UNIT - 1

(a) Differrance b/w NFA & DFA.

NFA	DFA
(i) NFA refers to non-deterministic finite automata.	(i) DFA refers to deterministic finite automata.
(ii) In NFA, two different inputs can transit from current state to next state.	(ii) In DFA, only one input symbol transit in next state.
(iii) Every DFA can be NFA or Every NFA is not DFA.	(iii) Every NFA can not be DFA.
(iv) NFA Consume less space & hardware to impliment.	(iv) DFA takes more space & hardware to impliment than NFA.

UNIT - 2

(a) closure property of Regular Grammar :-

(i) Union :- let L1 & L2 are two regular grammar then L3 which is L1 U L2 is also be a regular grammar

(ii) Intersection :-
let L1 & L2 are the two regular grammar then $L1 \cap L2$ ~~mea~~ L3 means L3 is a Intersection of L1 & L2 . can not be regular grammar

(iii) Concatination :-
When two regular grammar Concate & make a new ~~regular~~ grammar then the new grammar will also be a regular grammar

let $L1 = \{(a^n b^n c^n),\ n \geq 1\}$
& $L2 = \{(e^m d^m g^m),\ m \geq 0\}$.

then $L3 = \text{Concate}(L1, L2)$
$= \{(a^n b^n c^n e^m d^m g^m),\ n \geq 1$
$\& \ m \geq 0\}$.

(iv) Kleene closure :-

When ~~from~~ a regular grammer, we make its closure then the new grammer will also be regular grammer.

let $L1 = \{a^n b^n\},\ n \geq 1$

$L1^* = \{a^n b^n\},\ n \geq 1$

(v) Inverse :-

When we make a inverse grammer from a regular grammer then the ~~[illegible]~~ new grammer will be also regular

let $L1 = \{0^n 1^{2n}\}\ \ n \geq 2$

$L1' = \{a^n 1^{2n}\}\ \ n \leq 2$

(c) given language

$L = \{a^n b^n,\ n \geq 1\}$

So $L = \{ab, aabb, aaabbb \text{------} \}$

let consider a pumping lemma of length 3

$p = 3$

then $S = a^p b^p = a^3 b^3 =$ aaa bbb

we got a string aaabbb

a/c to pumping lemma we divide the string into x, y & z

So case I:

$x = aa, \quad y = ab, \quad z = bb$

Case II:

$x = aaa, \quad y = bb, \quad z = b$

Condition 1:

(i) $xy^iz \in L$ where $i \geq 1$

so for case I

for $i = 2$, $\Rightarrow xy^2z$

$\Rightarrow aa(ab)^2bb$

$\Rightarrow aa\,abab\,bb \notin L$

So this string is not belong to L

So for case II

for $i = 2$, $\Rightarrow xy^2z$

$\Rightarrow (aaa)(bb)^2 b$

$\Rightarrow aaa\,bbbbb \notin L$

So this string is also not belong to L

Hence proofed the given L is not a regular language.

UNIT - 3

(a) (i) regular expression for odd no. of 1's

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

(ii) regular expression for string ending with 00.

$\Rightarrow (0+1)^* 00$.

(b) Chomsky classification of Grammar :-

According to chomsky grammars are divided into 4 types

① Type 0 :-
This is a highly ~~advanced~~ modified grammar which is accepted by turing machine.

It has consegative language & consegative grammar- ~~which~~ where all inputs accepted by this language

This grammar is also called Recursive Enumer-able grammar

(ii) Type 1 :-

This type of grammer is accepted by Linear Bound automata. It has Context Sensitive Language & Context Sensitive grammar

It is rarely used grammar.

(iii) Type 2 :-

This type of grammer is accepted by PDA. PDA refers to push down automata. in this machine we use Stack to accept inputs.

This type of grammer has Context free Language & Context free grammer

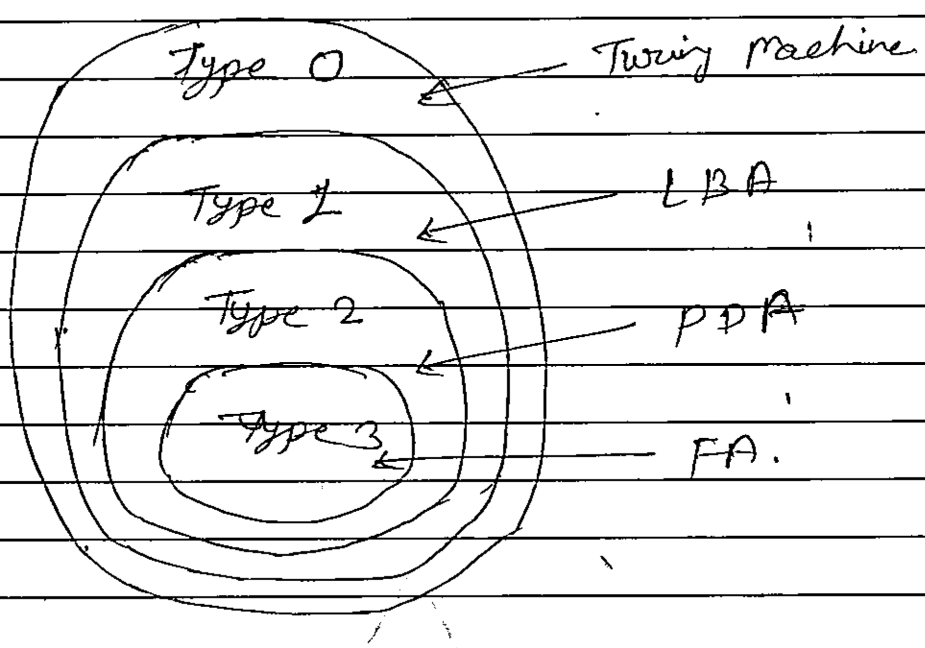
(iv) Type 3 :-

This is the lowest form of grammer. Which is accepted by finite automata.

It has input as a form of Regular Language & Regular Grammer.

In this type, there are two type of machines DFA & NFA

"Type 0 can accept all type of inputs & Languages".



[UNIT - 4]

(a) Difference b/w NPDA & DPDA.

	NPDA	DPDA.
(i)	NPDA, refers to Non-deterministic push down automata.	DPDA, refers to Deterministic push down automata
(ii)	NPDA is more powerful than DPDA	DPDA is less powerful than NPDA
(iii)	~~NDPA~~ NPDA consume minimum space as in hardware	DPDA consume more space than NPDA.

(iv)	NPDA is not easy to minimise	DPDA is easy to minimize.
(v)	all NPDA can be DPDA	but all DPDA can not be NPDA.
(vi)	In NPDA, we use two stack approach to accept the given string / input	In DPDA, we use only one stack to accept the string / input

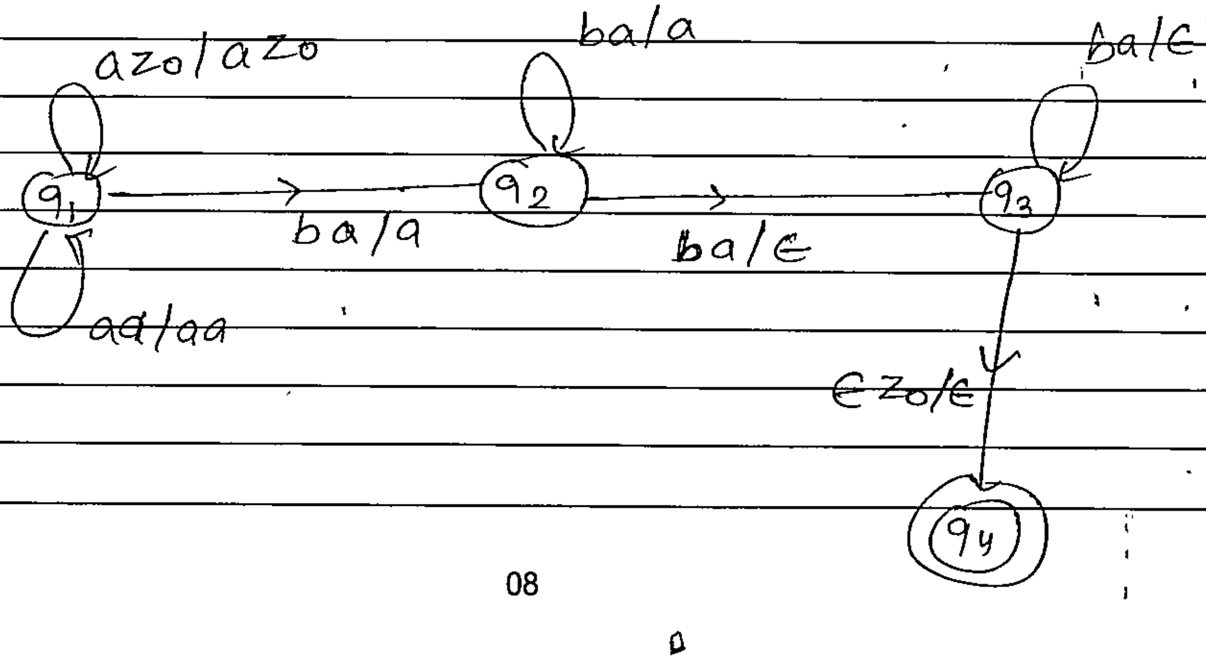
(b) given Language,

$$L = \{a^n b^{2n}\}, \text{ where } n \geq 1$$

So, $L = \{abb, aabbbb, \ldots\}$

$\therefore L = a^n b^n b^n$

PDA :-



transition for the above PDA :-

$\delta(q_1, a, z_0) \longrightarrow (q_1, az_0)$

$\delta(q_1, a, a) \longrightarrow (q_1, aa)$

$\delta(q_1, b, a) \longrightarrow (q_2, a)$

$\delta(q_2, b, a) \longrightarrow (q_2, a)$

$\delta(q_2, b, a) \longrightarrow (q_3, \epsilon)$

$\delta(q_3, b, \cancel{z_0}) \longrightarrow (q_3, \epsilon)$

$\delta(q_3, \epsilon, z_0) \longrightarrow (q_4, \epsilon)$

by this PDA ~~the~~ all the set of string $a^n b^{2n}$ will accept.

(c) given language is

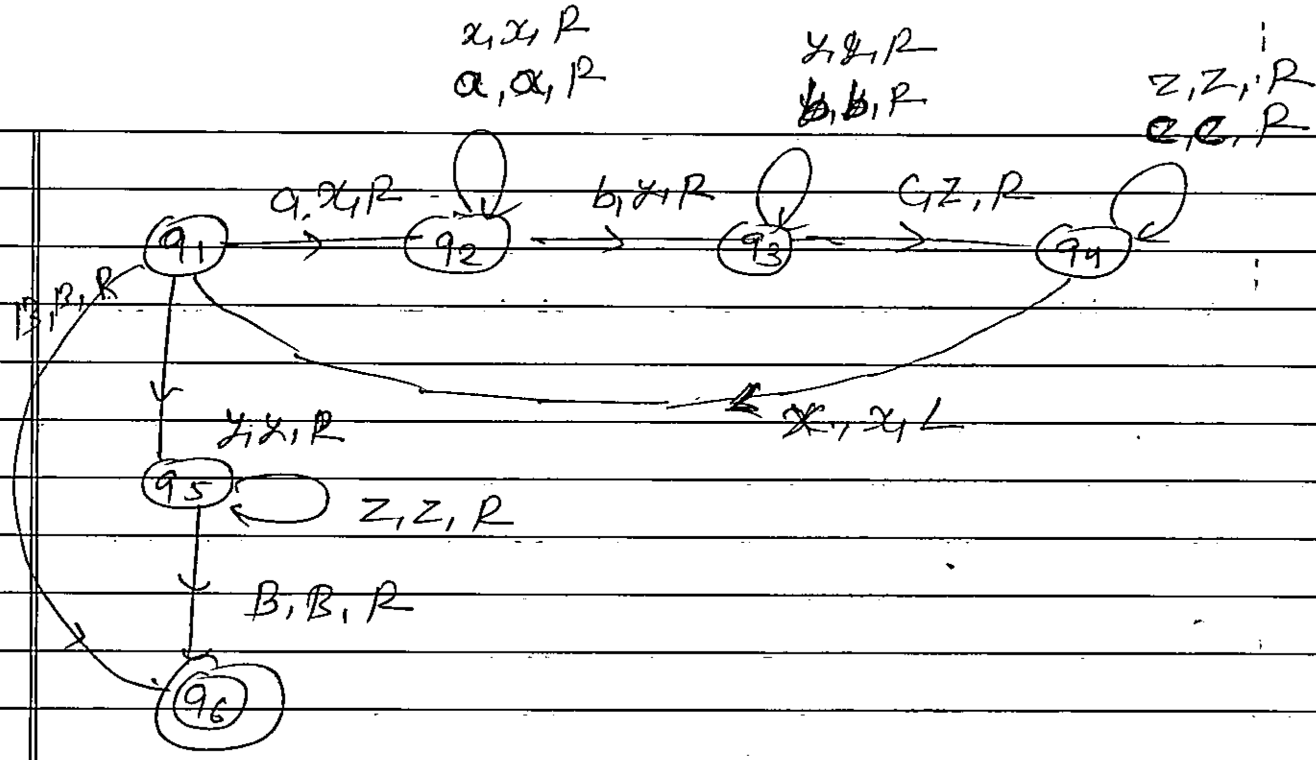
$$L = \{a^n b^n c^n\} \quad \text{where } n \geq 0$$

when $n = 1$, $S = abc$

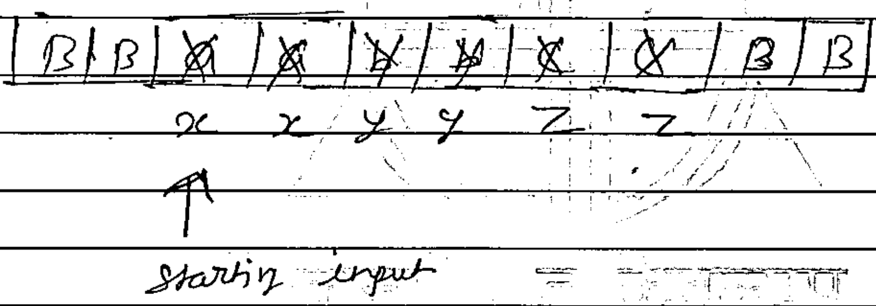
$n = 2$, $S = aabbcc$

$n = 3$, $S = aaabbbccc$

so $L = \{\epsilon, abc, aabbcc, aaabbbccc, \ldots\}$



Input tape:



So, this is the turing machine for $a^n b^n c^n$.

UNIT - 5

ⓐ **Partial function :-**

let consider a function $f(n)$ where $x_1, x_2 \ldots - - x_n$ are the elements of $f(n)$ & then again a function $g(n)$ where $y_1, y_2, y_3 \ldots \ldots y_n$ its elements

then the partial function is defined as if some of elements of $f(n)$ belongs to $g(n)$ or some of the values of $f(n)$ is exists in $g(n)$ then it is called partial function.

or some of the arguements are belongs to an another function is consider as partial function

let $f(n) = \sqrt{n^2}$, it is a total recursive function but not partial function.

Initial function :-

These function that are available at start of transitions is called initial function. these can be natural numbers as well.

There are 2 types of "Initial functions:-

(I) Over N (natural no):-

(i) Zero function:-
When any arguement of function give 0 as an answer is called zero function.

$$f(x) = 0. \quad or \quad f(1) = 0$$

(ii) Successor function:
when we add one one go in next function is consider as Successor function.

$$f(y) = y + 1.$$

(iii) preservive function:-
when each function is shown in U_n^m form is called preservive fun.

let. $f(y) = U_3^3 = \{3, 5, 7, 9\}$

$= 7$

in this function we can get direct value of that particular arguement

(N) Over $\propto$ (alphabetic form):-

(1) Null function:

When a function is equals to nil value the it is Null function.

$$f(y) = \Lambda \text{ or } \phi.$$

(b)

(I) $f(x,y) = x^{*} y$, is premitive R f.

let (I) $f(x, 0) = x * 0 = 0$

which is zero function (Initial function).

(II) $f(x,y)$ is also written as, ~~pae~~

$$f(x,y) = \phi \{ U_3^3 (x, y, f(x,y) \}$$

So this is also a form of ~~sta fir~~ initial function

Here, $f(x,y) = x^{*} y$ is belongs to initial function then this is a premitive Recursive function.

(IV) $f(x,y) = x^y$.

let (I) $f(0,y) = 0 = 0^y = 0$

mean when the x is 0 the ~~total~~ value of function is 0. which is a zero. function

(II) $f(x,y)$ is written as,

$f(x,y) = f\{ \mu_n^m (x, y, f(x,y)),$

$\mu_n^m (x, y, f(x,y) \}$

this is also a form of initial function.

therefore given $f(x,y) = x^y$ is a primitive recursive function.

UNIT - 1

(b) given Mealy Machine transaction table.

	$a=0$		$a=1$	
	S	O	S	O
$\to q_1$	q_1	1	q_2	0
q_2	q_4	1	q_4	1
q_3	q_2	0	q_3	1
q_4	q_3	0	q_2	1

for moore Machine

$q_1 \longrightarrow 1 \longrightarrow q_{11} \longrightarrow 1$

$q_2 \longrightarrow 0 \longrightarrow q_{20} \longrightarrow 0$

$\quad \searrow 1 \longrightarrow q_{21} \longrightarrow 1$

$q_3 \longrightarrow 0 \longrightarrow q_{30} \longrightarrow 0$

$\quad \searrow 1 \longrightarrow q_{31} \longrightarrow 1$

outputs.

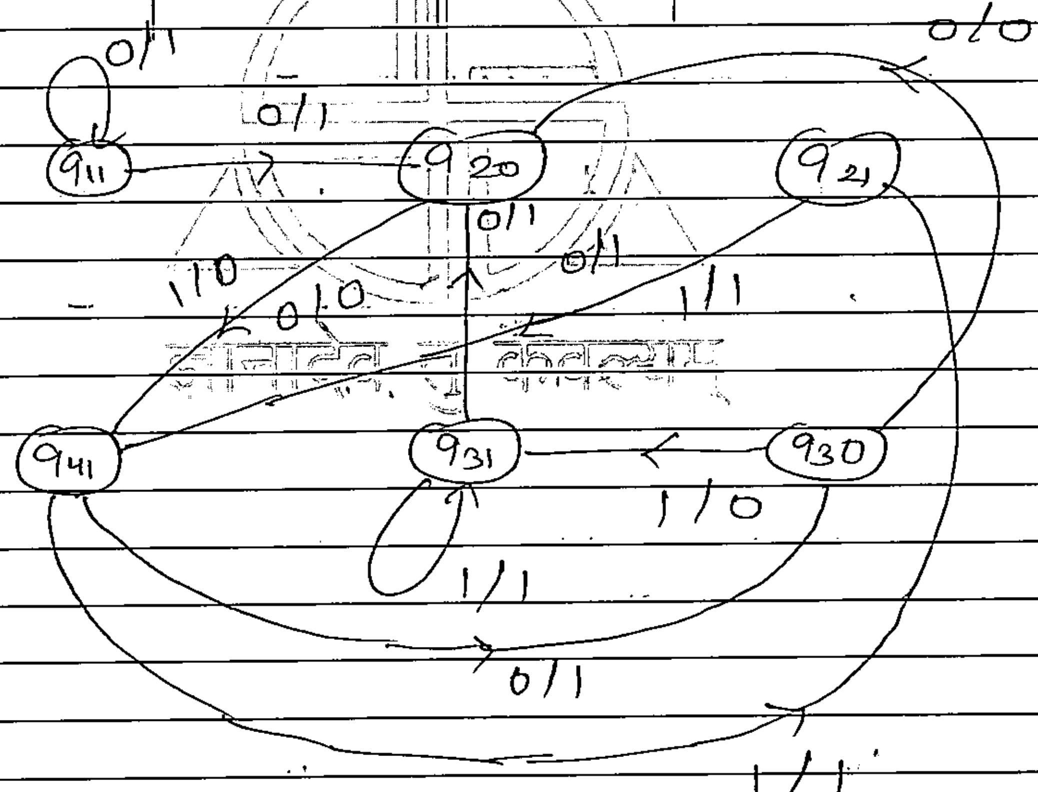
$q_4 \longrightarrow 1 \longrightarrow q_{41} \longrightarrow 1$

States.

transation table for Moore Machine

State	a=0	a=1	output
q_{11}	q_{11}	q_{20}	1
q_{20}	q_{41}	q_{41}	0
q_{21}	q_{41}	q_{41}	1
q_{30}	q_{20}	q_{31}	0
q_{31}	q_{20}	q_{31}	1
q_{41}	q_{30}	q_{21}	1

automata



Moore Machine (.

UNIT - 3

(d) given. $S \rightarrow 0B / 1A$

$A \rightarrow 0 / 0S / 1AA$

$B \rightarrow 1 / 1S / 0BB$

$w = 00110101$

LMD :- Left Most Derivative.

$S \rightarrow 0B$

$\downarrow$

$0BB$

$\downarrow \quad \downarrow$

$1 \quad 1S$

$\downarrow$

$0B$

$\downarrow$

$1S$

$\downarrow$

$0B$

$\downarrow$

1

$w = 00110101$ (derived).

RMD → Right Most Derivative.

$B \to 0BB$

$0BB$ → $0BB$ → $1\ 1$

$0BB$ → $1S$ → $0B$ → 1

$w = 00110101$ (derived)

Normal Derivation tree.

$S \to 0B$

↓

$0BB$

↓ ↓

$1\ \ 1S$

↓

$0B$

↓

$1S$

↓

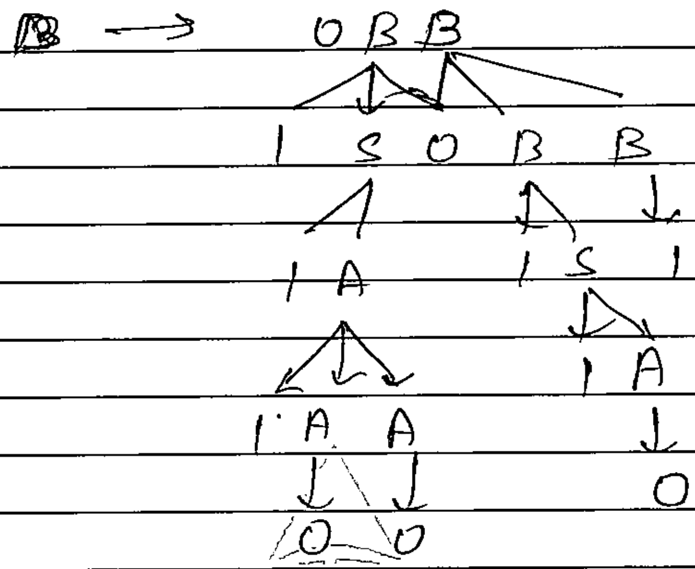
$0B$

↓

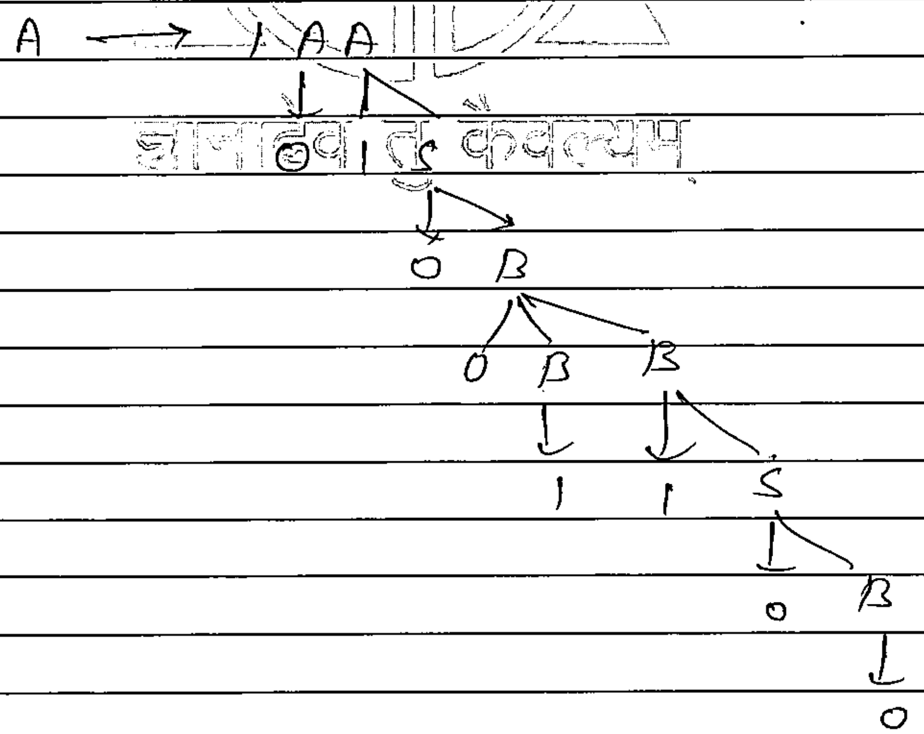
1

00110101;

LMD ⇒



RMD :-



UNIT - 2

(b) given = $(0+1)^*(00+11)(0+1)^*$

finite automata.

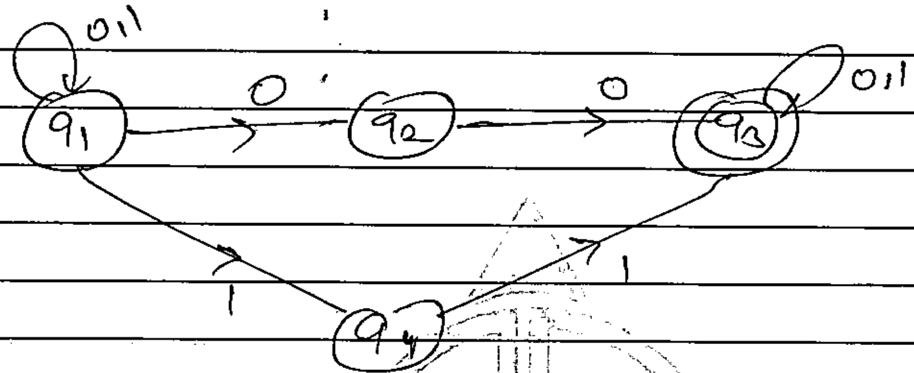


Table:

State	0	1
q_1	(q_1, q_2)	(q_1, q_4)
q_2	q_2	ϕ
q_3	q_3	q_3
q_4	ϕ	q_3

UNIT - 5

(i) ~~system~~ Space Complexity :-
In the term of Computer, How less is space taking for any algorithm is called Space Complexity.

Space complexity is not calculated for any particular system, the best complexity is calculated for beyond all systems.

In old times, we don't have enough space so we try to minimize the space but its cure is we got to time to impliment an algorithm.

In this modern era, we have infinite space.

The best space complexity is O(1). & the worst can be depend on that algo.

(ii) Time Complexity :-
The amount of time is taking by algorithm for running, is called time complexity.

It is not dependent on ~~time~~ device.

It is not calculated for any particular device, calculated for beyond the devices.

Thee best time complexity is $O(1)$.

In this modern era we have fast computer so we try to take less time for any algo.

There are 5 types of complexity

O (big Oh), o (little oh), Ω (deldos)

ω (big omega), ω (little omega)

$O(1) < O(n) < O(\log n) < O(n \log n)$

 etc.

UNIT - 1

DPA by Myhills Nerode

S	0	1
q_0	q_1	q_4
q_1	q_2	q_3
q_2	q_7	q_8
q_3	q_8	q_7
q_4	q_5	q_6
q_5	q_7	q_8
q_6	q_7	q_8
q_7	q_7	q_7
q_8	q_8	q_8

MC MHN.

	q_0	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8
q_0									
q_1									
q_2									
q_3									
q_4									
q_5									
q_6									
q_7			✓	✓		✓	✓		
q_8			✓	✓		✓	✓		

remain. , q_0q_1, q_0q_4, q_1q_2, q_1q_3,

q_4q_5, q_4q_6, ~~q_7q_7~~ q_0q_3, q_0q_2, q_0q_5

q_0q_6, q_0q_7, q_0q_8, q_1q_4, q_1q_5, q_1q_6,

q_1q_7, q_1q_8, q_2q_3, q_2q_4, q_2q_5, q_2q_6

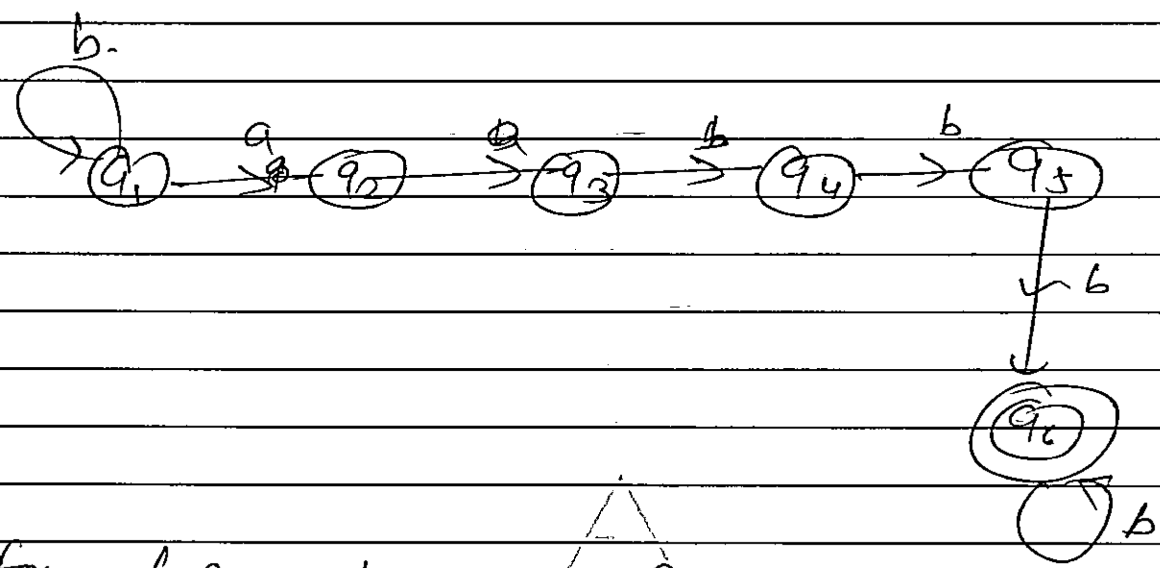
q_3q_4, q_3q_5, q_3q_6, etc.

for q_0q_1,

$\delta(q_0, 0) = q_1$ $\quad$ $\delta(q_0, 1) = q_4$

$\delta(q_0, 1) = q_2$ $\quad$ $\delta(q_\phi, q_1) = q_3$ $\quad$ X

(6) DFA



for $(w$, where $a \geq 2$, $b \geq 3)$.

A => 0S

0B

0S

01

1A

0

0BB

0BB

$a^2 b^{30BB}$

0

0S

1A

1AA

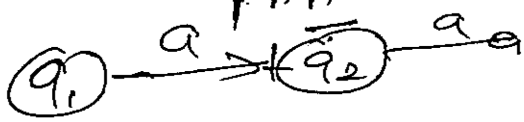
0BP

00110·1 01

a_1 a_2 a_3

bbb

0BB



B = 0BB

01

0BB

1S

0B

0S

00110·1

0BB

1S

0B

0B

S = 1A

0S

0011 · 01 01

B = 1

1S

0B

0101

1 0 or

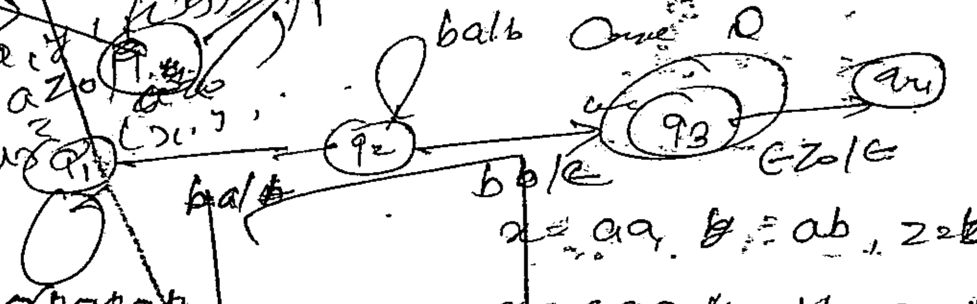
Prove a_1

10/00 101.

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

preservative fn

$n \geq 0$



$x = aa,\ y = ab,\ z = bb$

$x = aaa,\ y = bb,\ z = b$

00000

aalaa

$(q_3, b, Z_0) \rightarrow$

$(q_3, \epsilon, Z_0) \rightarrow (q_4, \epsilon)$

xyz

aa ob ob bb

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

aabbkc

aa bbbb b

L M

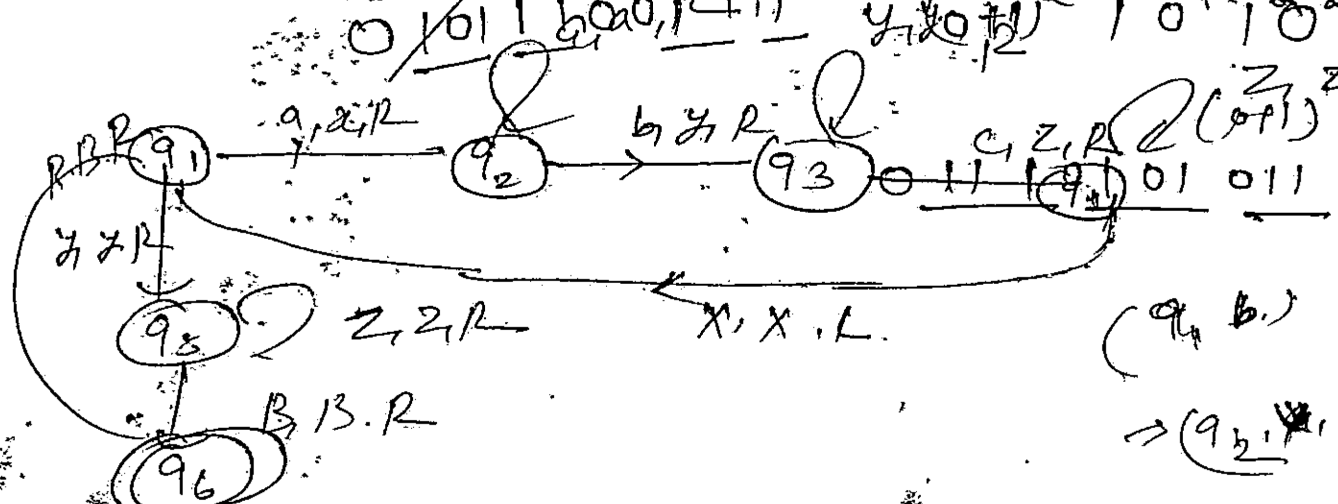
0	Cnsy (0+1)		CM
1	Cs	LB	RB
2	Cnss	CL	
3	Rec		Fin

Categories

$(0+1)^*$ 1

0010.1001

0 1011 b,a0,R 11 10^*10^*1



(q_4, b)

$\rightarrow (q_2, , R)$