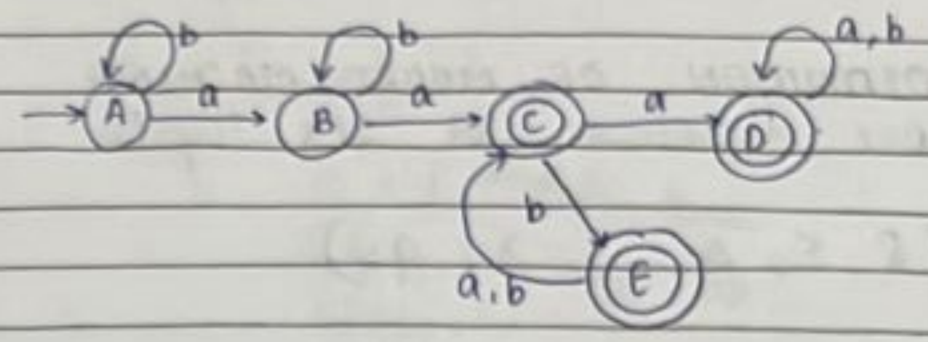
TRANSITION FUNCTION δ'

	STATE	a	b
$\rightarrow A$	$\rightarrow [q_0]$	$[q_0, q_1]$	$[q_0]$
B	$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
C	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
D	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$
E	$[q_0, q_1, q_3]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$

STATE	a	b
$\rightarrow A$	B	A
B	C	B
C	D	E
D	D	D
E	C	C

MODIFIED TRANSITION TABLE

TRANSITION GRAPH



02/02/23 FINITE AUTOMATA WITH OUTPUT

~~MEA~~ MEALY MACHINE

The value of the output function $z(t)$ in the most general case is a function of the present state $q(t)$ and the present input $x(t)$

$$\boxed{z(t) = \lambda(\underset{\text{present state}}{q(t)}, \underset{\text{present I/p}}{x(t)})}$$

where λ is called output function. This generalised model is usually called Mealy machine.

MOORE MACHINE

If the output function $z(t)$ depends only on the present state & is independent of the current input alphabet the output function may be written as

$$\boxed{z(t) = \lambda(\underset{\text{PRESENT STATE}}{q(t)})}$$

This restricted model is called moore

machine

FORMAL DEFINITION OF MOORE MACHINE

The Moore machine is six Tuple

$M = (Q, \Sigma, \underset{\text{all}}{\overset{\text{small}}{\Delta}}, \underset{\text{all}}{\overset{\text{capital}}{\delta}}, \lambda, q_0)$

where

$Q \to$ Finite set of states

$\Sigma \to$ Finite set of input alphabets

$\Delta \to$ output alphabets

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

$\lambda \to$ Output function

$\lambda : Q \to \Delta$

$q_0 \to$ Initial state

PRESENT STATE	NEXT STATE a = 0	NEXT STATE a = 1	OUTPUT
$\to q_0$	q_1	q_2	1
q_1	q_3	q_2	0
q_2	q_2	q_1	1
q_3	q_0	q_3	1

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

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

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

$\delta : Q \times \Sigma \to Q$

$\delta(q_0, 0) = q_1$

$\delta(q_0, 1) = q_2$

$\lambda : Q \to \Delta$

Q For the input string $w = 0101$ the transition of states is given by

$w = 0101$

$$q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_2 \xrightarrow{1} q_1$$

q_0	q_1	q_2	q_2	q_1
1	0	1	1	0

OUTPUT = 10110

* If the input string is of length m, then the output string will be of length $m+1$

FORMAL DEFINITION OF MELAY MACHINE

It is a 6 tuple

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

$Q \rightarrow$ Finite set of state
$\Sigma \rightarrow$ Finite set of input alphabet
$\Delta \rightarrow$ Output alphabets
$\delta \rightarrow$ Transition function which maps
$\delta : Q \times \Sigma \rightarrow Q$

$\lambda \rightarrow$ Output function which maps
$\lambda : Q \times \Sigma \rightarrow \Delta$

$q_0 \rightarrow$ Initial state.

PRESENT STATE	NEXT STATE a=0	OUTPUT	a=1	OUTPUT
$\rightarrow q_1$	q_1	1	q_2	0
q_2	q_4	1	q_4	1
q_3	q_2	1	q_3	1
q_4	q_3	0	q_1	1

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

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

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

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

For the input string $w = 0011$ the transition of states is given

$w = 0011$

$$q_1 \xrightarrow{0} q_1 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{1} q_4$$
$$1 \quad 1 \quad 0 \quad 1$$

op = 1101

In the condition of mealy machine if the input string is of length n, then the output string is also of same length n.

CONVERSION OF MOORE MACHINE INTO MEALY MACHINE

Let $M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ is the moore machine M/C

Q = Finite set of state
Σ = Finite set of input alphabet
Δ : Output alphabets
δ : Transition function which maps
$\delta : Q \times \Sigma \rightarrow Q$
λ = Output function which maps
$\lambda : Q \times \Sigma \rightarrow \Delta$
q_0 = Initial state

let $M' = (Q, \Sigma, \Delta, \delta, \lambda', q_0)$ is the new machine

Q : Finite ~~state~~ set of state
Σ : Finite set of input alphabet
Δ : Output alphabet
δ : transition function which maps
$\delta : Q \times \Sigma \to Q$
λ' : Output function
q_0 : initial state

we made M & M' are equivalent by ignoring the output of ~~mea~~ moore machine on the input symbol.

$$\boxed{\lambda'(q, a) = \lambda(\delta(q, a))}$$

PRESENT STATE	NEXT STATE a=0	NEXT STATE a=1	OUTPUT
$\to q_0$	q_1	q_2	1
q_1	q_3	q_2	0
q_2	q_2	q_1	1
q_3	q_0	q_3	1

$\lambda'(q_0, 0) = \lambda(\delta(q_0, 0))$
$= \lambda(q_1)$
$= 0$

$\lambda'(q_0, 1) = \lambda(\delta(q_0, 1))$
$= \lambda(q_2)$
$= 1$

$\lambda'(q_1, 0) = \lambda(\delta(q_1, 0))$
$= \lambda(q_3) = 1$

$\lambda'(q_1, 1) = \lambda(\delta(q_1, 1))$
$= \lambda(q_2) = 1$

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

$\lambda'(q_2, 1) = \lambda(\delta(q_2, 1))$
$= \lambda(q_1) = 0$

$\lambda'(q_3, 0) = \lambda(\delta(q_3, 0))$
$= \lambda(q_0) = 1$

$\lambda'(q_3, 1) = \lambda(\delta(q_3, 1))$
$= \lambda(q_3) = 1$

PRESENT STATE	a=0	o/p	a=1	o/p
q_0	q_1	0	q_2	1
q_1	q_3	1	q_2	1
q_2	q_2	1	q_1	0
q_3	q_0	1	q_3	1

construct a mealy machine ~~m~~ which is ~~equiva~~ equivalent to moore machine given in the table.

PRESENT STATE	a=0	a=1	OUTPUT
$\rightarrow q_0$	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

let $M = (Q, \Sigma, \delta, \Delta, q_0)$ be moore machine

Q = Finite set of state

Σ = Finite set of input alphabet

δ = transition funct^n which maps $\delta : Q \times \Sigma \rightarrow Q$

Δ = Output alphabets

q_0 : inital state

let $M' = (Q, \Sigma, \delta, \Delta', q_0)$ be mealy machine

Q = Finite set of state

Σ = Finite set of input alphabet

Δ' = Output alphabet

δ : transition funct^n $\delta : Q \times \Sigma \rightarrow Q$

q_0 : inital state

we made M & M' are equivalent by ignoring the output of moore machine on the input symbol

$$\lambda'(q_0, 0) = \lambda(\delta(q_0, 0)) = \lambda(q_3) = 0$$

$$\lambda'(q_0, 1) = \lambda(\delta(q_0, 1)) = \lambda(q_1) = 1$$

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

$$\lambda'(q_1, 1) = \lambda(\delta(q_1, 1)) = \lambda(q_2) = 0$$

$$\lambda'(q_2, 0) = \lambda(\delta(q_2, 0))$$

$\lambda'(q_2, 1) = \lambda(\delta(q_2, 1))$
$= \lambda(q_3) = 0$

$\lambda'(q_3, 0) = \lambda(\delta(q_3, 0))$
$= \lambda(q_3) = 0$

$\lambda'(q_3, 1) = \lambda(\delta(q_3, 1))$
$= \lambda(q_0) = 0$

PRESENT STATE	NEXT STATE a=0	O/P	a=1	O/P
$\to q_0$	q_3	0	q_1	1
q_1	q_1	1	q_2	0
q_2	q_2	0	q_3	0
q_3	q_3	0	q_0	0

Q Construct a Moore machine that takes input from $(0+1+2)^*$ and print the residue mod 5 of the input treated as a number with base 3

$\Sigma = \{0, 1, 2\}$
$\Delta = \{0, 1, 2, 3, 4\}$

5 |3| $\quad$ 3 % 5 $\quad \frac{3}{5} = 0.6$

PRESENT STATE	NEXT STATE a=0	a=1	a=2	OUTPUT
q_0	$(0\times3+0)\%5 = q_0$	$(0\times3+1)\%5 = q_1$	$(0\times3+2)\%5 = q_2$	0
q_1	$(1\times3+0)\%5 = q_3$	$(1\times3+1)\%5 = q_4$	$(1\times3+2)\%5 = q_0$	1
q_2	$(2\times3+0)\%5 = q_1$	$(2\times3+1)\%5 = q_2$	$(2\times3+2)\%5 = q_3$	2
q_3	$(3\times3+0)\%5 = q_4$	$(3\times3+1)\%5 = q_0$	$(3\times3+2)\%5 = q_1$	3
q_4	$(4\times3+0)\%5 = q_2$	$(4\times3+1)\%5 = q_3$	$(4\times3+2)\%5 = q_4$	4

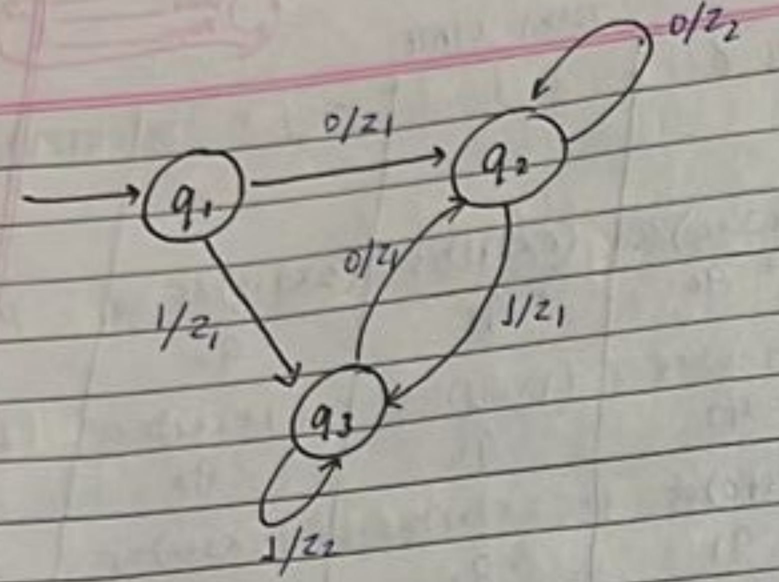
CONVERSION OF MEALY MACHINE INTO MOORE MACHINE

STEP 1 → Check different number of output associated with each state.

STEP 2 → Split each state into states equal to the number of output associated with that state

STEP 3 → Remove all output associated with each state. Write output directly related to the state.

Ex.



$Q = \cancel{\{q_1, q_3}\ \{q_1, q_2, q_3\}$

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

$\Delta = \{z_1, z_2, \cancel{z_3}\}$

PRESENT STATE	NEXT STATE a=0	OUTPUT	a=1	OUTPUT
$\rightarrow q_1$	q_2	z_1	q_3	z_1
q_2	q_2	z_2	q_3	z_1
q_3	q_2	z_1	q_3	z_2

STEP 1. Check diff no. of ~~an~~ outputs associated with each state.

$q_1 = -$

$q_2 = z_1, z_2$

$q_3 = z_1, z_2$

STEP 2 Split each state into states equal to the no of output associated with that state.

$q_2 \to z_1, z_2$: q_{21} q_{22}

$q_3 \to z_1, z_2$: q_{31} q_{32}

PRESENT STATE	NEXT STATE a=0	OUTPUT	a=1	OUTPUT
$\rightarrow q_1$	q_{21}	z_1	q_{31}	z_1
q_{21}	q_{22}	z_2	q_{31}	z_1
q_{22}	q_{22}	z_2	q_{31}	z_1
q_{31}	q_{21}	z_1	q_{32}	z_2
q_{32}	q_{21}	z_1	q_{32}	z_2

STEP 3. Remove all the output associated with each

PRESENT STATE	NEXT STATE a = 0	a = 1	OUTPUT
q_1	q_{21}	q_{31}	-
q_{11}	q_{22}	q_{21}	z_1
q_{22}	q_{22}	q_{31}	z_2
q_{31}	q_{21}	q_{32}	z_1
q_{32}	q_{21}	q_{32}	z_2

Q. Consider the mealy machine describe by the transition table given in table construct a moore machine which is equivalent to mealy machine.

PRESENT STATE	NEXT STATE a = 0	O/P	a = 1	O/P
q_1	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

Step 1 $q_1 = \cancel{0}, 1$
$q_2 = 0, 1$
$q_3 = 0$
$q_4 = 0, 1$

Step 2 q_1 — 1 q_2 — 0, 1 q_3 — 0 q_4 — 0, 1

q_{11} q_{20} q_{21} q_{30} q_{40} q_{41}

PRESENT STATE	NEXT a = 0	O/P	a = 1	O/P
q_{11}	q_{30}	0	q_{20}	0
q_{20}	q_{11}	1	q_{40}	0
q_{21}	q_{11}	1	q_{40}	0
q_{30}	q_{21}	1	q_{11}	1
q_{40}	q_{41}	1	q_{30}	0
q_{41}	q_{41}	1	q_{30}	0

STEP 3. Remove.

PRESENT STATE	NEXT STATE $a=0$	NEXT STATE $a=1$	O/P
q_{11}	q_{30}	q_{20}	1
q_{20}	q_{11}	q_{40}	0
q_{21}	q_{21}	q_{40}	1
q_{30}	q_{21}	q_{11}	0
q_{40}	q_{41}	q_{30}	0
q_{41}	q_{41}	q_{30}	1

convert it into zero

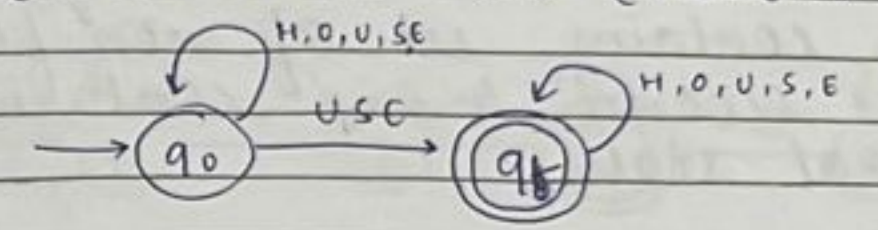
PRESENT STATE	NEXT STATE $a=0$	NEXT STATE $a=1$	O/P
$\rightarrow q_{01}$	q_{30}	q_{20}	0
q_{11}	q_{30}	q_{20}	1
q_{20}	q_{11}	q_{40}	0
q_{21}	q_{11}	q_{40}	1
q_{30}	q_{21}	q_{11}	0
q_{40}	q_{41}	q_{30}	0

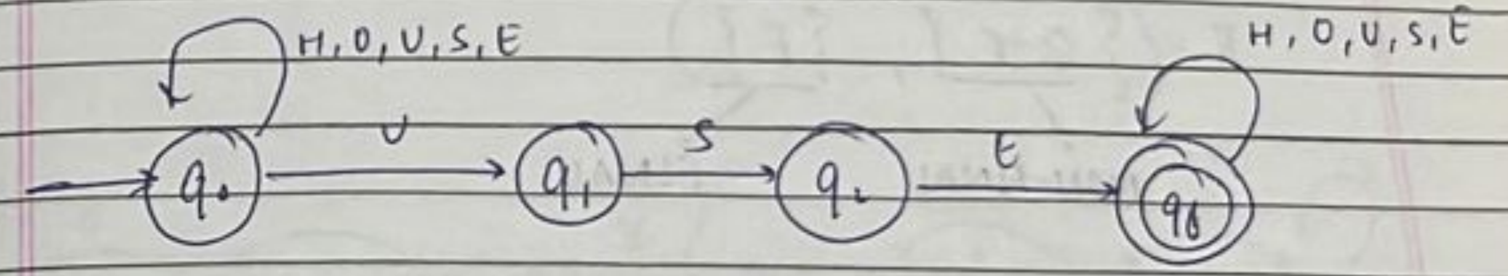
q_0 a, b $(a+b)^*$

NFA TO

Q. Desgin a finite automata that reads strings made up of letters in the word "HOUSE" & recognises those string that contain the word "USE" at any way.

$L = (H+O+U+S+E)^*\ USE\ (H+O+U+S+E)^*$





PRESENT STATE	$a = H$	O	U	S	E
$\rightarrow q_0$	q_0	q_0	q_0, q_1	q_0	q_0
q_1	ϕ	ϕ	ϕ	q_2	ϕ
q_2	ϕ	ϕ	ϕ	ϕ	q_f
q_f	q_f	q_f	q_f	q_f	q_f

$$Q' = 2^{|Q|}$$
$$= 2^4 = 16$$

MINIMIZATION OF FINITE AUTOMATA.

ALGO

STEP 1 : Split the state into two groups one group contains set of non-final state and second group contains the set of final states.

$\pi = (\underbrace{\{Q-F\}}_{\text{NON-FINAL}}, \underbrace{\{F\}}_{\text{FINAL}})$

STEP 2 For each group of π do split each group of π into subgroups. place the subgroup in π new

END

STEP 3 if $\pi \neq \pi_{new}$ Repeat the step II with $\pi = \pi_{new}$

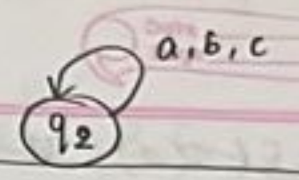
STEP 4 Choose one representative from each group of π new and place them in Q'

STEP 5 Eliminate the dead state and unreachable state from M'

DEAD

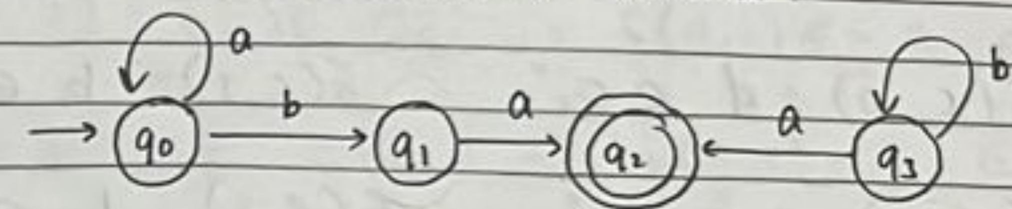
A non-final state is a dead state if it has transition onto itself on all

input input symbol.

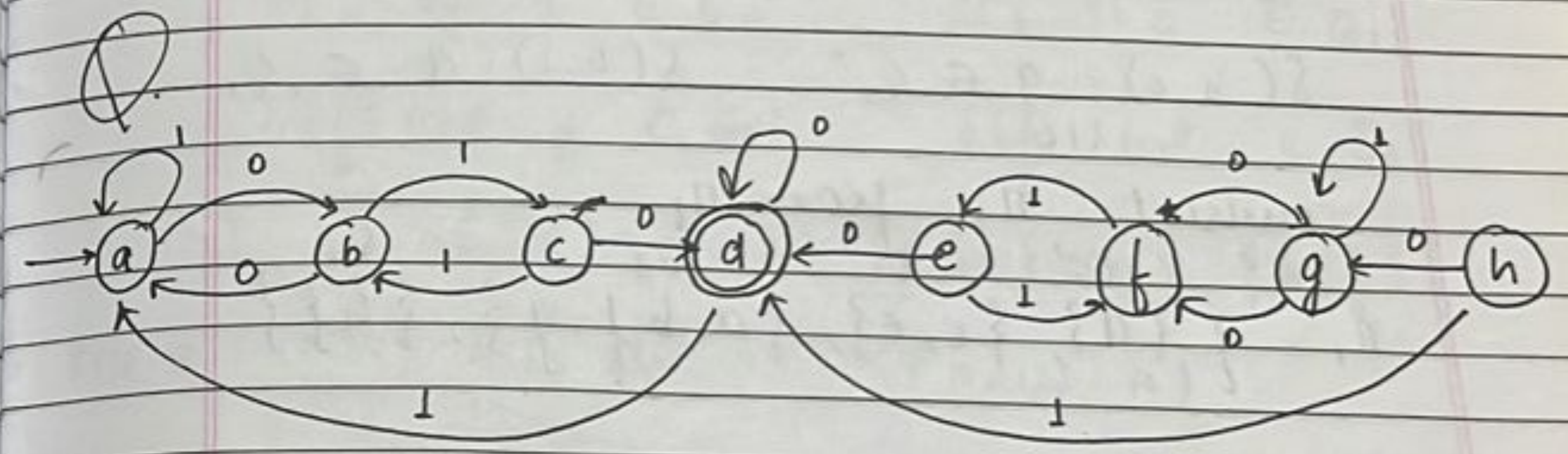


UNR

Any final or non-final state is not reachable from the ~~se~~ start state is called a unreachable state.



Q.



STEP - 1 $\pi = (\{Q-F\}, \{F\})$

STATE	a	1
a	b	a
b	a	c
c	d	b
d	d	a
e	d	f
f	g	e
g	f	g
h	g	d

Q_2^0 Q_1^0

$\pi = (\{a,b,c,d,e,f,g,h\}, \{d\})$

final state wale split nhi honge.

Q_1° $\quad$ Q_2°

~~STEP 2 δ(q0)~~ $\pi_0 = (\{d\}, \{a, b, c, e, f, g$

construct π_1 from π_0

$\pi_1 = \{\{d\}, \{c, e\}, \{a, b, f, g\}, \{h\}\}$

$\delta(c,0) = d \in Q_1^{\circ}$ $\quad$ $\delta(c,1) = b \in Q_2^{\circ}$

$\delta(e,0) = d \in Q_1^{\circ}$ $\quad$ $\delta(e,1) = f \in Q_2^{\circ}$

$\delta(h,0) = g \in Q_2^{\circ}$ $\quad$ $\delta(h,1) = d \in Q_1^{\circ}$

construct π_2 from π_1

$\pi_1 = \{\underset{Q_1'}{\{d\}}, \underset{Q_2'}{\{c, e\}}, \underset{Q_3'}{\{a, b, f, g\}}, \underset{Q_4'}{\{h\}}\}$

$\pi_2 = \{\{d\}, \{c, e\}, \{h\}, \{a, g\}, \{b, f\}\}$

$\delta(a,0) = b \in Q_3'$ $\quad$ $\delta(a,1) = a \in Q_3'$

$\delta(g,0) = f \in Q_3'$ $\quad$ $\delta(g,1) = g \in Q_3'$

$\delta(b,0) = a \in Q_3'$ $\quad$ $\delta(b,1) = c \in Q_2'$

$\delta(f,0) = g \in Q_3'$ $\quad$ $\delta(f,1) = e \in Q_2'$

$\pi_2 = \{\underset{Q_1''}{\{d\}}, \underset{Q_2''}{\{c, e\}}, \underset{Q_3''}{\{h\}}, \underset{Q_4''}{\{a, g\}}, \underset{Q_5''}{\{b, f\}}\}$

$\delta(c,0) = d \in Q_1''$ $\quad$ $\delta(c,1) = b \in Q_5''$

$\delta(e,0) = d \in Q_1''$ $\quad$ $\delta(e,1) = f \in Q_5''$

$\delta(a,0) = b \in Q_5''$ $\quad$ $\delta(a,1) = a \in Q_4''$

$\delta(g,0) = f \in Q_5''$ $\quad$ $\delta(g,1) = g \in Q_4''$

$\delta(b,0) = a \in Q_4''$ $\quad$ $\delta(b,1) = c \in Q_2''$

$\delta(f,0) = g \in Q_4''$ $\quad$ $\delta(f,1) = e \in Q_2''$

$\pi_3 = \{\{d\}, \{c, e\}, \{h\}, \{a, g\}, \{b, f\}\}$

STEP 3 $\quad$ $\pi_3 = \pi_2$ $\quad$ ($\pi_{new} = \pi$)

STEP 4. select one representative from each group & place them in Q'

$Q' = \{a, b, c, d, h\}$

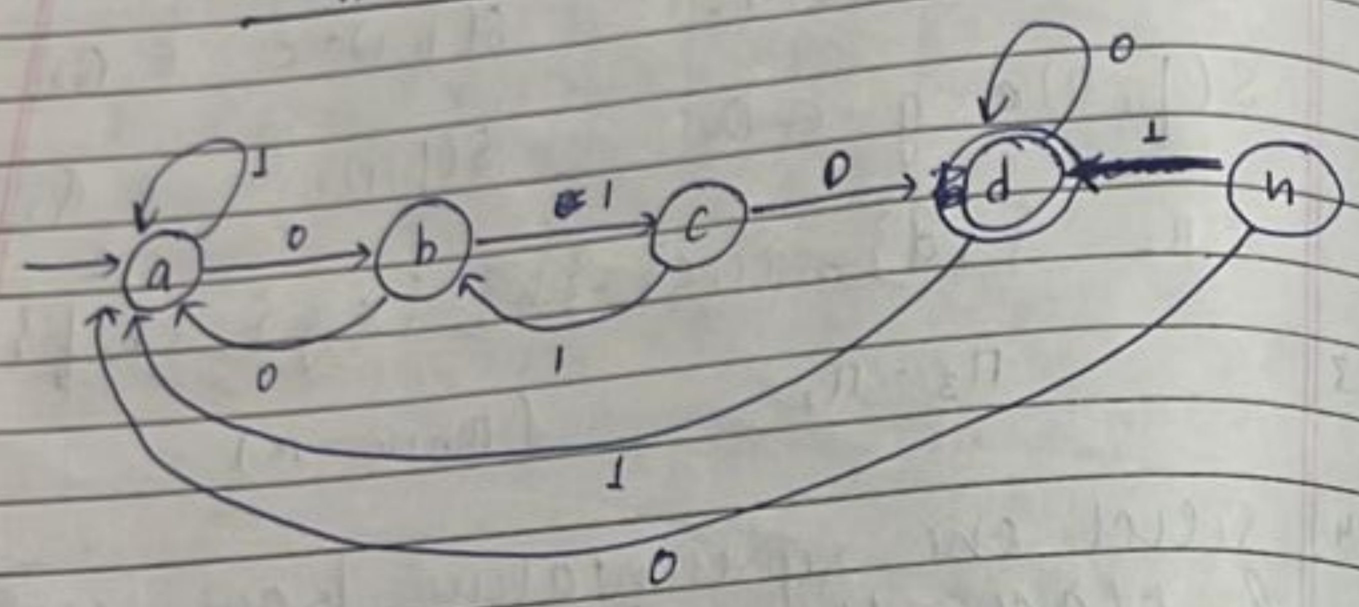
STATE	I/P 0	1
→ a	b	a
b	a	c
c	d	b
d	d	a
h	g	d

Replace g by a because a is representative of g

The modified table is

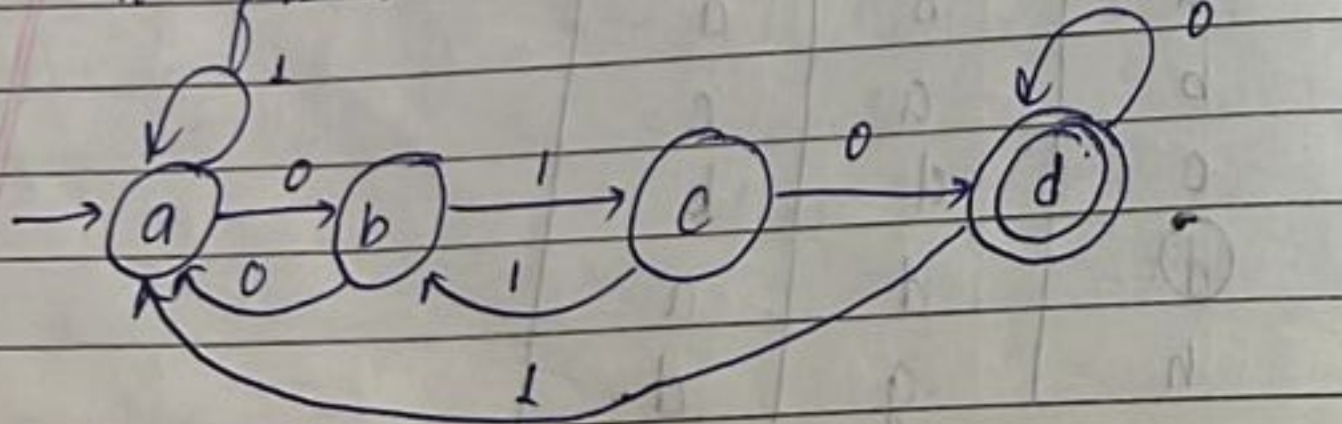
STATE	I/P 0	1
→ a	b	a
b	a	c
c	d	b
(d)	d	a
h	a	d

$M' =$



STEP 5. Eliminate the deadstate and unreachable state from M'

Here, in the present problem H is unreachable state so we can eliminate h from M'



Ex.

STATE	I/P ALPHABET 0	1
→ q_0	q_1	q_5
q_1	q_6	q_2
(q_2)	q_0	q_2
q_3	q_2	q_6
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

q_0 q_1 (q_5)

q_4 q_7 (q_5)

q_6 q_4

STEP - 1

$$\Pi = (\{Q-F\}, \{F\})$$

$$\Pi_0 = (\{q_2\}, \{q_0, q_1, q_3, q_4, q_5, q_6, q_7\})$$

STEP-2

$$\Pi_0 = (\underset{Q_1^0}{\{q_2\}}, \underset{Q_2^0}{\{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}})$$

CONSTR

$$\Pi_1 = (\underset{Q_1'}{\{q_2\}}, \underset{Q_2'}{\{q_1, q_7\}}, \underset{Q_3'}{\{q_3, q_5\}}, \underset{Q_4'}{\{q_0, q_4, q_6\}})$$

CONSTRUCT Π_2 from Π_1

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

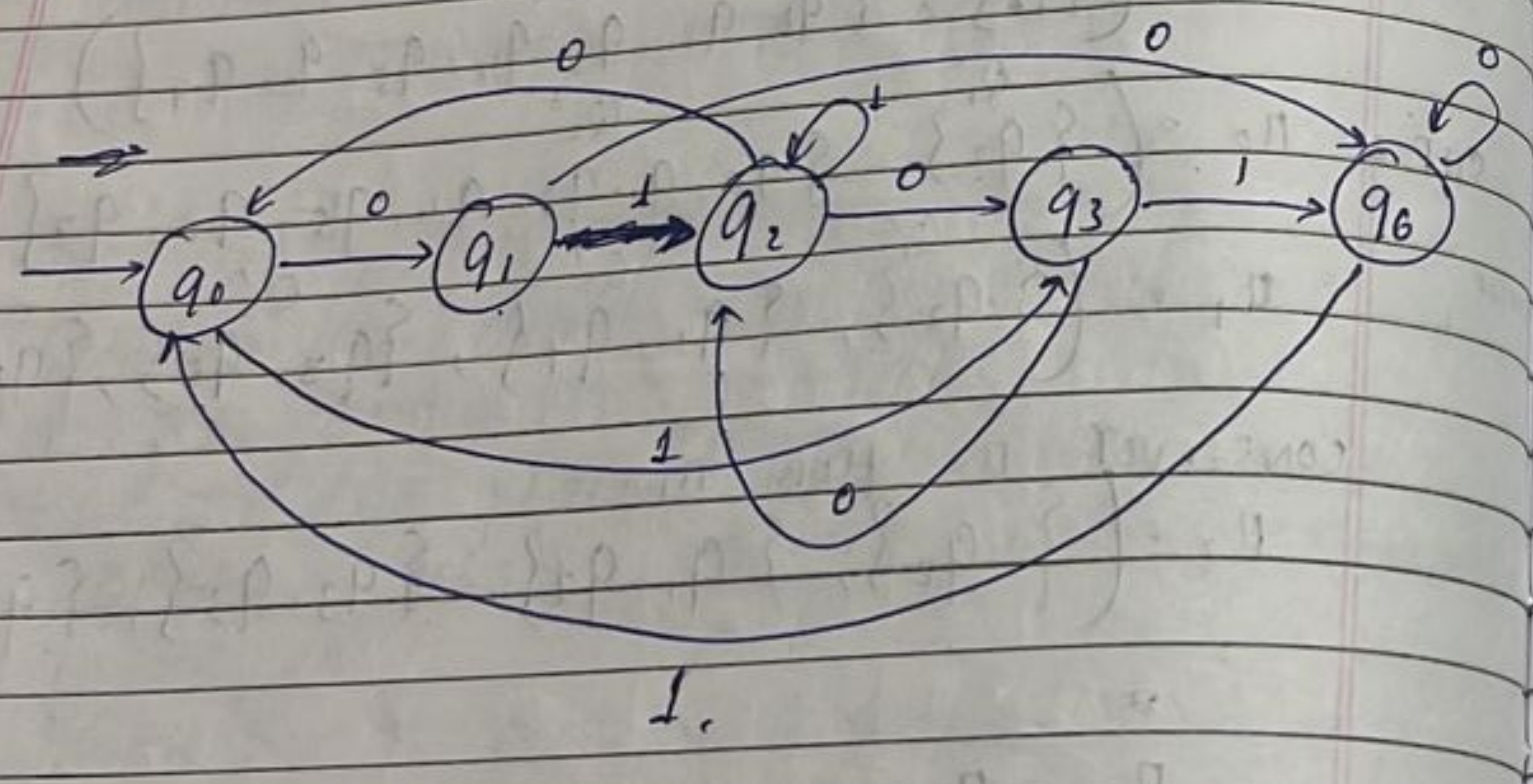
$$\Pi_3 = \Pi_2$$

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

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
q_6	q_6	q_4

Replace q_5 by q_3 & q_4 by q_0

	0	1
$\rightarrow q_0$	q_1	q_3
q_1	q_6	q_2
q_2	q_0	q_2
q_3	q_2	q_6
q_6	q_6	q_0



MYHILL NERODE THEOREM

CONCEPT :

Every language $L \subseteq \Sigma^*$

$x R_L y$, iff

xw is in L exactly when yw is in L for every $w \in \Sigma^*$

1. if $x R_L x$ therefore R_L is Reflexive
2. $x R_L y \Rightarrow y R_L x$ therefore R_L is symmetric
3. $x R_L y$ & $y R_L z \Rightarrow x R_L z$ therefore R_L is Transitive

So, R_L is an equivalence relation

$\therefore$ R_L divides equivalence relation into equivalent classes.

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

Q = Finite ~~state of set~~ set of state

Σ = Finite set of input alphabet

δ = Transition function which maps

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

q_0 = Initial state

F = Final state ($F \subseteq Q$)

Therefore, R_M such that $x R_M y$

$\delta(q_0, x) = \delta(q_0, y)$

$x \, R_M \, x \Rightarrow \delta(q_0, x) = \delta(q_0, x)$ [REFLEXIVE]

$x \, R_M \, y \Rightarrow \delta(q_0, x) = \delta(q_0, y)$ [SYMMETRIC]

$y \, R_M \, x \Rightarrow \delta(q_0, y) = \delta(q_0, x)$

$y \, R_M \, z \Rightarrow \delta(q_0, y) = \delta(q_0, z)$

$\Rightarrow \delta(q_0, x) = \delta(q_0, z)$ [TRANSITIVE]

When all the three properties is satisfied then the relation is equivalent.

~~right invariant~~

RIGHT INVARIANT EQUIVALENCE RELATION

If add a string to already entiting string to their right side and still ~~string~~ they are related then they are called right invariant equivalence relation.

If $x R_M y \Rightarrow x z R_M y z$ [z IS RIGHT INVARIENT]

$\delta(q_0, x) = \delta(q_0, y)$ — (1)

$\delta(q_0, x) = \delta(\delta(q_0, x), z)$ [$\because \delta(q_0, x) = \delta(q_0, y)$]

$= \delta(\delta(q_0, y), z)$

$\delta(q_0, xz) = \delta(q_0, yz)$.

Therefore $xz \, R_M \, yz$
So, R_M is right invariant relation.

V.IMP

STATEMENT OF MYHILL NERODE THEOREM

The following three statement are equivalent

1.) Let $L \subseteq \Sigma^*$ is accepted by some finite automata

2.) L is the union of equivalence classes of a right invariant Equivalence relation which is of finite index

3.) Let R_L is a relation defined on L such that $x \, R_L \, y$ iff xw is in L exactly when yw is in L for every $w \in \Sigma^*$ then R_L is of finite index.

PROOF (1) $\Rightarrow$ (2)

Let $L \subseteq \Sigma^*$ is accepted by a finite automata $M = (Q, \Sigma, \delta, q_0, F)$ define a relation R_M on Σ^*

$x \, R_M \, y$ if

$\Rightarrow \delta(q_0, x) = \delta(q_0, y)$

R_M is an equivalence because.

1. $x R_M x$: $\delta(q_0, x) = \delta(q_0, x)$

$\therefore R_M$ is Reflexive

2. if $x R_M y$ then $y R_M x$: $\delta(q_0, x) = \delta(q_0, y)$

$\therefore R_M$ is symmetric

3. if $x R_M y$ & $y R_M z$ then $x R_M z$

$x R_M y = \delta(q_0, x) = \delta(q_0, y)$

$y R_M z = \delta(q_0, y) = \delta(q_0, z)$

$\Rightarrow \delta(q_0, x) = \delta(q_0, z)$

$\therefore R_M$ is transitive.

$\therefore R_M$ is equivalence relation.

R_M divides the set of Σ^* into set of equivalence classes. No of equivalence classes cannot exceeds the number of state in the finite automata therefore R_M is of finite index

L is the union of equivalence classes corresponding the final state that means L is union of x such that $\delta(q_0, x)$ is in F

$\therefore R_M$ is right invarient because whenever $x R_M y$ then $xz R_M yz$.

PROOF (2) $\Rightarrow$ (3)

Let R_M is an Right invariant equivalence relation ~~relation~~ which is of finite index.

$xRy \Rightarrow xwRyw$, $\forall w \in \Sigma^*$

$x R_L y$ iff xw is in L exactly when yw is in R_L where R_L is the ~~refinement~~ refinement of R if xRy then $x R_L y$.

So, every equivalence classes of ~~capital~~ R will be some of equivalences classes of R_L, because R is of finite index. Therefore R_L will be of finite index

PROOF (3) $\Rightarrow$ (1)

~~R_L is right invariant $x R_L y \Rightarrow x w R_L$~~

R_L is right invariant

$x R_L y \Rightarrow xw R_L yw$ $\forall w \in \Sigma^*$

xwz should be exactly when wz is in L $\forall z \in \Sigma^*$,

Suppose $v = wz$, should be in exactly when

yv is in L, $\forall v \in \Sigma^*$

$\therefore xw R_L yw$

$\therefore R_L$ is right invariant equivalence relation

13/02/23

UNIT - II

REGULAR EXPRESSION

ALPHABET, STRING AND LANGUAGE

ALPHABET

- An alphabet usually denoted by Σ. It is a finite non-empty set of symbol
- $\Sigma = \{0, 1, 2, 3 - - - - 9\}$ is the decimal alphabet.
- $\Sigma = \{0, 1\}$ is the binary alphabet
- $\Sigma = \{a, b, c - - - z, A, B - - - Z\}$ is the basic english alphabet
- Similarly ASCII code & UNI CODE are the important example of the alphabet used to represent character in computer memory.
- Note that empty set is ~~set~~ represented by $\phi = \{\}$ and any infinite set like $N = \{1, 2, 3 - - - \}$ are not alphabets

STRING (OR WORD)

- String or word is a finite sequence of symbol drawn from some alphabet for eg, "apple", it is string over english alphabet.
- Let w be a string from alphabet Σ the length of string w is the no of occurance of symbols in w & denoted by $|w|$

→ Let $\Sigma = \{a, t, c\}$ then "at", "cat", "act" are the string from the alphabet Σ.
$|at| = 2$ $|cat| = 3$ $|act| = 3$

→ Let Σ 1011 01 is the string from alphabet $\Sigma = \{0, 1\}$, $|\Sigma| = 6$.

POWERS OF AN ALPHABET

If Σ is an alphabet we define Σ^l to be the set of string of length each of whose symbol is in Σ

* $\Sigma^0 = \{\epsilon\}$ it is only string of length 0

Ex. if $\Sigma^1 = \{a, t, c\}$

$$\Sigma^2 = \Sigma^1 . \Sigma^1$$
$$= \{a, t, c\} . \{a, t, c\}$$
$$= \{aa, at, ac, ta, tt, tc, ca, ct, cc\}$$

$$\Sigma^3 = \Sigma^2 . \Sigma$$
$$\Sigma^4 = \Sigma^3 . \Sigma$$

Σ^* (KNOWN AS KLEEN CLOSURE OR STAR CLOSURE OF Σ)
is the set of string over an alphabet Σ

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \cdots \cup \Sigma^\infty$

$\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i$

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

$\Sigma^0 = \{\epsilon\}$

$\Sigma^1 = \{a, b\}$

$\Sigma^2 = \Sigma.\Sigma = \{a, b\}.\{a, b\} = \{aa, ab, ab, bb\}$

$\Sigma^3 = \Sigma^2.\Sigma$

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

$= \{\epsilon\} \cup \{a, b\} \cup \{aa, ab, ab, bb\} \cup \cdots$

$= \{\epsilon, a, b, aa, ab, bc, bb \cdots \}$

Σ^+ (KNOWN AS POSITIVE CLOSURE OF Σ)
Is the set of all string over an alphabet Σ except the empty string.

$\Sigma^+ = \Sigma^* - \{\epsilon\}$

$\Sigma^+ = \bigcup_{i=1}^{\infty} \Sigma^i$

$= \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \cdots$

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

~~STRING~~ STRING RELATED TERM

PREFIX A prefix of string w is a string obtained by removing 0 or more symbol from the end of the string w
w = "bit"

For eg $\{\epsilon$, b, bi, bit$\}$ are prefix of string bit

SUFFIX
A suffix of string w is a string obtain by removing 0 or more symbol from the beginning of the string

$\{\epsilon$, it, t, bit$\}$ are suffixes of string bit.

SUBSTRING
A substring of string w is a string obtained by removing prefix & suffic from the string w
$\{\epsilon$, b, bi, it, t, i, bit$\}$

4 marks

LANGUAGE
A language over the alphabet Σ is

a set of string drawn from/obtain from Σ^*

DIFFERENT OPERATION PERFORMED BY LANG

If L_1 & L_2 are two language over the input alphabet Σ

→ UNION

~~If~~ the union of L_1 & L_2 is the language that contain all string that are either in L_1 or L_2

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

let $L_1 = \{a, ab, bb\}$
$L_2 = \{b, ab, aa\}$

$L_1 \cup L_2 = \{a, ab, bb, b, aa\}$

→ INTERSECTION

Intersection L_1 & L_2 is the language that contains all string that are both in L_1 & L_2

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

let $L_1 = \{a, ab, bb\}$
$L_2 = \{b, ab, aa\}$

$L_1 \cap L_2 = \{ab\}$

+ = or
· = and

→ DIFFERENCE

difference of L_1 & L_2 is the language that contain all string that are in L_1 & not in L_2.

$L_1 - L_2 = \{ x \mid x \in L_1 \text{ and } x \notin L_2 \}$

$L_1 - L_2 = \{bb, a\}$

→ CONCATENATION

concatenation of L_1 & L_2 is a language of all string formed by concatenating a string from L_1 with a string from L_2.

$L_1 . L_2 = \{ xy \mid x \in L_1 \text{ \& } y \in L_2 \}$

$= \{ab, aab, aaa, abb, abab, bbaa$

→ KLEEN STAR OR KLEEN CLOSURE

of a language L is denoted by L^* is set of all string obtain by concatenating zero or more string from L.

$L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \cdots$

$$\boxed{L^* = \bigcup_{i=0}^{\infty} L^i}$$

$L^0 = \{\epsilon\}$, $L^1 = \{0, 1\}$, $L^2 = L.L$
$= \{00, 10, 01, 11\}$

$L^* = \{\epsilon, 0, 1, 00, 11, 01, 10\}$

→ ~~KLEEN~~ POSITIVE CLOSURE

$$\boxed{L^+ = \bigcup_{i=1}^{\infty} L^i}$$

REGULAR EXPRESSION

DEFINITION

Let Σ be the alphabet, ϵ is the symbol for the null string then the regular expression over Σ is defined recursively.

1. ϕ is a Regular Expression denoting the language ϕ that is
$L(\phi) = \phi$

2. ϵ is a Regular Expression representing the language $\{\epsilon\}$. That is $L(\epsilon) = \{\epsilon\}$

3. For each $a \in \Sigma$, "a" is a regular expression denoting the language $\{a\}$
That is $L(a) = \{a\}$.

4. If r & s are two regular expression over the alphabet Σ denoting the language L, r, s respectively, then

(a) $r+s$ is a Regular Expression denoting the set
$L(r) \cup L(s)$

(b) $r.s$ is a Regular Exp denoting the set $L(r).L(s)$

(c) r^* is a Regular Exp denoting the set $(L(r))^*$

(d) (r) is a Regular Exp denoting the set $L(r)$

$a\epsilon = a$
$\epsilon a = a$

$(a+b)^2 = \{a,b\}.\{a,b\}$

Ex. Here, are some example of regular exp of some languages over the alphabet $\Sigma = \{a, b\}$

REGULAR EXPRESSION	CORRESPONDING REGULAR LANGUAGE
ϵ	$\{\epsilon\}$
a	$\{a\}$
b	$\{b\}$
$a+b$	$\{a, b\}$
ab	$\{ab\}$
$aa+b+ba$	$\{aa, b, ba\}$
$a(a+b)b$	$\{aab + abb\}$
$(a+b)^2$	$\{aa, ab, ba, bb\}$
a^*	$\{\epsilon, a, aa, aaa ----\}$
a^*b	$\{b, ab, aab, aaab ----\}$
$(a+b)^*$	$\{\epsilon, a, b, aa, ab, ba, bb ---\}$
$(a+b)^+$	$\{a, b, aa, ab, ba, bb ---\}$

2 marks

Regular Languages are regular sets. The language that are defined by Regular expression are called regular

sets
Let r be the regular expression then ~~L(r)~~ r the language denoted by r is called L(r). If a language L is regular we can construct finite automaton accepting regular language L.

MORE EXAMPLE OF REGULAR EXP

Let $\Sigma = \{1, 2, 3\}$, give regular expression for each of the following regular lang over i/p alphabet Σ

1. $\{1, 2, 3\} = 1 + 2 + 3$

2. $\{\epsilon, 1, 11, 111 \ldots\}$
It is a set of 0 or more occurances of 1. It is denoted by the regular exp 1^*

3. $\{\epsilon, 1, 2, 11, 12, 21, 22, 111, 112, 121, 122 \ldots\}$
It is a set of all strings formed by 1 and 2. It is denoted by the regular expression $(1+2)^*$

4. $\{1, 12, 122, 1222, \ldots\}$
Set of string beginning with 1 followed by 0 or more occurance of 2. It is denoted by regular exp $1(2)^*$

5. $\{\epsilon, 2, 3, 33, 333 \ldots\}$
Set with word 2 and 0 or more occurance of 3. It is denoted by regular expression $2 + (3)^*$

* Let $\Sigma = \{0, 1\}$ with regular exp. for each of the following ~~exp.~~ regular exp. over Σ.

1. Set of string with second symbol from the last is 1.

$(0+1)^* \; 1 \; (0+1) \longrightarrow$ LAST SYMBOL CAN EITHER 0 OR 1

ARBITARY STRING; SECOND LAST SYMBOL AS 1

2. Set of string with 0 as third symbol from the last

$(0+1)^* \; 0 \; (0+1)(0+1)$

3. Set of string with 011 as substring
$(0+1)^* \; 011 \; (0+1)^*$

4. Set of strings with beginning with 0 & ending with 1.
$0 \; (0+1)^* \; 1$

5. Set of string beginning with 11

$11(0+1)^*$

6. Set of strings with atleast one 0.

$(0+1)^*0(0+1)^*$

7. Set of strings with excatly 3 one's

$~~(0+1)^*111(0+1)^*~~$ $0^*10^*10^*10^*$

8. Set of strings with length 6 or less

$(\epsilon+0+1)^6$

9. Set of string with length 6.

$(0+1)^6$ or $(0+1)(0+1)(0+1)(0+1)(0+1)(0+1)$

Q. Set of all strings over the input alphabet a & b such that the 10^{th} symbol from right end is b

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

$(a+b)^* b(a+b)^9$

or

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

IDENITIES FOR REGULAR EXPRESSION

I1 $\phi + R = R$

I2 $\phi . R = R . \phi = \phi$

I3 $\epsilon . R = R . \epsilon = R$

I4 $\epsilon^* = \epsilon$ and $\phi^* = \epsilon$

I5 $R + R = R$

I6 $R^* . R^* = R^*$

I7 $RR^* = R^* . R$

I8 $(R^*)^* = R^*$

* I9 $\epsilon + R.R^* = \epsilon + R^*R = R^*$

I10 $(PQ)^* . P = P(QP)^*$

* I11 $(P+Q)^* = (P^*Q^*)^* = (P^* + Q^*)^*$

I12 $(P+Q).R = PR + QR$ & $R(P+Q) = RP + RQ$

* I13 $\epsilon . R^* = R^*$

Simplification of regular expression

Q. Prove that $(1+00^*1) + (1+00^*1)(0+10^*1)^*(0+10^*1) = 0^*1(0+10^*1)^*$

Taking L.H.S

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

$= (1+00^*1)(\epsilon + (0+10^*1)^*(0+10^*1))$ [Taking $(1+00^*1)$ common]

$= (1+00^*1)(0+10^*1)^*$ [$R = (0+10^*1)$, $R^* = (0+10^*1)^*$, I9: $\epsilon + R^*R = R^*$]

$= (\epsilon+00^*)1(0+10^*1)^*$

$= 0^*1(0+10^*1)^*$ [$R = 0$, $R^* = 0$, I9: $\epsilon + R^*R = R^*$]

$=$ R.H.S

Q. Prove that $\epsilon + 1^*(011)^*(1^*(011)^*)^* = (1+011)^*$

$= \epsilon + 1^*(011)^*(1^*(011)^*)^*$

$R = 1^*(011)^*$

$R^* = (1^*(011)^*)^*$

I9: $\epsilon + RR^*$

$= (1^*(011)^*)^*$

$= (P^*Q^*)^*$ [$P = 1$, $Q = (011)$, $P^*.Q^* = 1^*(011)^*$]

$= (1+011)^*$ [I11: $(P^*.Q^*)^* = (P+Q)^*$]

If P contains ϵ then arden's theorem

IMP ARDEN'S THEOREM

THEOREM Let P & Q be two Regular Expression over Σ. If P does not contains ϵ then the following equation in R.

$$\boxed{R = Q + RP} \quad (1)$$

has a unique solution (i.e one and only one solution) given by $R = QP^*$

PROOF: L.H.S R

$= Q + RP$

$= Q + (QP^*)P$

$= Q(\epsilon + P^*.P)$

$R = QP^*$ [$\epsilon + R^*R = R^*$]

Hence eq^n (1) is satisfied then $R = QP^*$. This means $R = QP^*$ is a solution of eq^n (1)

UNIQUENESS

To prove uniqueness consider eq^n (1) here replacing R by Q + RP on the R.H.S. we get the solution

R.H.S

$= Q + RP$

$= Q + (Q + RP)P$

$= Q + QP + RP^2$

$= Q + QP + (Q + RP).P^2$

$= Q + QP + QP^2 + RP^3$

$= Q + QP + QP^2 + (Q + RP)P^3$

$= Q + QP + QP^2 + QP^3 + RP^4$

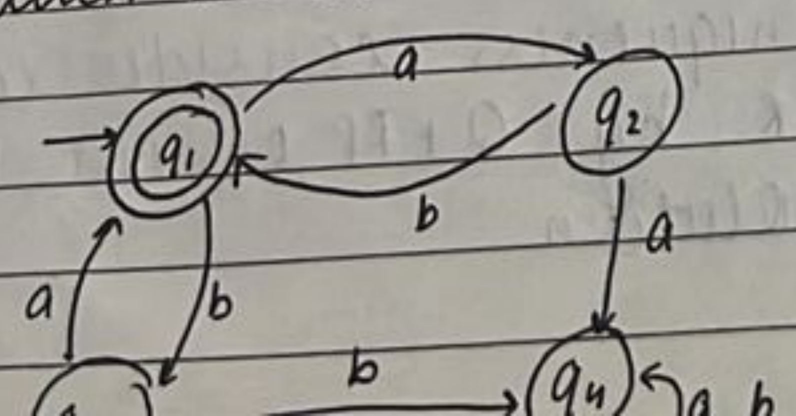
$= Q + QP + QP^2 + QP^3 \cdots + QP^i + RP^{i+1}$

$= Q + QP + QP^2 + QP^3 \cdots + P^i) + RP^{i+1} \quad \forall\, i \geq 0$

$= Q(\epsilon + P + P^2 + P^3 \cdots + P^i) + RP^{i+1}$ — (2)

We ~~k~~now show that any solution of eq (1) is equivalent to QP^*. Suppose R satisfy eq^n (2) then it satisfies eq^n (1) then it belongs to set $Q(\epsilon + P + P^2 \cdots + P^i) + RP^{i+1}$. w belongs to set $Q(\epsilon + P + P^2 \cdots + P^i) + RP^{i+1}$. As P does not contain ϵ, RP^{i+1} has no string of length less than $i+1$ and so w is not in the set RP^{i+1}. This means $w \in Q(\epsilon + P + P^2 + P^3 + \cdots P^i)$ and hence QP^*

Prove that the finite automata whose transition diagram is given in the figure accepts the set of all string over the alphabet {a, b} with an equal no of a's & b's i.e each prefix has atmost one more a then b and atmost one more b than a's



$q_1 = \epsilon + q_2 b + q_3 a$ — (1)

$q_2 = q_1 a$ — (2)

$q_3 = q_1 b$ — (3)

$q_4 = q_2 a + q_3 b + q_4 a + q_4 b$ — (4)

from eq (1)

$q_1 = \epsilon + q_2 b + q_3 a$

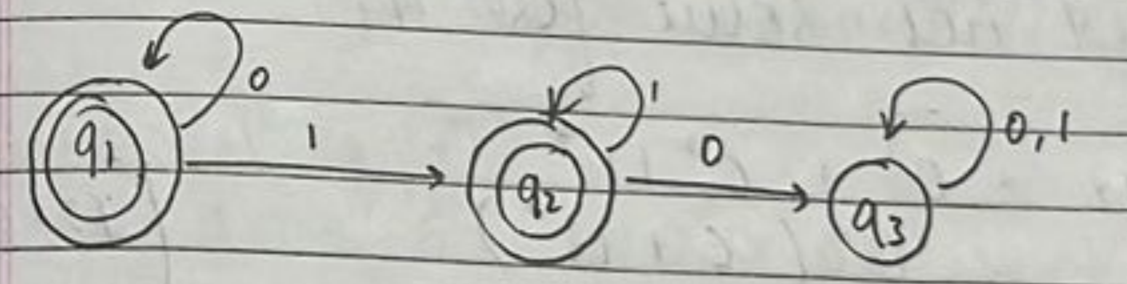
$q_1 = \epsilon + q_1 ab + q_1 ba$

$\overset{R}{q_1} = \overset{Q}{\epsilon} + \overset{R}{q}\,\overset{P}{(ab + ba)}$ $\quad \begin{bmatrix} R = q_1 \\ Q = \epsilon \\ P = (ab + ba) \end{bmatrix}$

$q_1 = \epsilon \cdot (ab + ba)^*$

$= (ab + ba)^*$ $\quad [\because \epsilon \cdot R^* = R^*]$

Q.



$q_1 = \epsilon + q_1 0$ — (I)

$q_2 = q_1 1 + q_2 1$ — (II)

$q_3 = q_2 0 + q_3 0 + q_3 1$ — (III)

from eq^n (I)

$q_1 = \epsilon + q_1 0$ $\quad \left[\because \begin{array}{l} R = q_1 \\ Q = \epsilon \\ P = 0 \end{array} \right]$

$R = Q + RP$

$q_1 = \epsilon 0^*$

$q_1 = 0^*$ — (4)

from eq^n (11)

$q_2 = q_1 1 + q_2 1$

from eq (4),

$q_2 = 0^* 1 + q_2 1$ $\quad [q_1 = 0^*]$

$R = Q + RP$ $\quad \left[\begin{array}{l} R = q_2 \\ Q = 0^* 1 \\ P = 1 \end{array} \right]$

$q_2 = 0^* 1 1^*$ — (5)

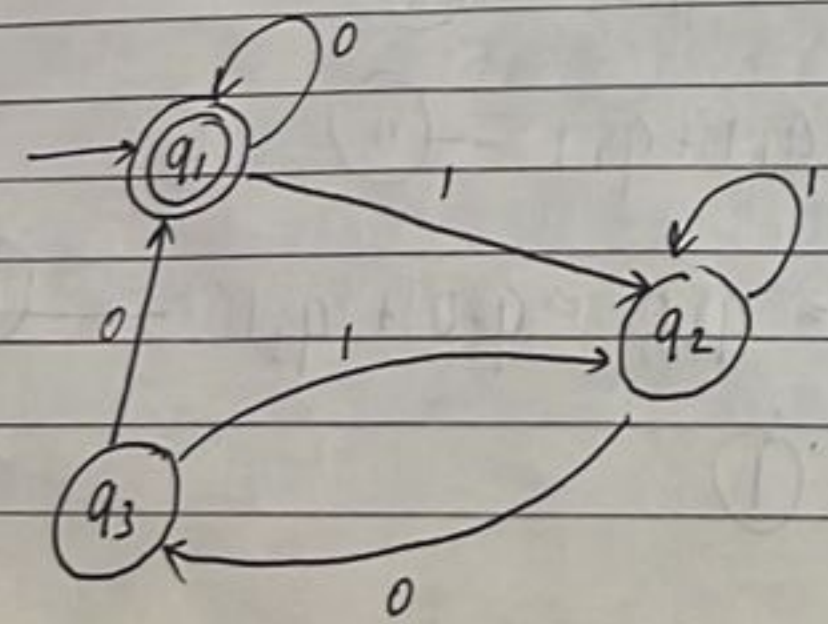
Here there are two final states q_1 & q_2 we need not solve for q_3

$q_1 + q_2 = 0^* + 0^* 1 1^*$

$= 0^* (\epsilon + 1 1^*)$ $\quad [\because \epsilon + RR^* = R^*]$

$q_1 + q_2 = 0^* 1^*$

Q. Imp



$q_1 = \epsilon + q_1 0 + q_3 0$ — (1)

$q_2 = \cancel{q_2 0 + q_2 1}$ $q_2 1 + q_1 1 + q_3 1$ — (2)

$q_3 = q_2 0$ — (3)

$q_1 = \epsilon + q_1 0 + q_3 0$ — FINAL STATE

from eq^n (2)

$q_2 = q_1 1 + q_2 1 + q_3 1$

From (3), $q_3 = q_2 0$

$= q_1 1 + q_2 1 + q_2 0 1$

$q_2 = q_1 1 + q_2 (1 + 01)$

$R = Q + RP$ $\quad \left[\begin{array}{l} R = q_2 \\ Q = q_1 1 \\ P = (1 + 01) \end{array} \right]$

$q_2 = q_1 1 (1+01)^*$ — (4)

From eq^n (4) & (3)

$q_3 = (q_1 1 (1+01)^*) 0$ — (5) $\quad [q_2 = q_1 1 (1+01)^*]$

From eq^n (1)

$q_1 = \epsilon + q_1 0 + q_3 0$

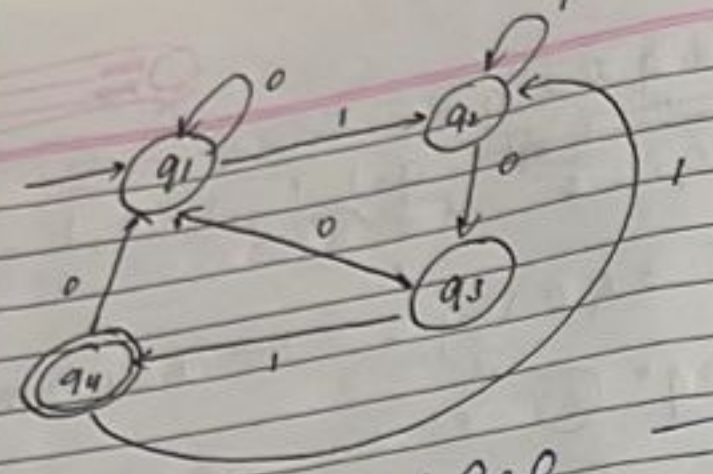
$q_1 = \epsilon + q_1 0 + q_1 1 (1+01)^* 00$ $\quad \left[\begin{array}{l} \text{From 5} \\ q_3 = q_1 1 (1+01)^* 0 \end{array} \right]$

$q_1 = \epsilon + q_1 [0 + 1(1+01)^* 00]$

$R = Q + RP$ $\quad \left[\begin{array}{l} R = q_1 \\ Q = \epsilon \\ P = [0 + 1(1+01)^* 00 \end{array} \right]$

$q_1 = \epsilon [0 + 1(1+01)^* 00]^*$

$q_1 = 0 + 1(1+01)^* 00$

Q 

$q_1 = \epsilon + q_1 0 + q_4 0 + q_3 0$ —— ①

$q_2 = q_2 1 + q_1 1 + q_4 1$ —— ②

$q_3 = q_2 0$ —— ③

$q_4 = q_3 1$ —— ④ — Final state.

~~Take~~ $q_4 = q_3 1$

From ③ $q_4 = (q_2 0) 1$ —— ⑤

~~$q_3 = (q_2 1 + q_1 1 + q_4 1) 0$~~ [From 2 $q_2 = q_2 1 + q_1 1 + q_4 1$]

~~q_4~~ From eq ②

$q_2 = q_2 1 + q_1 1 + (q_2 0) 1 1$

$q_2 = q_1 1 + q_2 (1 + 011)$

$R = Q + RP$ $\left[\begin{array}{l} R = q_2 \\ Q = q_1 1 \\ P = (1+011) \end{array}\right]$

$q_2 = q_1 1 (1+011)^*$ —— ⑥

From ①

$q_1 = \epsilon + q_1 0 + (q_2 0) 10 + q_2 00$

$= \epsilon + q_1 0 + q_1 1 (1+011)^* 010 + q_1 1 (1+011)^* 00$

$= \epsilon + q_1 0 + q_1 1 (1+011)^* (010 + 00)$

$= \epsilon + q_1 [0 + 1(1+011)^* (00 + 010)]$

$R = Q + RP$ $\left[\begin{array}{l} R = q_1 \\ Q = \epsilon \\ P = 0 + 1(1+011)^* (00+010) \end{array}\right]$

$q_1 = \epsilon [0 + 1(1+011)^* (00+010)]^*$

$q_1 = [0 + 1(1+011)^* (00+010)]^*$

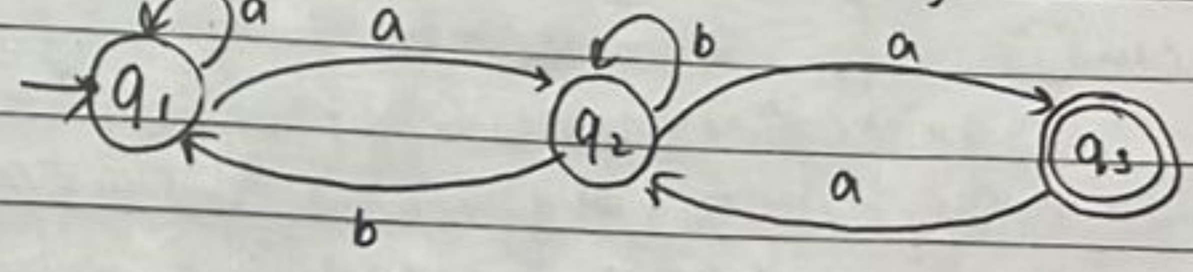
From ⑤

$q_4 = (q_2 0) 1$

$q_4 = (q_1 1 (1+011)^*) 01$ From ⑥ $[q_2 = q_1 1 (1+011)^*]$

$q_4 = [0 + 1(1+011)^* [00 + 010]]^* 1 (1+011)^* 01$

Q. Consider the transition system given in figure prove that the strings recognised $(a + a(b+aa)^* b)^* a (b+aa)^* a$



$q_1 = \epsilon + q_1 a + q_2 b$ —— (1)

$q_2 = q_1 a + q_2 b + q_3 a$ —— (2)

$q_3 = q_2 a$ —— (3)

~~$q_3 = q_2 a$ — final state~~

~~$q_3 = (q_2 b + q_3 a) a$ [From (2)]~~

~~From (2)~~

~~$q_2 = q_2 b + q_3 a$~~

~~$= q_2 b + (q_2 a) a$~~

~~$q_2 = q_2 (b + aa)$~~

$q_3 = (q_2 a)$ — final stat

From (2)

$q_2 = q_1 a + q_2 b + (q_2 a) a$

$q_2 = q_1 a + q_2 (b + aa)$

$R = Q + RP$

$\begin{bmatrix} R = q_2 \\ Q = q_1 a \\ P = (b + aa) \end{bmatrix}$

Arden's theorem

$R = QP^*$

$R\ q_2 = q_1 a (b + aa)^*$ —— (4)

From (1) $q_1 = \epsilon + q_1 a + q_2 b$

$q_1 = \epsilon + q_1 a + (q_1 a (b + aa)^*) b$ [From 4]

$q_1 = \epsilon + q_1 (a + a(b + aab)^* b)$

$= R = Q + RP$

$R = QP^*$

$\begin{bmatrix} R = q_1 \\ Q = \epsilon \\ P = a + (a(b+aab)^* b) \end{bmatrix}$

$q_1 = \epsilon\ (a + a(b + aab)^*) b)^*$

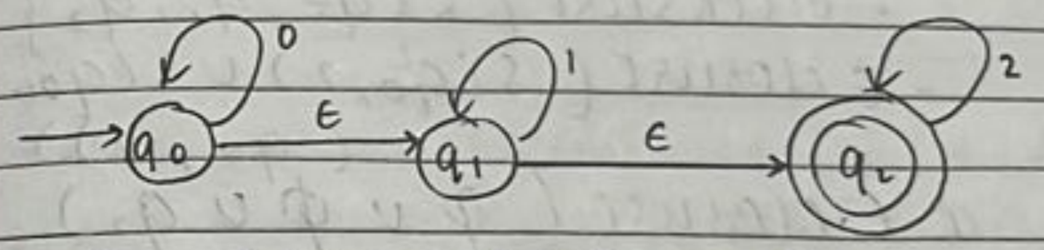
$q_1 = (a + a(b + aab)^* b)^*$ —— (5)

$q_3 = q_2 a$

$q_3 = [q_1 a (b + aa)^*] a$
From (5)

$q_3 = [a + a(b + aab)^* b)^*][a(b + aa)^*] a$ —— (6)

CONVERSION OF NFA WITH ϵ MOVES TO WITHOUT ϵ MOVES NFA.



ϵ closure (q) = set of states which are reachable from state q on ϵ-input including state 'q'

ϵ closure $q_0 = \{q_0, q_1, q_2\}$ jo state hai voh khe ga in plus ϵ jaha tak pahuch rahe hai

ϵ closure $q_1 = \{q_1, q_2\}$

ϵ closure $q_2 = \{q_2\}$

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

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

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

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

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

$\delta'(q_0, 2) = \epsilon\ closure(\delta(\epsilon\ closure(q_0), 2))$
$= \epsilon\ closure(\delta\{q_0, q_1, q_2\}, 2))$
$= \epsilon\ closure(\delta(q_0, 2) \cup \delta(q_1, 2) \cup \delta(q_2, 2))$
$= \epsilon\ closure(\phi \cup \phi \cup q_2)$
$= \epsilon\ closure(q_2)$
$\delta'(q_0, 2) = \{q_2\}$

$\delta'(q_1, 0) = \epsilon\ closure(\delta(\epsilon\ closure(q_1), 0)$
$= \epsilon\ closure(\delta(q_1, q_2), 0))$
$= \epsilon\ closure(\delta(q_1, 0) \cup \delta(q_2, 0))$
$= \epsilon\ closure(\phi \cup \phi)$
$= \epsilon\ closure(\phi)$
$= \phi$

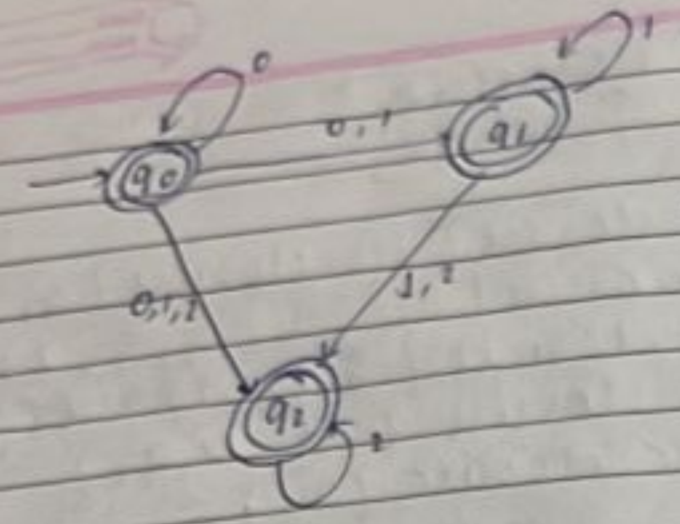
$\delta'(q_1, 1) = \epsilon\ closure(\delta(\epsilon\ closure(q_1), 1)$
$= \epsilon\ closure(\delta(q_1, q_2), 1)$
$= \epsilon\ closure(\delta(q_1, 1) \cup \delta(q_2, 1))$
$= \epsilon\ closure(q_1 \cup \phi)$
$= \epsilon\ closure(q_1)$
$= \{q_1, q_2\}$

$\delta'(q_1, 2) = \epsilon\ closure(\delta(\epsilon\ closure(q_1, 2))$
$= \epsilon\ closure(\delta(\{q_1, q_2\}, 2))$
$= \epsilon\ closure(\delta(q_1, 2) \cup \delta(q_2, 2))$
$= \epsilon\ closure(\phi \cup q_2)$
$= \epsilon\ closure(q_2)$
$= \{q_2\}$

$\delta'(q_2, 0) = \epsilon\ closure(\delta(\epsilon\ closure(q_2, 0))$
$= \epsilon\ closure(\delta(q_2, 0))$
$= \epsilon\ closure(\phi)$
$= \phi$

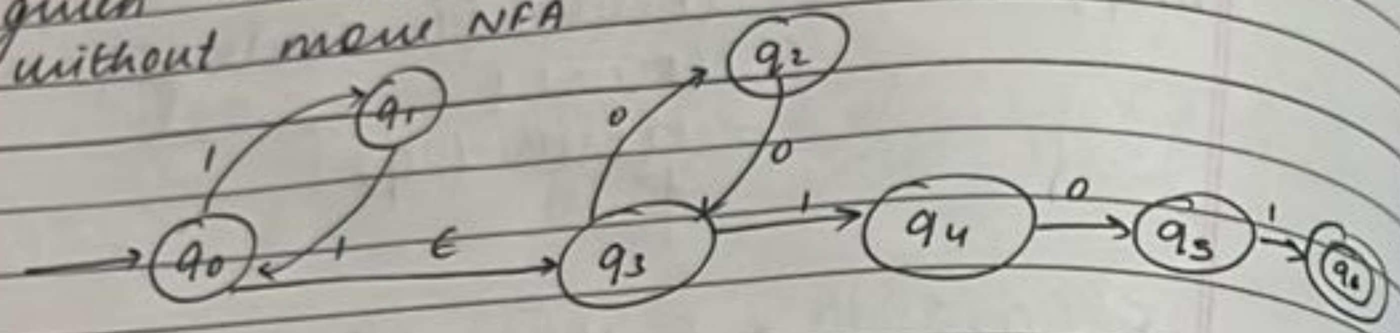
$\delta'(q_2, 1) = \epsilon\ closure(\delta(\epsilon\ closure(q_2, 1))$
$\delta'(q_2, 1) = \epsilon\ closure(\delta(q_2, 1))$
$\delta'(q_2, 1) = \epsilon\ closure(\phi)$
$= \phi$

$\delta'(q_2, 2) = \epsilon\ closure(\delta(\epsilon\ closure(q_2, 2))$
$= \epsilon\ closure(\delta\{q_2\}, 2))$
$= \epsilon\ closure(q_2)$
$= q_2$
$\delta'(q_2, 2) = \{q_2\}$



WITHOUT ε-MOVE
NFA

(Q) given NFA with ε moves is convert to without move NFA



$\epsilon\,closure(q_0) = \{q_0, q_3\}$
$\epsilon\,closure(q_1) = \{q_1\}$
$q_2 = \{q_2\}$
$q_3 = \{q_3\}$
$q_4 = \{q_4\}$
$q_5 = \{q_5\}$
$q_6 = \{q_6\}$

$\delta'(q_0, 0) = \epsilon\,closure(\delta(\epsilon\,closure(q_0), 0)$
$= \epsilon\,closure(\delta\{q_0, q_3\}, 0)$
$= \epsilon\,closure(\delta(q_0, 0) \cup \delta(q_3, 0))$
$= \epsilon\,closure(\phi \cup q_2)$
$= \epsilon\,closure\ q_2$
$\delta'(q_0, 0) = \{q_2\}$

$\delta'(q_0, 1) = \epsilon\,closure(\delta(\epsilon\,closure(q_0), 1)$
$= \epsilon\,closure(\delta(q_0, 1) \cup \delta(q_3, 1))$
$= \epsilon\,closure(q_1 \cup q_4)$
$= \{q_1, q_4\}$

$\delta'(q_1, 0) = \epsilon\,closure(\delta(\epsilon\,closure(q_1), 0)$
$= \epsilon\,closure(\delta(q_1, 0))$
$= \epsilon\,closure(\phi)$
$= \cancel{\{q_1\}}\ \phi$

$\delta'(q_1, 1) = \epsilon\,closure(\delta(\epsilon\,closure(q_1, 1)$
$= \epsilon\,closure(\delta\,\epsilon(q_1, 1))$
$= \epsilon\,closure(q_0)$
$= \{q_0, q_3\}$

$\delta'(q_2, 0) = \epsilon\,closure(\delta(\epsilon\,closure(q_2, 0))$
$= \epsilon\,closure(\delta(q_2, 0))$
$= \epsilon\,closure(q_3)$
$= \{q_3\}$

$\delta'(q_2, 1) = \epsilon$ $(q_2, 1)$
$= \epsilon$ $q_2, 1$
$= \epsilon$ ϕ
$= \phi$

$\delta'(q_3, 0) = \epsilon$ $(q_3, 0)$
$=$ $(q_3, 0)$
$=$ q_2
$= \{q_2\}$

$\delta'(q_3,1) = \epsilon \quad (q_3,1)$
$= \epsilon \quad q_3, 1$
$= \epsilon \quad q_4$
$= \{q_4\}$

$\delta'(q_4,0) = \epsilon \quad (q_4,0)$
$\epsilon \quad (q_4,0)$
$\epsilon \quad q_5$
$= \{q_5\}$

$\delta'(q_4,1) = \epsilon$
$=$
$=$
$= \phi$

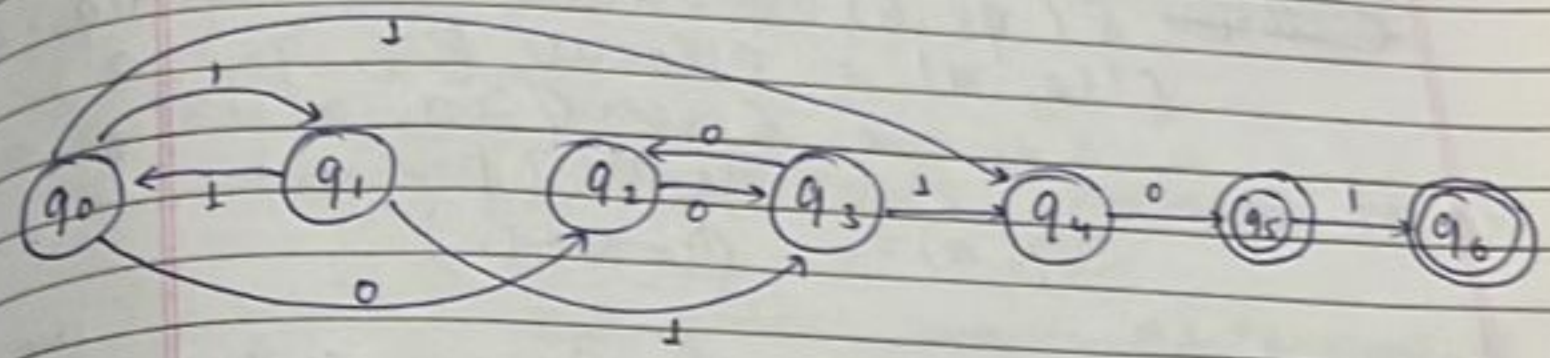
$\delta'(q_5,0) =$

$= \phi$

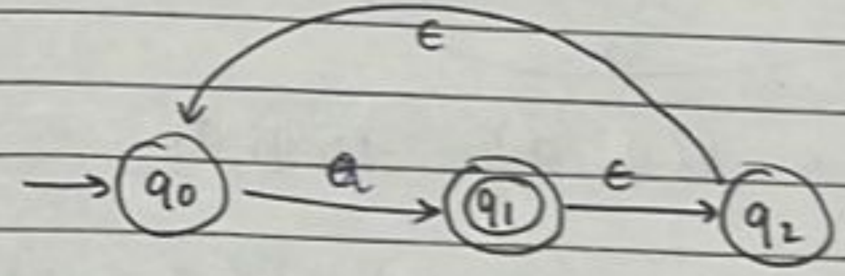
$\delta'(q_5,1) =$
$=$
$=$
$= \{q_6\}$

$\delta'(q_6,0) =$
$=$
$=$
$= \phi$

$\delta'(q_6,1) =$
$=$
$=$
$= \phi$



Q. convert the following ϵ NFA to DFA



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

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

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

$= \epsilon\ closure\ (\phi \cup (q_0, q_1, q_2)) \cup \phi)$

$= \epsilon\ closure\ (q_1)$

$= (q_0, q_1, q_2)$

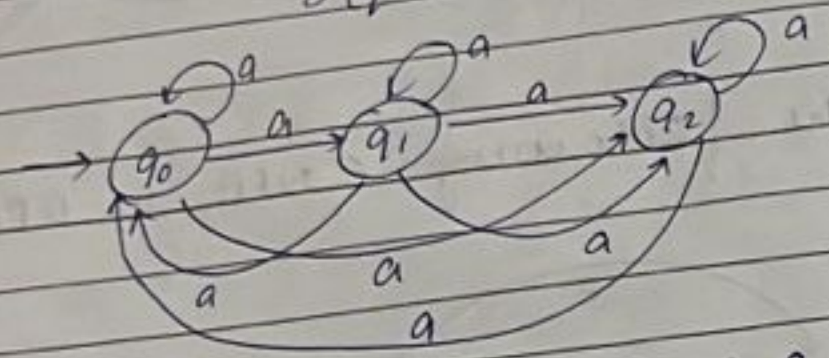
$\delta'(q_2, a) = \delta\ \epsilon\ closure\ (\delta(\epsilon\ closure(q_2), a))$

$\delta'(q_2, a) = \epsilon\ closure\ (\delta(q_1, q_0), a)$

$= \epsilon\ closure\ (\delta(q_2, a) \cup \delta(q_0, a))$

$= \epsilon\ closure\ (\phi \cup q_1)$

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



State	Input a
q_0	q_0, q_1, q_2
q_1	q_0, q_1, q_2
q_2	q_0, q_1, q_2

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

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

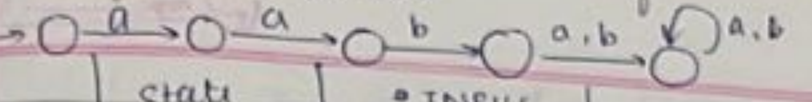
$= (q_0, q_1, q_2) \cup (q_0, q_1, q_2) \cup (q_0, q_1, q_2)$

$= (q_0, q_1, q_2)$

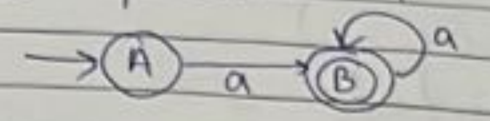
	State	INPUT a
A	$\rightarrow q_0$	q_0, q_1, q_2
B	q_0, q_1, q_2	q_0, q_1, q_2

star matlab itself pe transition.

$aab(a+b)(a+b)^*$



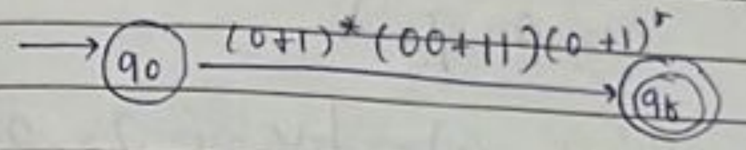
State	INPUT a
→ A	B
B	B

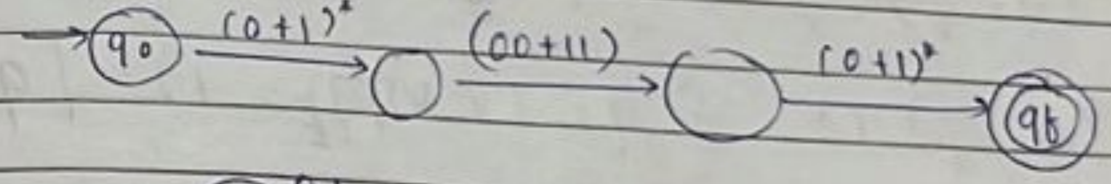


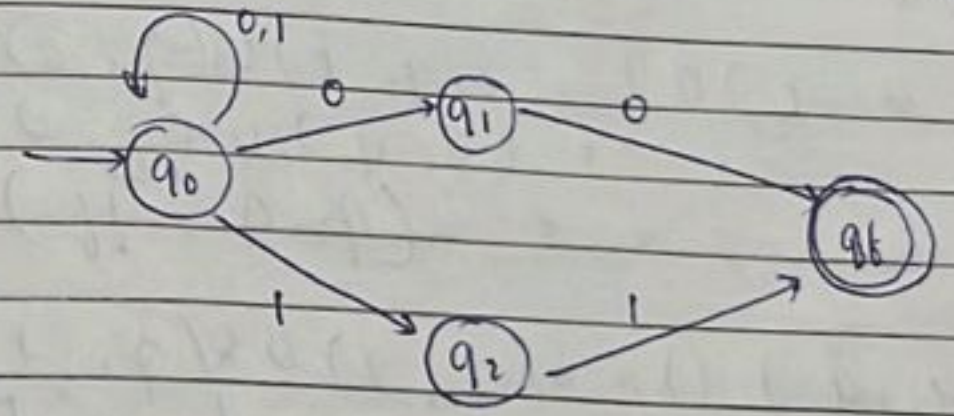
CONVERSION OF REGULAR EXPRESSION TO DFA.

Ex. Construct the finite automata Equivalent to the Regular expression

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







State	0	1
→ q_0	q_0, q_1	q_2, q_0
q_1	q_f	ϕ
q_2	ϕ	q_f
q_f	ϕ	ϕ

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

$\delta'(q_0, 1) = \delta^*(q_0, 1)$
$= (q_2, q_0)$

$\delta'((q_0, q_1), 0) = \delta^*(q_0, 0) \cup \delta^*(q_1, 0)$
$= (q_0, q_1) \cup (q_f)$
$= (q_0, q_1, q_f)$

$\delta'((q_0, q_1), 1) = \delta(q_0, 1) \cup \delta(q_1, 1)$
$= q_2 \cup \phi$
$= (q_2, q_0)$

$\delta'(q_2, q_0, 0) = \delta(q_2, 0) \cup \delta(q_0, 0) = [q_0, q_1]$

$\delta'((q_2, q_0), 1) = \delta(q_0, 1) \cup \delta(q_0, 1) = [q_0, q_2, q_f]$

$\delta'((q_0, q_1, q_f), 0) = \delta(q_0, 0) \cup \delta(q_1, 0) \cup \delta(q_f, 0)$
$= (q_0, q_1) \cup q_f \cup \phi$
$= (q_0, q_1, q_f)$

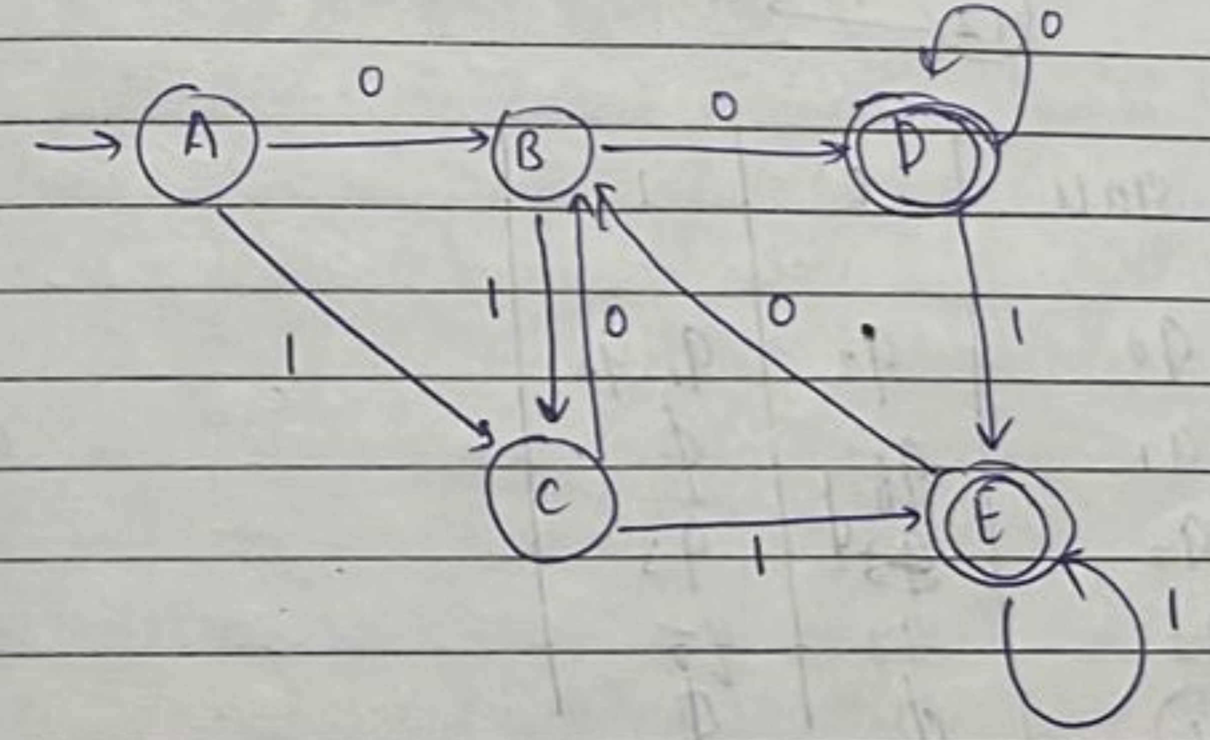
$\delta'((q_0, q_1, q_f), 1) = \delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_f, 1)$
$= q_2 \cup \phi \cup \phi$
$= (q_2, q_0, q_f)$

$\delta'(q_f, q_2, q_0, 0) = \delta(q_0, 0) \cup \delta(q_2, 0) \cup \delta(q_f, 0)$
$= (q_0, q_1) \cup \phi \cup \phi$
$= (q_0, q_1)$

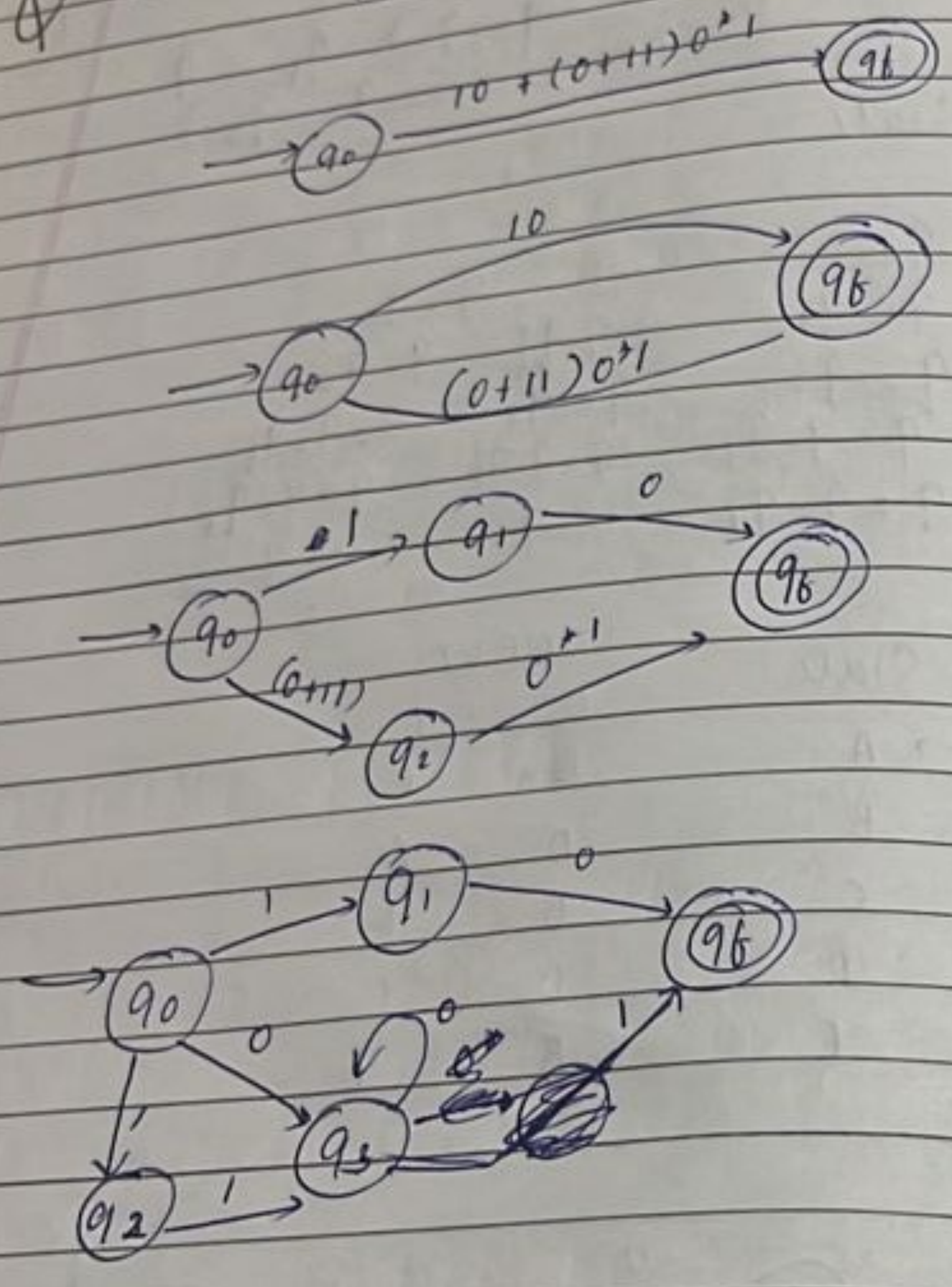
$\delta'((q_f, q_0, q_2), 1) = \delta(q_0, 1) \cup \delta(q_2, 1) \cup \delta(q_f, 1)$
$= (q_0, q_2) \cup q_f \cup \phi$
$= (q_0, q_2, q_f)$

	State	INPUT 0	1
A	$\rightarrow q_0$	$q_0 q_1$	$q_2 q_0$
B	$q_0 q_1$	$q_0 q_1 q_f$	$q_2 q_0$
C	q_2, q_0	q_0, q_1	$q_0 q_2 q_f$
D	q_0, q_1, q_f	$q_0 q_1 q_f$	$q_0 q_2 q_f$
E	$q_0 q_2 q_f$	$q_0 q_1$	$q_0 q_2 q_f$

State	INPUT 0	1
$\rightarrow$ A	B	C
B	D	C
C	B	E
D	D	E
E	B	E



Q construct the finite automata from
$10 + (0+11)0^*1$



State	0	1
$\rightarrow q_0$	q_3	q_1, q_2
q_1	q_f	ϕ
q_2	ϕ	q_3
q_3	q_3	q_f
q_f	ϕ	ϕ

$\delta'(q_0, 0) = \delta(q_0, 0)$
$= \{q_3\}$

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

$\delta'(q_3, 0) = \delta(q_3, 0)$
$= (q_3)$

$\delta'(q_3, 1) = \delta(q_3, 1) = q_f$

$\delta'[(q_1, q_2), 0] = \delta(q_1, 0) \cup \delta(q_2, 0)$
$= q_f \cup \phi$
$= q_f$

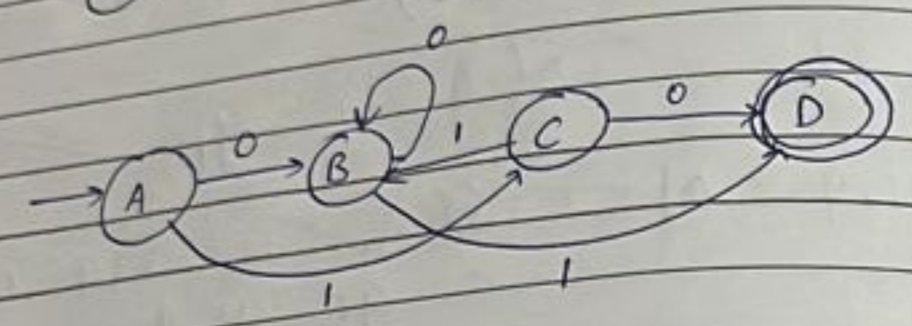
$\delta'[(q_1, q_2), 1] = \delta(q_1, 1) \cup \delta(q_2, 1)$
$= \phi \cup q_3$
$= q_3$

$\delta'(q_f, 0) = \delta(q_f, 0)$
$= \phi$

$\delta'(q_f, 1) = \delta(q_f, 1)$
$= \phi$

	STATE	0	1
A	$\rightarrow q_0$	q_3	q_1, q_2
B	q_3	q_3	q_f
C	q_1, q_2	q_f	q_3
D	q_f	ϕ	ϕ

State	INPUT 0	1
→ (A)	B	C
B	B	D
C	D	B
(D)	ϕ	ϕ



PUMPING LEMMA

Let L be a regular set, then ∃ n such that if Z is any word in L and $|Z| \geq n$ then we may write $z = uvw$ such that $|uv| \leq n$ and $|v| \geq 1$ for every $i \geq 0$, $uv^iw \in L$

where n can not be greater than the no. of states in the minimum state Finite automata ~~ac~~ accepting L.

APPLICATIONS.

- To prove that a language is not regular
- To check whether the language accepted by the finite automata ~~is~~ is finite or infinite

Steps involved in proof of L.

STEP - 1 Assume that the languages is regular then let ∃ a finite automata is M accepting L. Let x be the no. of states in finite automata.

STEP 2 Choose a string z such that $|s| \geq n$ by pumping lemma $z = uvw$ such that $|uv| \leq n$ & $|v| > 1$

STEP 3 Find any i such that $uv^iw \notin L$. This contridicts the assumption that L is Regular

Q $L = \{a^{i^2} \mid i \geq 1\}$ is not regular

1. assume L is regular then ~~∃~~ ∃ a finite automata accepting L let the no of set in finite automata is L.
2. consider $z = a^{n^2}$

$|z| \geq n$

$n^2 \geq n$

Such that by pumping ~~tmf~~ lemma

$z = uvw$

$|uv| \leq n$ & $|v| \geq 1$ & $|v| < \leq n$

3. choose ~~is~~ $i \leq 2$

so

$|uv^2w| = |uv^iw| + |v^i|$

$= z n^2 + |v^i|$ $\left(|uv^iw| = |z| = n^2 \right.$

$\leq n^2 + n$

$$\left[\text{if} \le n^2 + 2n + 1 \atop \le (n+1)^2 \right]$$

$$\boxed{n^2 < |uv^2w| < (n+1)^2}$$

because no perfect sq. can exist between n^2 & $(n+1)^2$ so uv^2w does not belong to the language L.

$$uv^2w \notin L$$

Therefore L is not regular as it ~~contr~~ contradicts the assumption.