Theory of Automata → It is an area of CS that deals with the study of abstract machines (mathematical models) as well as the computational problems that can be solved using them.

Different Names

Theory of Automata
Automata Theory
Theory of CS
Theory of computation
Computer Theory.

What is Automata? - A kind of device that acts like a computer executing a program, is a finite automata. It is the first mathematical model of how a computer functions while running a program. F.A. is a useful model for many kinds of s/w for designing & checking the behaviour of digital circuits & for verifying systems that have a finite no. of states such as communication protocals. F.A. is used in compilers.

Automata → Automatic machine (self controlled)

Eg comp., ATM, Calculator

Theory of Automata

Why we study → you don't know the speciality of it

If you get to know, you will study only Automata.

Computer works in 3 states ← I/P, processing, O/P

If " Now How comp. works it elobrated by Automata

But you can't understand its electrical or mechanical (Hw) part of computer

(1) Then this Automata will tell you that how computer works in these three states ← I/P proce O/P

without going into Hw details.

we know that,

Machines works in machine language.

(3) Design of theoretical models for machines

↓

Abstract machines

↓

(4) Course Requirement

↓

fundamental/ Core subject of computer

Compulsory subject of

* **Letters** - Character/Symbols out of which we build languages for machine. Eg:- a, b, c ---
1, 2, ---

* **Alphabet** - A set of letters, denoted by Σ.

* **String** - Concatenation of letters.
eg, aa, bb, ab

Eg $\Sigma = \{a, b\}$
$\Sigma = \{$set of key
of keybo
$= \{$set key
of calculato
$\Sigma = \{0, 1$

* **Language** - A set of strings with rules.

* **Word** - String that is permissible in the language.

Ex Make a language for a machine in which strings start with 'a' and ends with 'a' from Alphabet, $\{a, b\}$

Eg **Language** $\Rightarrow L1 = \{aa, aba, abba, ---\}$

Word $\rightarrow$ Eg $\{aa\}$

Empty String :- String that has no letter or Null string denoted by Λ, λ or ϵ. It's length is zero.

Length of String :- is the no. of letters in a string denoted by $|s|$

Example: $s = abab$

$|s| = 4$

or length(s) = 4

or length(abab) = 4.

Reverse of string :- is obtained by writing letters of string in reverse order denoted by Rev(s) or S^R or Reverse(s)

Example $s = abab$

Rev(s) = baba

Reverse(s) = baba

Power of Alphabet :- Determines that the strings made from alphabet will be of length equal to power of alphabet

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

Total no. of ~~letters~~ combinations in alphabet $= n^m$ | $2^2 = 4$

where n = no of character in the Alphabet

where m = Power of the ~~(illegible)~~ Alphabet.

Power of String :- Determines the length of string.

Eg $(bab)^2$ = babbab

(ba^2b) = ...

Ex. (2.2)
KLP (30) Determine Initial States, Final States and the acceptability of 101011, 111010.

Soln- Initial State = q_0, q_1
Final State = q_3

Acceptability of 101011 -

$q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_1 \xrightarrow{1011} (?)$ → (means no transition exists.)

$q_0 \xrightarrow{0} q_2 \xrightarrow{1011} (q_3)$ → Accepted by the transition system.

So, string 101011 is accepted.

Acceptability of 111010 -

$q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_2 \xrightarrow{1} q_2 \xrightarrow{0} (?)$

$q_0 \xrightarrow{0} q_1 \xrightarrow{10} (?)$

So, string 111010 is not accepted.

Q-(1.)
KLP(50) Table 2.1

States	Inputs 0	Inputs 1
→ q_0	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Find out acceptability of strings by M:
(a.) 10110 (b.) 11111 (c.) 000000

Sol^n:-
(a.) $\delta(q_0, \overset{\downarrow}{1}0110) = \delta(q_1, \overset{\downarrow}{0}110)$
$= \delta(q_3, \overset{\downarrow}{1}10)$
$= \delta(q_2, \overset{\downarrow}{1}0)$
$= \delta(q_3, \overset{\downarrow}{0})$
$= \delta(q_1, \overset{\downarrow}{1})$
$= \delta(q_0, \wedge) = q_0$

So, $q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_3 \xrightarrow{1} q_2 \xrightarrow{1} q_3 \xrightarrow{0} q_1 \xrightarrow{1} q_0$ is accepted by M.
Ans.

(b.) $\delta(q_0, \overset{\downarrow}{1}1111) = \delta(q_1, \overset{\downarrow}{1}111)$
$= \delta(q_0, \overset{\downarrow}{1}11)$
$= \delta(q_1, \overset{\downarrow}{1}1)$
$= \delta(q_0, \overset{\downarrow}{1})$
$= \delta(q_1, \wedge) = q_1$

So, $q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_0 \xrightarrow{1} q_1$ is not accepted by M.
Ans.

KLP(So)

(c) $\delta(q_0, \overset{\downarrow}{0}00000) = \delta(q_2, \overset{\downarrow}{0}0000)$
$= \delta(q_0, \overset{\downarrow}{0}000)$
$= \delta(q_2, \overset{\downarrow}{0}00)$
$= \delta(q_0, \overset{\downarrow}{0}0)$
$= \delta(q_2, \overset{\downarrow}{0})$
$= \delta(q_0, \wedge) = q_0$

So, $q_0 \xrightarrow{0} q_2 \xrightarrow{0} q_0 \xrightarrow{0} q_2 \xrightarrow{0} q_0 \xrightarrow{0} q_2 \xrightarrow{0} q_0$ is

✓ Ans. accepted by M.

Q(2.) KLP(So) For transition Graph given in Fig. 2.8, obtain the sequence of states for w = 000101

Transition Table for Fig. 2.8

States	Inputs		
	0	1	$\wedge$
$\rightarrow q_0$	q_0, q_3	q_0, q_1	—
q_1	—	q_2	—
q_2	q_2	q_2	q_4
q_3	q_4	—	—
(*q_4)	q_4	q_4	—

$\delta(q_0, \overset{\downarrow}{0}00101) = \delta(q_0, \overset{\downarrow}{0}0101)$
$= \delta(q_0, \overset{\downarrow}{0}101)$
$= \delta(q_0, \overset{\downarrow}{1}01)$
$= \delta(q_0, \overset{\downarrow}{0}1)$
$= \delta(q_3, \overset{\downarrow}{1})$

No Transitions possible further.

$\delta(q_0, \overset{\downarrow}{0}00101) = \delta(q_3, \overset{\downarrow}{0}0101)$
$= \delta(q_4, \overset{\downarrow}{0}101)$
$= \delta(q_4, \overset{\downarrow}{1}01)$
$= \delta(q_4, \overset{\downarrow}{0}1)$
$= \delta(q_4, \overset{\downarrow}{1})$
$= \delta(q_4, \wedge) = q_4$

So, $q_0 \xrightarrow{0} q_3 \xrightarrow{0} q_4 \xrightarrow{0} q_4 \xrightarrow{1} q_4 \xrightarrow{0} q_4 \xrightarrow{1} q_4$ is accepted by M. ✓ Ans.

Q-(3.) KLP(S0) Test whether 110011, 110110 are accepted by transition system given in Fig 2.6.

Sol^n - (a.) $w = 110011$

$\delta(q_0, \overset{\downarrow}{1}10011) = \delta(q_0, \overset{\downarrow}{1}0011)$
$= \delta(q_0, \overset{\downarrow}{0}011)$
$= \delta(q_2, \overset{\downarrow}{0}11)$
$=$ No further transitions possible

✓ Ans.

So, $w = 110011$ is not accepted by M.

(b.) $w = 110110$

$\delta(q_0, \overset{\downarrow}{1}10110) = \delta(q_0, \overset{\downarrow}{1}0110)$
$= \delta(q_0, \overset{\downarrow}{0}110)$
$= \delta(q_2, \overset{\downarrow}{1}10)$
$= \delta(q_2, \overset{\downarrow}{1}0)$
$= \delta(q_2, \overset{\downarrow}{0})$ No transitions possible further.

So, $w = 110110$ is not accepted by M. ✓ Ans.

Non-Deterministic finite Automata (NDFA/NFA)

In NFA, for a particular input symbol, the machine can move to any combination of the states in the machine. The exact state to which the machine moves can't be determined. So, it is called Non-Deterministic Automation.

Difference between DFA and NDFA -

DFA	NDFA
1. The transition from a state is to a single particular next state for each input symbol.	1. The transition from a state can be to multiple next states for each input symbol.
2. Empty string transitions are not seen in DFA.	2. NDFA permits empty string transitions.
3. Backtracking is allowed in DFA.	3. In NDFA, backtracking is not always possible.
4. Requires more space.	4. Requires less space.
5. A string is accepted by a DFA, if it transits to a final state.	5. A string is accepted by a NDFA, if at least one of transitions ends in a final state.

NDFA to DFA Conversion

(1.) Problem Statement - Let $X = (Q_X, \Sigma, \delta_X, q_0, F_X)$ be an NDFA which accepts the language $L(X)$. We have to design an equi-

-valent DFA $Y = (Q_Y, \Sigma, \delta_Y, q_0, F_Y)$ such that $L(Y) = L(X)$. The following procedure converts the NDFA to its equivalent DFA:-

(2.) Algorithm - Input - An NDFA
Output - An equivalent DFA

Step 1 - Create state table from the given NDFA.

Step 2 - Create a blank state table under possible input alphabets for the equivalent DFA.

Step 3 - Mark the starting state of the DFA by q_0 (Same as the NDFA).

Step 4 - Find out the combination of states $\{Q_0, Q_1, \ldots, Q_n\}$ for each possible input alphabet.

Step 5 - Each time we generate a new DFA state under the input alphabet columns, we have to apply step 4 again, otherwise go to step 6.

Step 6 - The states which contain any of the final states of the NDFA are the final states of the equivalent DFA.

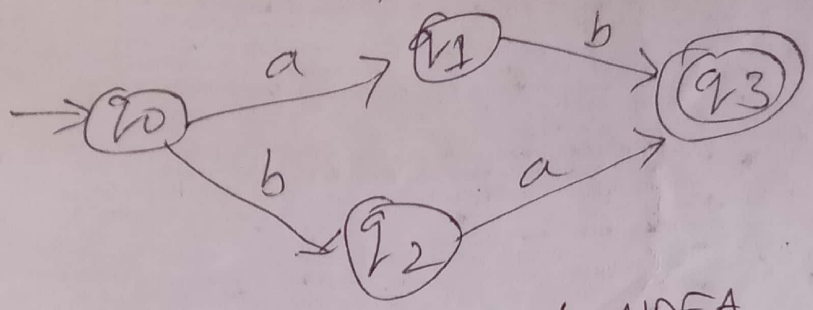
State Table for given NDFA:-

Present State	Next State	
	a=0	a=1
→ a	{a, b, c, d, e}	{d, e}
b	{c}	{e}

Q-(5.) (4)

KLP(5.1) Construct a NDFA accepting {ab, ba}, and use it to find a DFA

Sol:-



Transition Table of NDFA

State	Inputs a	Inputs b
→ q_0	q_1	q_2
q_1	—	q_3
q_2	q_3	—
(q_3)	—	—

Equivalent Table of DFA

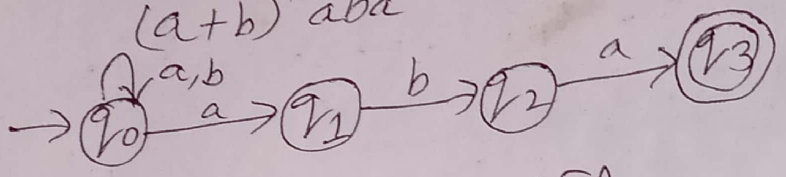
State	Inputs a	Inputs b
→ $[q_0]$	$[q_1]$	$[q_2]$
$[q_1]$	ϕ	$[q_3]$
$[q_2]$	$[q_3]$	ϕ
($[q_3]$)	ϕ	ϕ
ϕ	ϕ	ϕ

✓ Ans.

Q-(6.) KUP(51) Construct a NDFA accepting the set of all strings over {a, b} ending in aba. Use it to construct a DFA accepting the same set of strings.

Soln- set of all strings over {a, b} ending in aba.

$(a+b)^*aba$



State Table for NDFA

States	Inputs a	Inputs b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	—	q_2
q_2	q_3	—
q_3	—	—

Equivalent DFA will be

States	Inputs a	Inputs b
$\rightarrow[q_0]$	$[q_0, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_0, q_2]$
$[q_0, q_2]$	$[q_0, q_1, q_3]$	$[q_0]$
$[q_0, q_1, q_3]$	$[q_0, q_1]$	$[q_0, q_2]$

✓ Ans.

Q-(7.)
KLP(51) Construct a D.F.A. equivalent to N.D.F.A.

Transition Table for D.F.A.

State	Input 0	1	2
→[q_0]	[q_1, q_4]	[q_4]	[q_2, q_3]
[q_4]	ϕ	ϕ	ϕ
[q_1, q_4]	ϕ	[q_4]	ϕ
[q_2, q_3]	ϕ	[q_4]	[q_2, q_3]

✓ Ans.

Q-(8.)
KLP(51) State Table for NDFA (given)

State	Input 0	1
→q_0	q_0, q_3	q_0, q_1
q_1	—	q_2
q_2	q_2	q_2
q_3	q_4	—
(q_4)	q_4	q_4

State Table for DFA

State	Input 0	1
→[q_0]	[q_0, q_3]	[q_0, q_1]
[q_0, q_3]	[q_0, q_3, q_4]	[q_0, q_1]
[q_0, q_1]	[q_0, q_3]	[q_0, q_1, q_2]
([q_0, q_3, q_4])	[q_0, q_3, q_4]	[q_0, q_1, q_4]
[q_0, q_1, q_2]	[q_0, q_2, q_3]	[q_0, q_1, q_2]
([q_0, q_1, q_4])	[q_0, q_3, q_4]	[q_0, q_1, q_2, q_4]
[q_0, q_2, q_3]	[q_0, q_2, q_3, q_4]	[q_0, q_1, q_2]

State	Input 0	Input 1
$[q_0, q_1, q_2, q_4]$	$[q_0, q_2, q_3, q_4]$	$[q_0, q_1, q_2, q_4]$
$[q_0, q_2, q_3, q_4]$	$[q_0, q_2, q_3, q_4]$	$[q_0, q_1, q_2, q_4]$
$[q_4]$	$[q_4]$	$[q_4]$

✓ Ans.

Q-(9.)
KLP(51) State Table for NDFA (given)

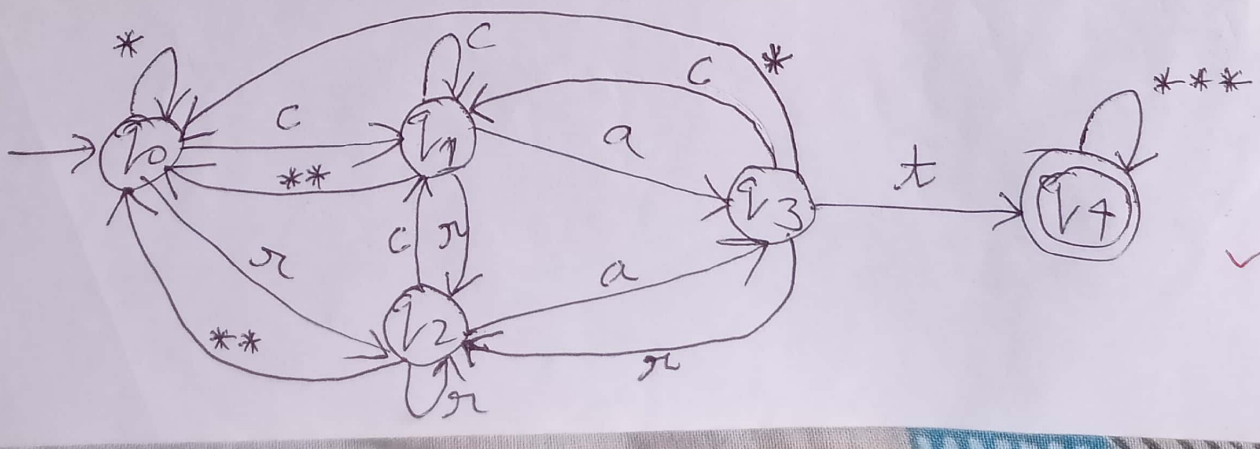
State	Input 0	Input 1
$\rightarrow q_1$	q_2, q_3	q_1
q_2	q_1, q_2	ϕ
(q_3)	q_2	q_1, q_2

Equivalent D.F.A. ~~for~~ State Table

State	Input 0	Input 1
$\rightarrow [q_1]$	$[q_2, q_3]$	$[q_1]$
$([q_2, q_3])$	$[q_1, q_2]$	$[q_1, q_2]$
$[q_1, q_2]$	$[q_1, q_2, q_3]$	$[q_1]$
$([q_1, q_2, q_3])$	$[q_1, q_2, q_3]$	$[q_1, q_2]$

✓ Ans.

Q-(10.)
KLP(52) Let $\Sigma = \{a, b, c, d, \ldots, x, y, z\} = ***$
Let $* = \Sigma - \{c, r\}$
Let $** = \Sigma - \{c, a, r\}$



✓ Ans.

* Equivalence of Two Finite Automata -

Steps to identify Equivalence

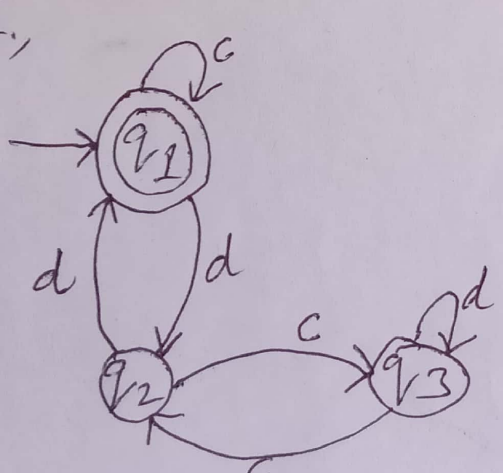
Step 1 - For any pair of states $\{q_i, q_j\}$, the transition for input $a \in \Sigma$ is defined by $\{q_a, q_b\}$, where $\delta\{q_i, a\} = q_a$ and $\delta\{q_j, a\} = q_b$.

The two automata are not equivalent if for a pair $\{q_a, q_b\}$, one is Intermediate State and the other is Final State.

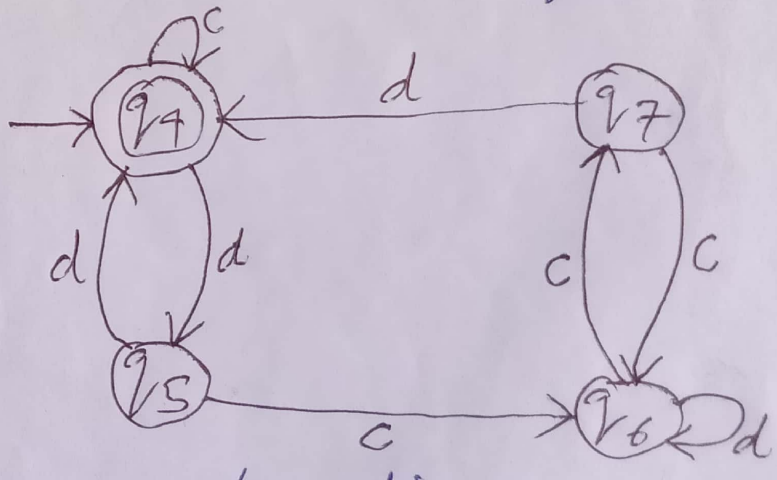
Step 2 - If Initial State is Final State of one automation, then in second automation also, Initial State must be Final State for them to be equivalent.

x - (4.15) KLP(96)

e.g.,



Automation A



Automation B

Now, check the automations A and B, Whether they are equivalent or not-

step 2 is satisfied by both automations A & B as both have their initial state as final state also.

Now, we will check whether step 1 holds true or not.

States	Inputs c	d
(q_1, q_4)	(q_1, q_4) F.S. F.S	(q_2, q_5) I.S. I.S.
(q_2, q_5)	(q_3, q_6) I.S. I.S.	(q_1, q_4) F.S. F.S
(q_3, q_6)	(q_2, q_7) I.S I.S.	(q_3, q_6) I.S I.S
(q_2, q_7)	(q_3, q_6) I.S. I.S.	(q_1, q_4) F.S. F.S.

So, It is clear that both the automation A and B are equivalent. Ans.

Ex.-(4.16)
KLP(97)

Sol^n:- Step 2 is satisfied by both the automations (a) & (b) as both have their initial state as final state also.

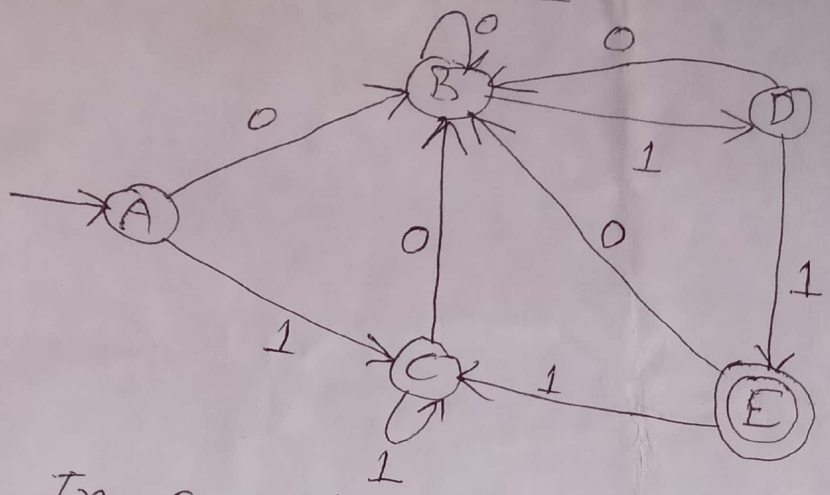
Now, we will check whether step 1 holds true or not.

States	Inputs: c	d
(q_1, q_4)	(q_1, q_4) F.S. F.S.	(q_2, q_5) I.S. I.S.
(q_2, q_5)	(q_3, q_7) I.S. I.S.	(q_1, q_6) F.S. I.S.
~~(q_3, q_7)~~	~~$(q_2, q_4 q_5)$~~ ~~I.S. F.S. I.S.~~	~~$(q_3,$~~

So, it is clear that both the automations (a) and (b) are not equivalent. ✓ Ans.

Q (1.) ... of D.F.A. -

(1)



State	Inputs	
	0	1
→A	B	C
B	B	D
C	B	C
D	B	E
(E)	B	C

Sol^n - In 0-Equivalence, we create two sets, i.e. one for final states & another for non-final states.

$\pi_0 = (\{A, B, C, D\}, \{E\})$

A, B, C are 1-equivalent but D is not 1-equivalent to them. So,

$\pi_1 = (\{A, B, C\}, \{D\}, \{E\})$

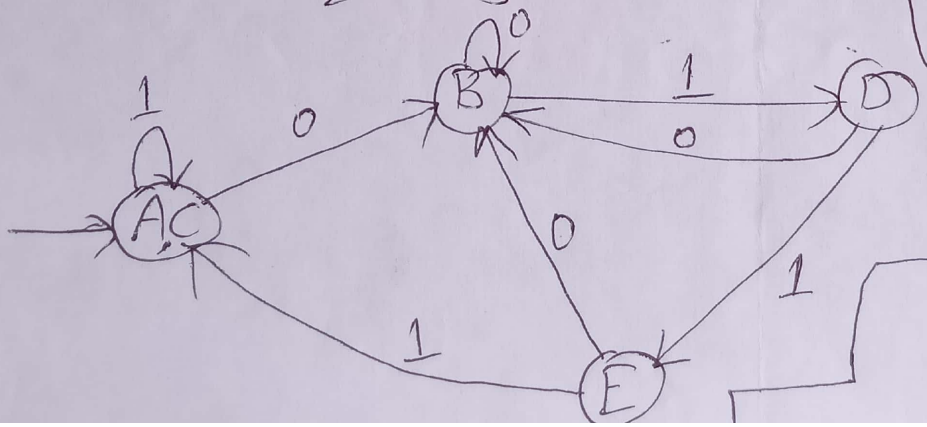
A, C are 2-equivalent but B is not 2-equivalent to them. So,

$\pi_2 = (\{A, C\}, \{B\}, \{D\}, \{E\})$

A, C are 3-equivalent to each other. So,

$\pi_3 = (\{A, C\}, \{B\}, \{D\}, \{E\})$

Here, $\pi_2 = \pi_3$

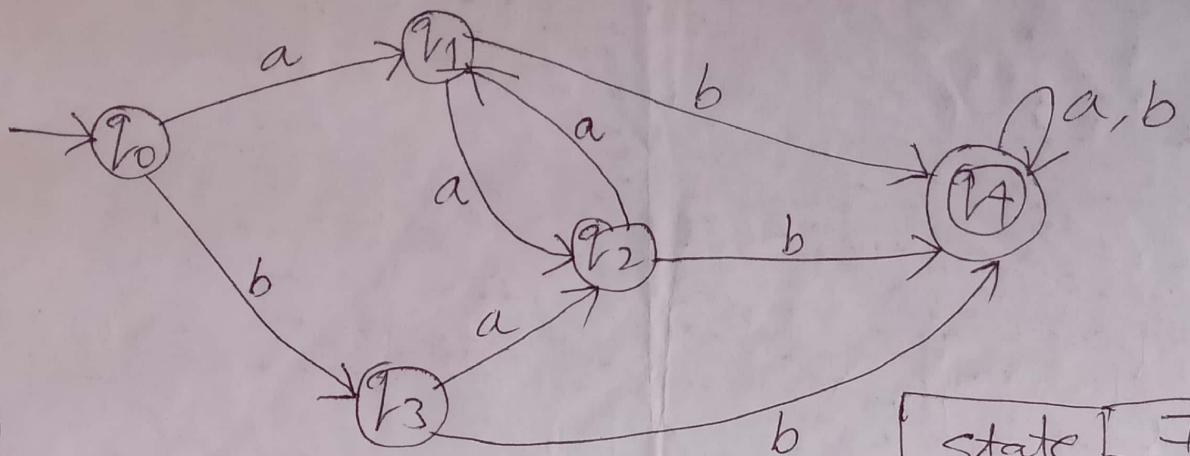


Transition Table for Minimum State Automaton

State	Inputs	
	0	1
→[A,C]	[B]	[A,C]
[B]	[B]	[D]
[D]	[B]	[E]
(E)	[B]	[A,C]

✓ Ans

Q-(2) Minimize the DFA



State	Inputs	
	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_2	q_4
q_2	q_1	q_4
q_3	q_2	q_4
(q_4)	q_4	q_4

Sol^n:- In 0-equivalence, we create two sets, i.e. one for final states & another for non-final states

$\pi_0 = (\{q_4\}, \{q_0, q_1, q_2, q_3\})$

q_1, q_2, q_3 are 1-equivalent to one-another but q_0 is not 1-equivalent to them. So,

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

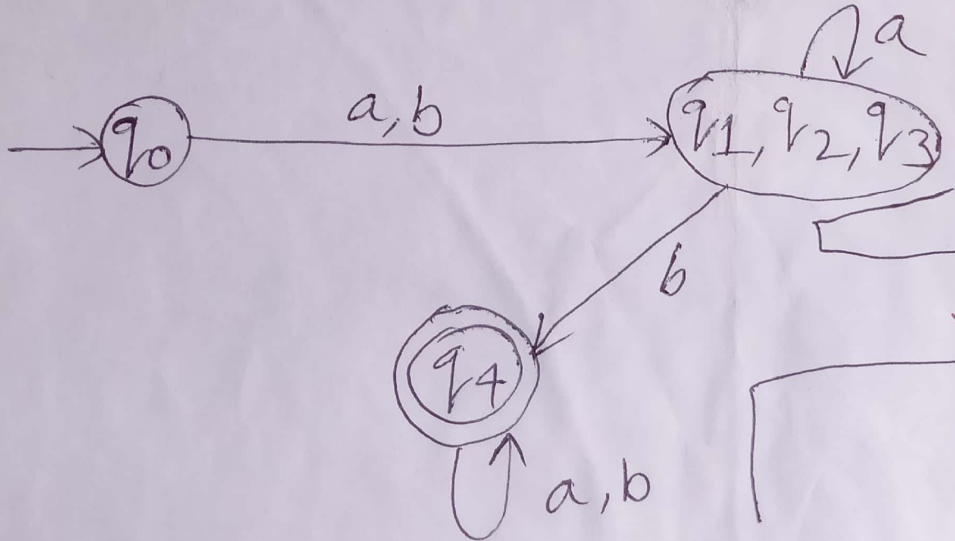
q_1, q_2, q_3 are 2-equivalent to one-another. So,

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

Here, $\pi_1 = \pi_2$

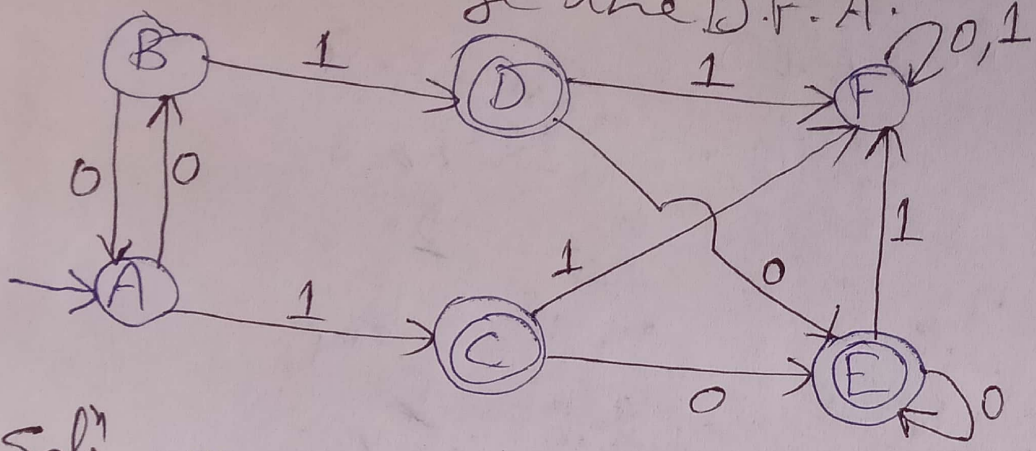
Transition Table for Minimum State Automation

State	Inputs	
	a	b
$\rightarrow [q_0]$	$[q_1, q_2, q_3]$	$[q_1, q_3, q_3]$
$[q_1, q_2, q_3]$	$[q_1, q_2, q_3]$	$[q_4]$
$([q_4])$	$[q_4]$	$[q_4]$



✓ Ans.

Q-(3.) Minimize the D.F.A.



(3)

State	Inputs	
	0	1
→A	B	C
B	A	D
(C)	E	F
(D)	E	F
(E)	E	F
F	F	F

Sol:- In 0-equivalence, we create two sets, i.e. one for final states & another for non-final states.

$\pi_0 = (\{C, D, E\}, \{A, B, F\})$

A, B are 1-equivalent to each other while F is not 1-equivalent to them. So,

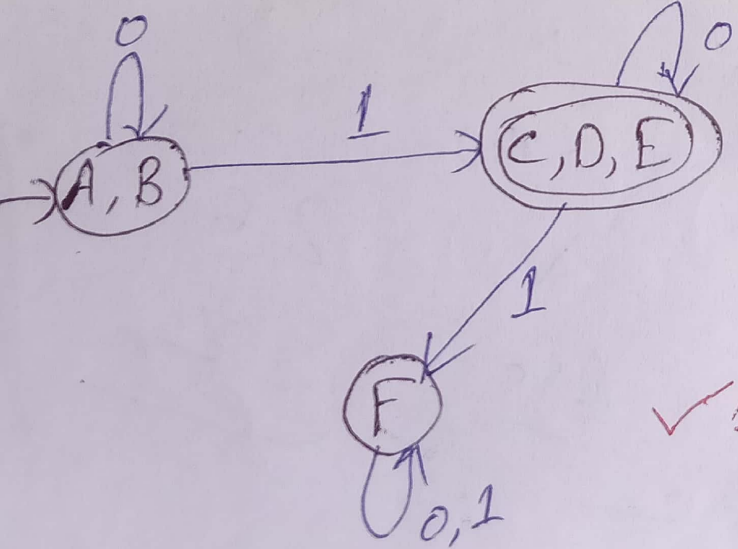
$\pi_1 = (\{C, D, E\}, \{A, B\}, \{F\})$

A, B are 2-equivalent each other. So,

$\pi_2 = (\{C, D, E\}, \{A, B\}, \{F\})$

Here, $\pi_1 = \pi_2$

State	Inputs	
	0	1
→[A,B]	[A,B]	[C,D,E]
[F]	[F]	[F]
[C,D,E]	[C,D,E]	[F]



✓ Ans.

KLP(46)

Sol:- In 0-equivalence, we create two sets, i.e. one for final states & another for non-final states.

$\pi_0 = (\{q_2\}, \{q_0, q_1, q_3, q_4, q_5, q_6, q_7\})$

q_0 is 1-equivalent to q_4 and q_6.

$\pi_1 = (\{q_2\}, \{q_0, q_4, q_6\}, \{q_1, q_7\}, \{q_3, q_5\})$

q_0, is 2-equivalent to q_4.

$\pi_2 = (\{q_2\}, \{q_0, q_4\}, \{q_6\}, \{q_1, q_7\}, \{q_3, q_5\})$

q_1 is 2-equivalent to q_7.

q_3 is 2-equivalent to q_5.

q_0 is 3-equivalent to q_4.

q_1 is 3-equivalent to q_7.

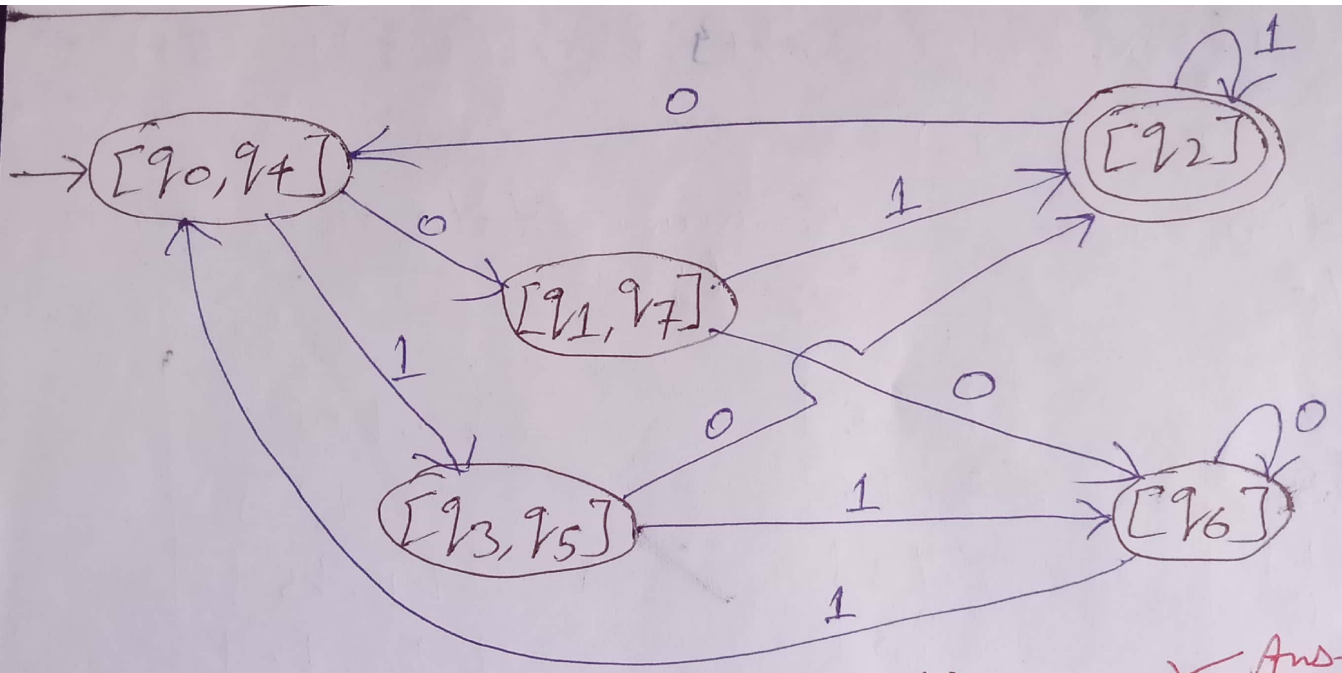
q_3 is 3-equivalent to q_5.

$\pi_3 = (\{q_2\}, \{q_6\}, \{q_0, q_4\}, \{q_1, q_7\}, \{q_3, q_5\})$

Here, $\pi_2 = \pi_3$

Transition Table for minimum state Automaton

state	Inputs	
	0	1
→ $[q_0, q_4]$	$[q_1, q_7]$	$[q_3, q_5]$
$[q_1, q_7]$	$[q_6]$	$[q_2]$
$[q_3, q_5]$	$[q_2]$	$[q_6]$
$[q_6]$	$[q_6]$	$[q_0, q_4]$
(*) $[q_2]$	$[q_0, q_4]$	$[q_2]$



Eg;- Minimum State Automation

✓ Ans-

Ex.-(2.14)
KLP(48)

Soln- In 0-equivalence, we create two sets, i.e. one for final states & another for non-final states

$\pi_0 = (\{q_3\}, \{q_0, q_1, q_2, q_4, q_5, q_6, q_7\})$

q_2 is 1-equivalent to q_1, q_5, q_6.

$\pi_1 = (\{q_3\}, \{q_0, q_1, q_5, q_6\}, \{q_2, q_4\}, \{q_7\})$

q_2 is 2-equivalent to q_4.

q_0 is 2-equivalent to q_4.

q_1 is 2-equivalent to q_6.

q_1 is 2-equivalent to q_5.

$\pi_2 = (\{q_3\}, \{q_7\}, \{q_2, q_4\}, \{q_0, q_6\}, \{q_1, q_5\})$

q_2 is 3-equivalent to q_4.

q_0 is 3-equivalent to q_6.

q_1 is 3-equivalent to q_5.

$\pi_3 = (\{q_3\}, \{q_7\}, \{q_2, q_4\}, \{q_0, q_6\}, \{q_1, q_5\})$

Here, $\pi_2 = \pi_3$

Transition Table for Minimum State Automation

State	Inputs	
	a	b
→ $[q_0, q_6]$	$[q_1, q_5]$	$[q_0, q_6]$
$[q_1, q_5]$	$[q_0, q_6]$	$[q_2, q_4]$
$[q_2, q_4]$	$[q_3]$	$[q_1, q_5]$
(*) $[q_3]$	$[q_3]$	$[q_0, q_6]$
$[q_7]$	$[q_0, q_6]$	$[q_3]$

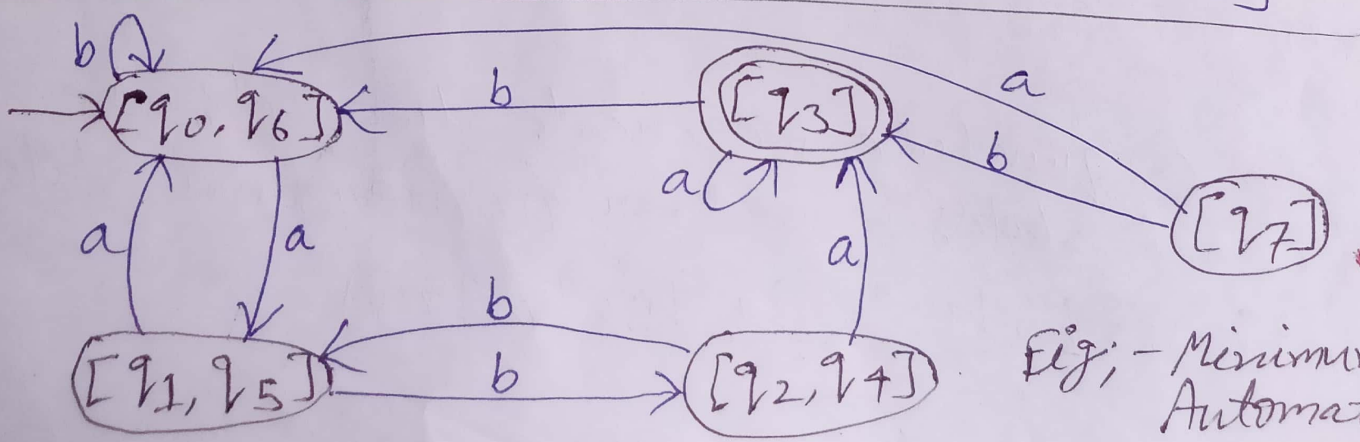


Fig:- Minimum State Automation

And:-

Q-(14.)
KLP(52)

Sol^n- In 0-equivalence, we create two sets, ie. one for final sets and another for non-final sets.

$\pi_0 = (\{q_6\}, \{q_0, q_1, q_2, q_3, q_4, q_5\})$

q_0 is 1-equivalent to q_1, q_2, q_3, q_5.

$\pi_1 = (\{q_6\}, \{q_4\}, \{q_0, q_1, q_2, q_3, q_5\})$

q_0 is 2-equivalent to q_1 & q_3.

q_2 is 2-equivalent to q_5.

$\pi_2 = (\{q_6\}, \{q_4\}, \{q_0, q_1, q_3\}, \{q_2, q_5\})$

q_2 is 3-equivalent to q_5.

q_0, q_1 & q_3 are not 3-equivalent to one another.

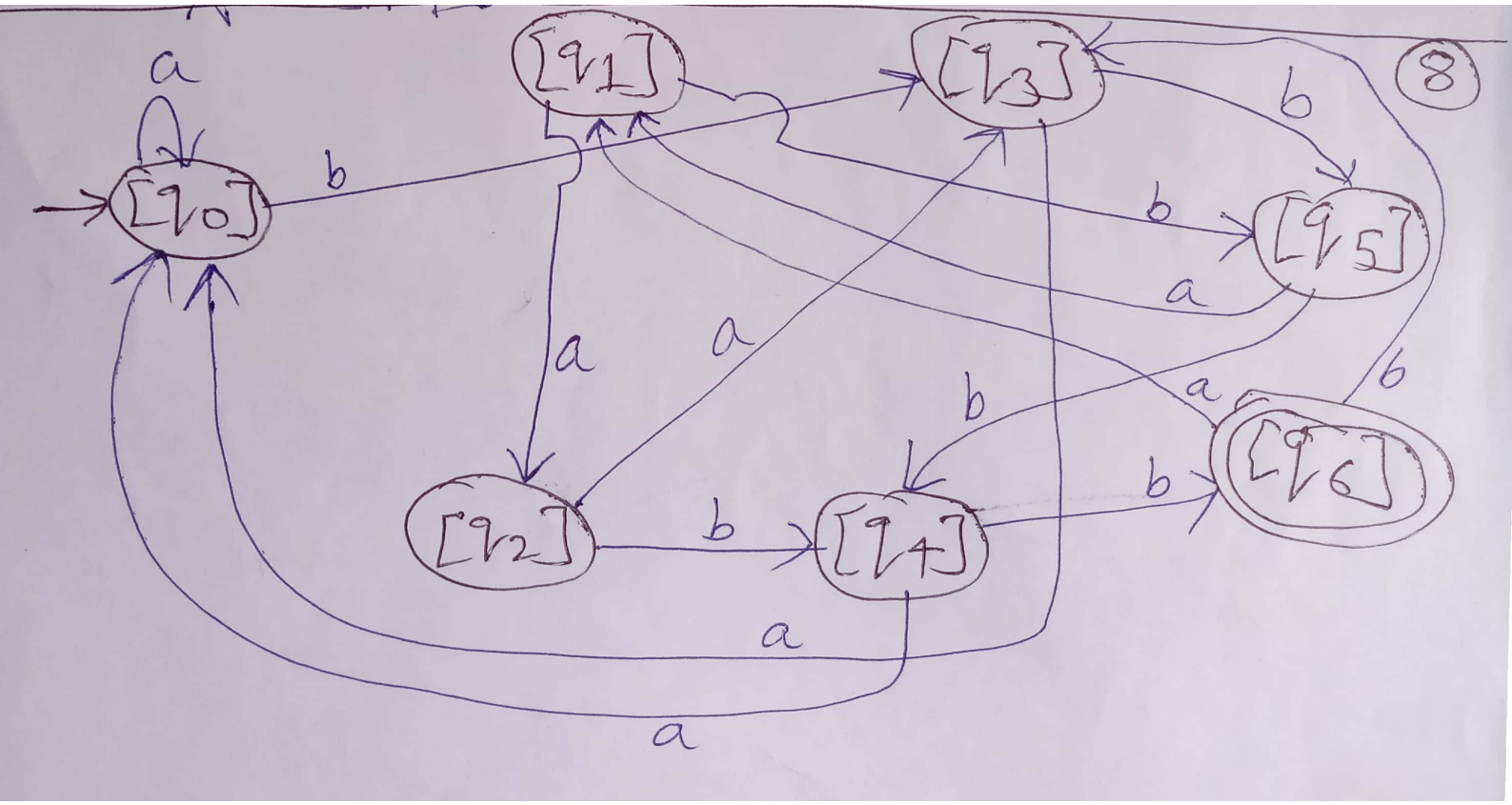
$\pi_3 = (\{q_6\}, \{q_4\}, \{q_2, q_5\}, \{q_0\}, \{q_1\}, \{q_3\})$

q_2 is not 4-equivalent to q_5.

$\pi_4 = (\{q_6\}, \{q_4\}, \{q_0\}, \{q_1\}, \{q_3\}, \{q_2\}, \{q_5\})$

Transition Table for Minimum State Automation

State	Input a	Input b
→[q_0]	[q_0]	[q_3]
[q_3]	[q_0]	[q_5]
[q_5]	[q_1]	[q_4]
[q_1]	[q_2]	[q_5]
[q_4]	[q_0]	[q_6]
[q_2]	[q_3]	[q_4]
([q_6])	[q_1]	[q_3]


[q0]
[q1]
[q2]
[q3]
[q4]
[q5]
[q6]
a
b
8