ZOOM
1 INTERNET

Theory of Computation.

\# In formal language we give importance only for formation of string rather than meaning of string.

\# Σ → alphabet → finite set

$\Sigma^* \rightarrow$ set of all strings → Countable set

$2^{\Sigma^*} \rightarrow$ set of all formal language → Uncountable

S → Countable

$2^S \rightarrow$ Uncountable

\# Subset of Countable set is Countable.

REL → T.M Countable
CSL → LBA "
CFL → PDA "
RL → F.A "

Grammer :- Set of production rules which are used in generation of string is called a grammer.

eg. S → AB
A → a → terminal
B → b
$\uparrow$
non Terminal.

*1) Grammer is a generating device.

2) A grammer generates only one language.

$\alpha \rightarrow \beta$
No rules
At LHS & RHS
we can allow
terminals & Non Terminals
T_0

$\alpha \rightarrow \beta$
$|\alpha| \leq \beta$
T_1

$\alpha \rightarrow \beta$
$\downarrow$
Variables
T_2

Types of Grammers :-

1) Type 0 or R.E.

2) Type 1 or C.S.L

3) Type 2 or CFG

4) Type 3 or Regular Grammer.

$T_3 < T_2 < T_1 < T_0$ Chomsky Hierarchy

Automata → Accepting Device.

Automata :- Mathematical representation of formal Language is c/d Automata

nguage.

→ Automata is accepting Device.

3

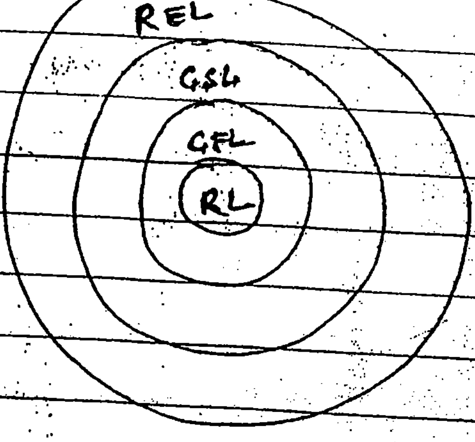
Types of Automata.

1) Finite Automata.
2) PDA
3) LBA
4) T.M. Turing M/c

Types of Formal Language:-

1) Type 0 R.E.L.
2) Type 1 C.S.L → Right side ki string Humesha Badi Hogi left se
3) 2 CFL → Left me Humesha Variable
4) 3 Regular Lang → left me Teminal

ky Hierachy.



¶ Non deterministic M/C is more powerful than deterministic.

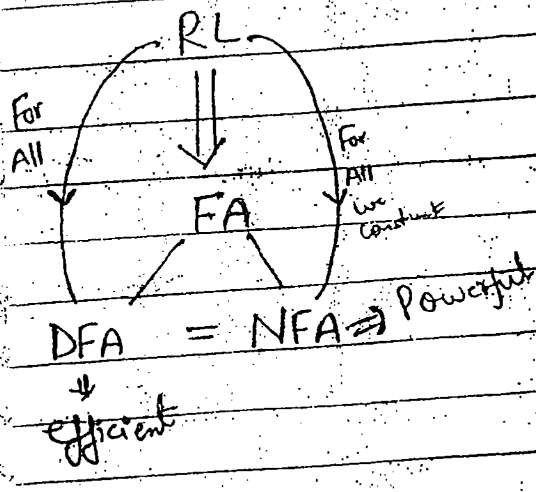
Expression power of acceptable power means the number of languages that is accepted by an Automata

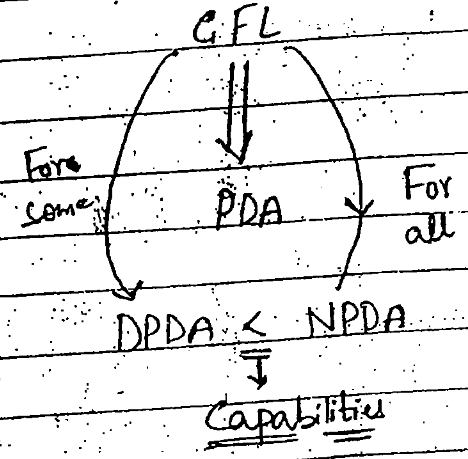
E(FA) = 1

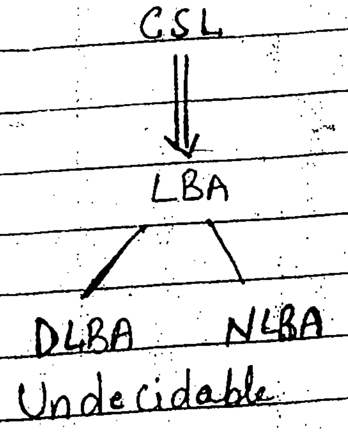
E(LBA) = 3

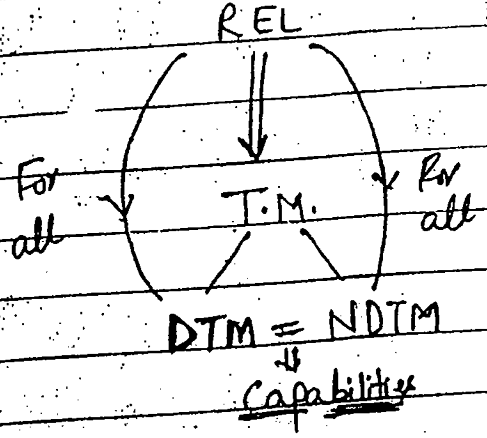
E(PDA) = 2

E(TM) = 4.









LBA { Non deterministic Turing M/c }.

Gate 2

Q :-

a) Co

b)

c)

d)

NP

Q :-

a)

Ans

Q :-

a)

b) N

c)

d)

means the
pted by

For
all

DA

u

For
all

M

'c }.

Gate 2009.

Q:- Which of following is false.

a) Capabilities of DFA & NFA same.
b) " " DPDA & NPDA is same ✓
c) " " DTM & NTM is same.
d) None

NPDA is more powerful than PDA.

Q:- Which of following cases can we construct an automata in both deterministic & Non Deterministic mode to accept same language.

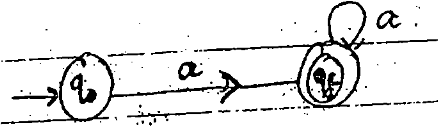
a) RL b) CFL c) CSL d) REL

Ans a, d. (2 No. R Ke case me hum Deterministic aur Non Deterministic Bana Skte hai aur 2 No ki power Bhi same Hoti Hai).

Q:- Which of following is incorrect.

a) r DFA is more efficient than NFA.
b) NPDA is more powerful than DPDA.
✓c) DTM is more powerful than NTM.
d) None.

∈ FA has limited & static Memory b/c of limitat of memory FA can't accept some of formal Language.



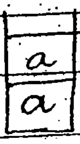
$L = \{ a^n / n \geq 1 \}$

PDA = FA + 1 stack

T.M = PDA + 1 stack = FA + 2 stacks

= FA + n stacks $\{n \geq 2\}$.

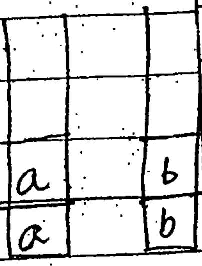
$L = \{ a^n b^n / n \geq 1 \}$ → PDA



for every a push in stack.

for every every b pop a. If stack is empty accept string.

$L = \{ a^n b^n c^n / n \geq 1 \}$ → TM



for a & b push in 2 seperate stacks

for every c pop a & b. If both stack are empty accept string.

Q:- Which of following statement is wrong.

a) PDA is more powerful than FA.

b) TM is more powerful than PAA.

✓ FA+3 stack is morepowerful than T.M.

d) None

Alphabet :- A

Called

Σ

Ex :- Σ = {

Σ = {0

STRING :-

Σ is calle

Ex :- {0,1

ω = 102

|W| = le

If ω is

No. of

is ca

denoted

Note :- l

Empty stri

Called

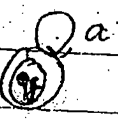
Denoted

W = ε

W ≠ ε

Date: 30/08/10

b/c of limitation
come of formal



Alphabet :- A non empty set of symbol is called an Alphabet & denoted by Σ.

Ex :- Σ = {a, b, ... x, y, z}   Σ = {0, 1, 2, 3 ... 9}
Σ = {0 - 9, a - z}.

ww {n≥2}.

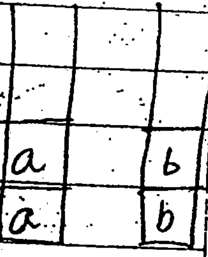
STRING :- Sequence of symbols from alphabet Σ is called a string.

Ex :- {0, 1}   ω = 011, 1011, 101010.   i.e. any comb. of 0 & 1.
ω = 102, 3012 Not string.

stack is empty

|W| = length of string 3, 4, 6.

If ω is any string over alphabet Σ the No. of symbols involved in string W is called Length of string & denoted by |W|.



Note :- |W| ≥ 0

is wrong.

Empty string :- A string of length 0 is called empty string.

FA.
PDA.
than T.M.

Denoted by ∈ (epsilon or Null).

W = ∈ ⇒ |W| = |∈| = 0   empty.
W ≠ ∈ ⇒ |W = |∈| ≠ 0

Page No.
Date : / /

SUBSTRING :- U, W be the string from alphabet Σ = {a..z} then U is said substring of W if U is contained in W.

W = {R,AHUL}   U = {HUL}.

- Every string is substring to itself.
- ε is substring for every string.

If U is substring of W then

$$|U| \le |W|$$

Example :- W = ACE

u → 0 = ε
Non trivial [1 = A, C, E
2 = AC, CE]
3 = ACE
] Trivial

GATE

ε
Non Trivial [G, A, T, E
GA, AT, TE
GAT, ATE]
GATE
] Triv

↓
ε aur w particular string ko hi lete hai.

Types of Substring :- A string can have 2 types of substrings :-

1) Trivial / Improper :- If W is any string from alphabet Σ then substring W itself & ε is called Trivial / Improper Substring.

W

b) Non
string
of

ACE

u = 0 = ε
1 = A, C, E
2 = AC, C
3 = A C E

If
K

a) No.
b) No.
c) Non

PREFI
Sepr

W = AC

Suffix:
C

~~m~~.
U is
is Contained
- }.

b) Non Trivial / Proper :- If W is any string from Σ then any substrings of W other than W & ∈ is Called Non Trivial.

ACE GATE

f

u = 0 = ∈ 1
1 = A, C, E 3
2 = AC, CE 2
3 = ACE 1

$\Sigma 3$

$\Sigma 3 + 1$

If W is any string with distinct Symbol & |W| = n then.

a) No. of Substrings = $\Sigma n + 1$ = $\frac{n(n+1)}{2} + 1$

b) No. of Trivial Substrings = 2 (∈ & itself).

c) Non Trivial = $\Sigma n - 1$

C, E
IT, TE
ATE
∈
Triv

or string ko

w 2 types

PREFIX :- Sequence of starting or leading Symbols is Called prefix

W = ACE :- ACE, CE, E, ∈

string from
itself &
or Substring

Suffix :- Sequence of ending & trailing symbol is Called Suffix.

ACE, CE, E, ∈

If $|w| = N$ then No. of perfix & suffix is $(n+1)$.

Trivial substring is both suffix & prefix

POWER of alphabets :- If Σ is any alphabet then set Σ^k Contains all string of length k.

$\Sigma^k = \{w \mid |w| = k\}$.

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

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

$\Sigma^2 = \Sigma \cdot \Sigma = \{00, 01, 10, 11\}$.

$\Sigma^3 = \Sigma \cdot \Sigma \cdot \Sigma = \{000, 001, 010, 110, 100, 101, 110, 111\}$.

If Σ is any alphabet then.

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \ldots$$
$$= \epsilon \cup \{a, b\} \cup \{aa, ab, ba, bb\} \ldots$$

$$\Sigma^* = \bigcup_{i > 0} \Sigma^i = \{w \mid |w| \geq 0\}$$

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \ldots \cup \Sigma^\infty$$
$$= \bigcup_{i \geq 1} \Sigma^i_{i \geq 1} = \{ w \mid |w| \geq 1\}$$

Σ^+ c

$\Sigma^* =$

$\Sigma^+ =$

ANGUAGE :-

Σ is

ex:- Σ

$L = \Sigma^*$

$L = \Sigma^+ =$

$L = \{0^n$

$L = \{0^n 1^n$

$L = \{0^n . 1^n$

NOTE :-

1) If Σ Univ

2) If L is any

rfix & Suffix is) $\Sigma^+ \subset \Sigma^*$

$\Sigma^* = \Sigma^+ \cup \{\epsilon\}$. Kleen Closure = including

ffix & prefix $\Sigma^+ = \Sigma^* - \{\epsilon\}$. +ve closure = exculinding ϵ

any alphabet LANGUAGE :- Colletion of string from the alphabet Σ is Called Language.

length K. ex:- $\Sigma = \{0, 1\}$.

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

$L = \Sigma^+ = \{0, 1, 00, 01, 10, 11 \ldots\}$.

$L = \{0^n \mid n \geq 1\} = \{0, 00, 000, 0000 \ldots\}$.

01, 110, 111}. $L = \{0^m 1^n \mid m, n \geq 0\}$

$L = \{0^n 1^m / m = n\}$.

n.

4. . . . .

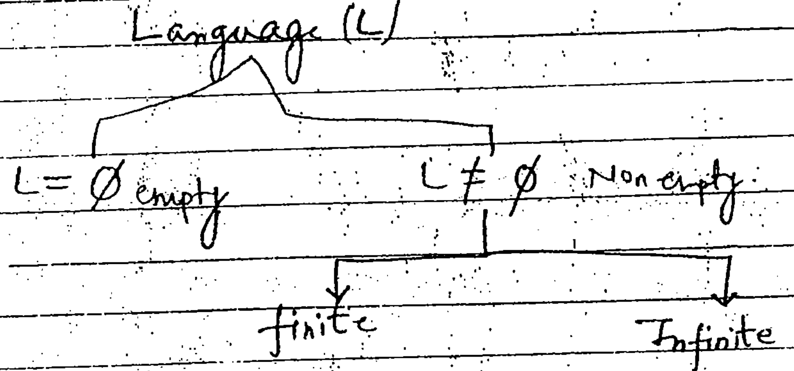
NOTE :-

, ba, bb}. . . 1) If Σ is any alphabet Σ^* is Called Universal Language.

2) If L is any language over alphabet Σ then

$L \subseteq \Sigma^*$

3)



Empty
R.L
C.F.L
C.S.L
R.E.L

Empty Language :- The language that doesn't contain any string even empty string (ϵ) is called empty language.

$L = \{\ \} = \emptyset \Rightarrow |L| = 0$

F.A.

Non Empty ([illegible]) :- Language that contain atleast 1 string is called Non empty Language.

P.D.A

T.M.

$L = \{\epsilon\}$. $|L| = 1$

$L = \{a\}$. $|L| = 1$

$L = \{a^n / \geq 1\}$ $|L| = \infty$

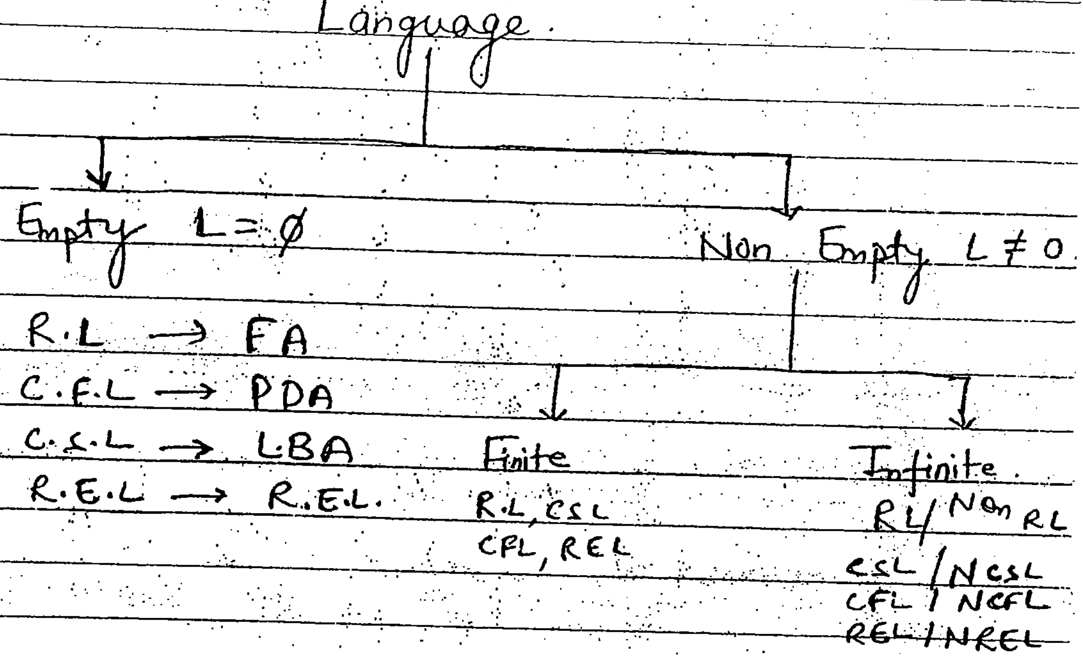
Finite Language :- Contains finite No. of things where length of string is finite.

Infinite :- It contains infinite no. of things where length of each string is finite is called Infinite Language.

pty

finite

t doesn't
npty string (ε)
e.



F.A → Acceptor / Transducer

ntain atleast
pty Language.

P.D.A → Acceptor

T.M. → Acceptor / Transducer / Generator.

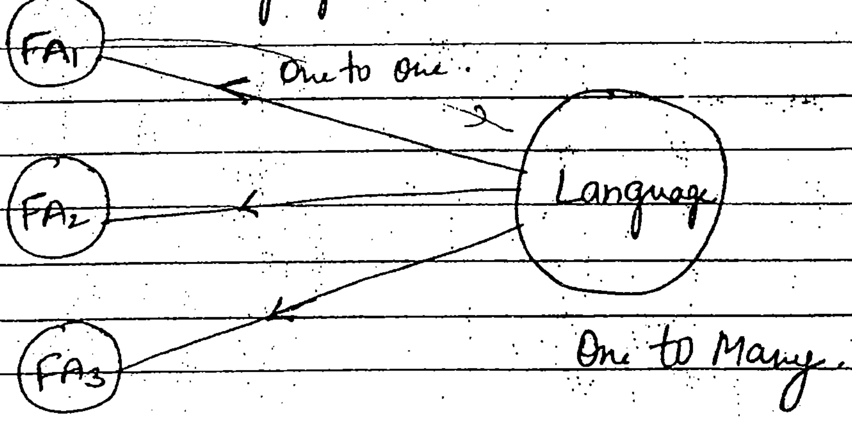
hings where

ings where
finite
uage.

}.

Page No.
Date: / /

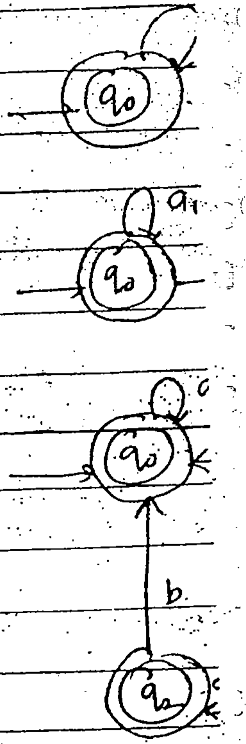
F.A. 1 hi language Ko accept kr rha hai



Lekin 1 language Ko Bahut sare FA accept kr rhe hai

Pehle ka Baad mein

5. F.A. in
X all
accepts



7. Every F

8. A Language
i.e. F

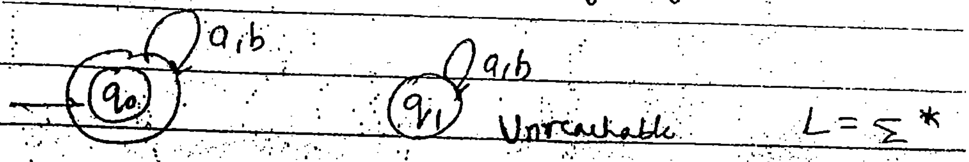
9. F.A w
Unreach

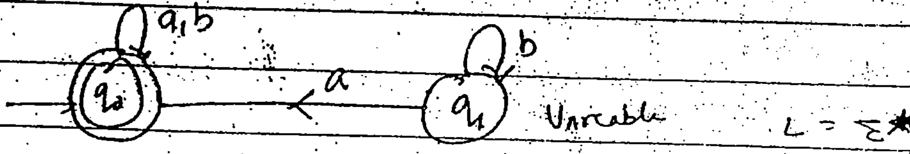
10. Every la
Minimal

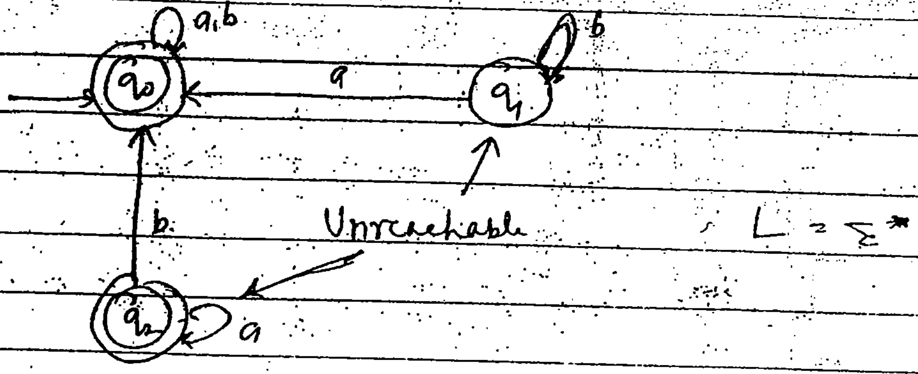
Page No.
Date: / /

Pehle ka baaki hai

; F.A. in which Initial ~~all~~ states is final state & all non final state is Unreachable accepts Universal language Σ*.







Kr rha hai

7. Every F.A. accepts only One Language.

uage)

3. A Language can be accepted by more than 1 F.A. i.e. F.A. is not Unique.

to Many.

3 F.A which is free from equal & Unreachable state is Called Minimal FA

re FA accept

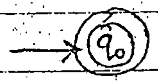
0. Every language is accepted by only One Minimal F.A. i.e. Minimal F.A. is Unique.

31/8/2010

Construction of F.A. for Finite Languages :-

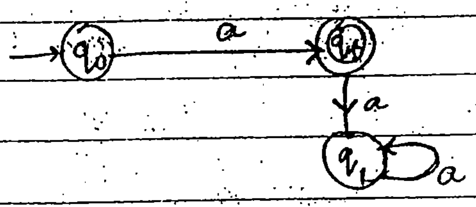
While constructing F.A we have also take cr of Dead state also.

a) $L = \{\epsilon\}$



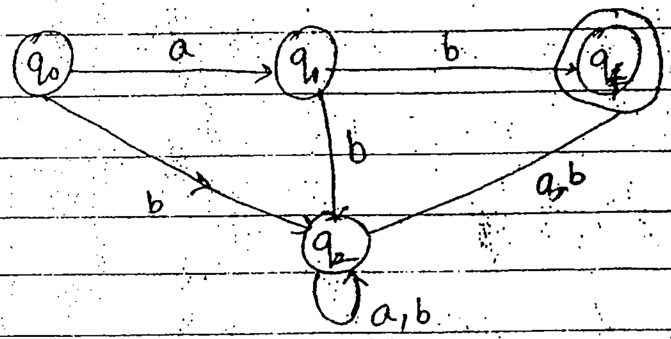
b) $L = \{a\}$ $\Sigma = \{a\}$.

$\Sigma^* = \{\epsilon, a, aa, aaa, \ldots$



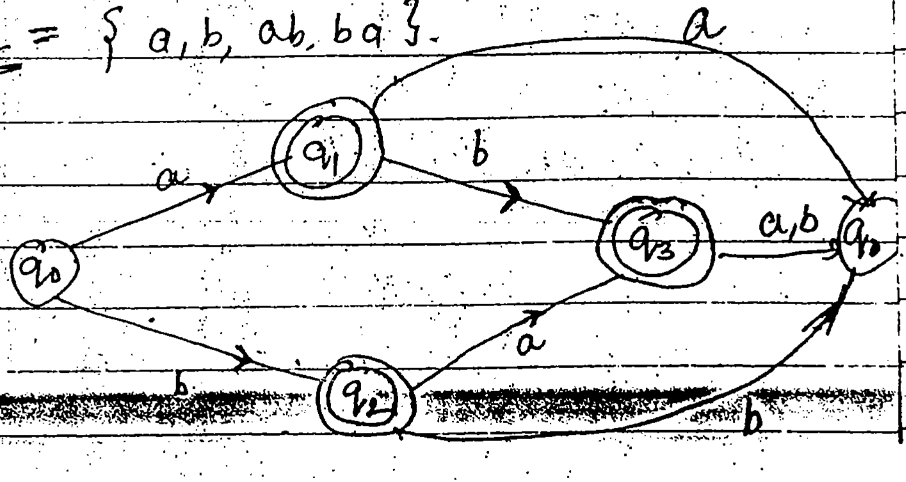
c) $\Sigma = \{a, b\}$. $\Sigma^* = \{\epsilon, a, b, ab, ba, \ldots$

$L = \{ab\}$.



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

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



Finite Autor

F.A without

DFA

(E

Mathematic

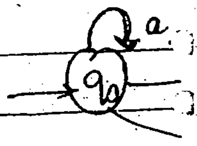
Regular

F.A i~~s~~

state Inp

$\delta: Q \times \Sigma$

$\Sigma = \{ 0$



$\delta(q_0, a)$

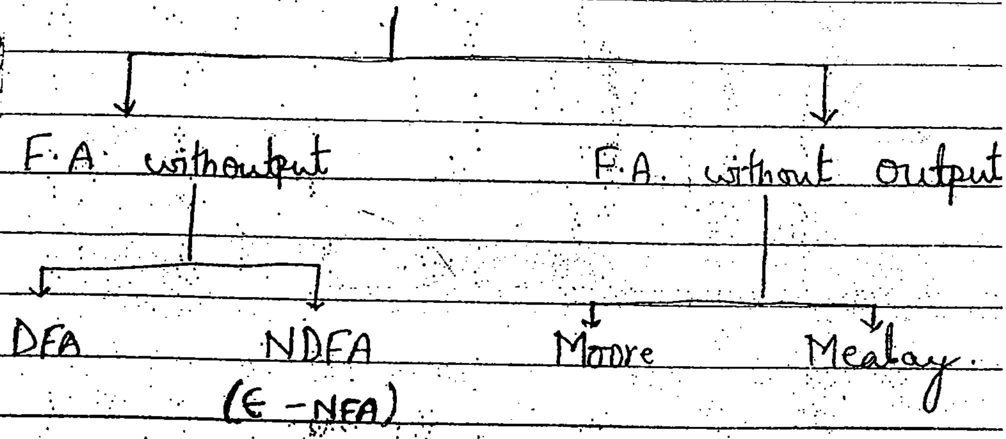
$\delta(q_0 b)$

31/8/2010 Theory of Computation.

inite Languages :-

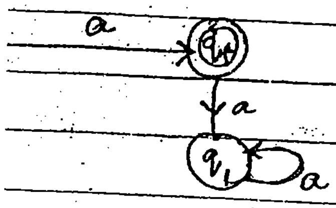
are also taken of

Finite Automata :-



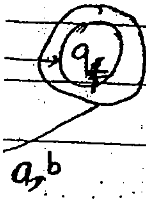
} $\Sigma = \{a\}$.

$\epsilon, a, aa, aaa, \ldots$



$a, b, ab, ba, \ldots$

Mathematic representation of regular Regular language is called F.A.



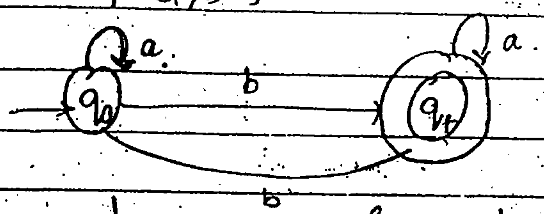
F.A is a 5 tuple $\{Q, \Sigma, \delta, q_0, F\}$

Q — set of all state; Σ — I/P alphabet; δ — Transition fn; q_0 — initial state; F — Final st

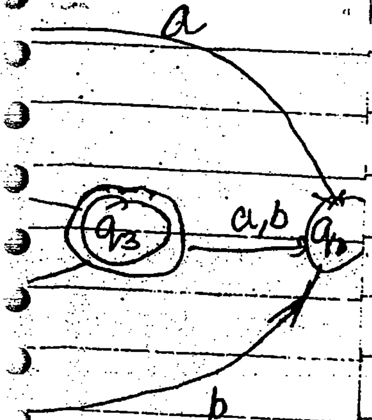
state Input → state

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

$\Sigma = \{a, b\}$ $Q = \{q_0, q_f\}$



$F = \{a, b\}$.



$\delta(q_0, a) = q_0$ $\delta(q_f, a) = q_f$ $q = q_0$ $q_f = q$

$\delta(q, b) = q_f$ $\delta(q_f, b) = q_0$.

Note :-

F.A. has any No. of final state.

By default F.A is DFA.

F.A. is a physical device which recognize states.

F.A. has only One initial state.

F.A Can be Cons. with & without final states {0, 1, 2 .... ∞}.

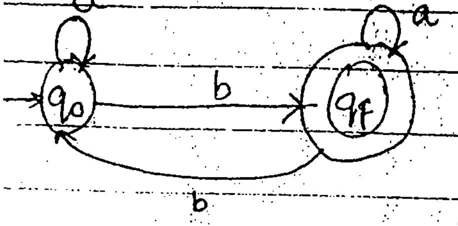
DFA is a complete system which respond for every I/P symbol, without defining transition for each & every I/P symbol & each & every state.
Jaruri nhi hai Ki har I/P Ke liye transition ho.

No. of transitions of state = |Σ| = No. of I/P symbol

★ After taking input symbol from Σ DFA moves to exactly One state.

Representation of F.A:-

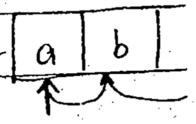
a) Transition Diagram.

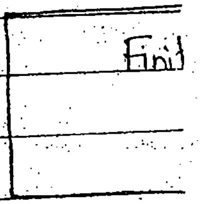


b) Transition Table.

δ	a	b
$\rightarrow q_0$	q_0	q_f
q_f	q_f	q_0

Architecture





F.A has

1. Input
2. Taple
3. Finite C

1 a) Divided
Capable

) At any
only 1

2) Tape He
tape start

Every time
& move
Right
But Not

ial state.

recognize states
nitial state.
& without final

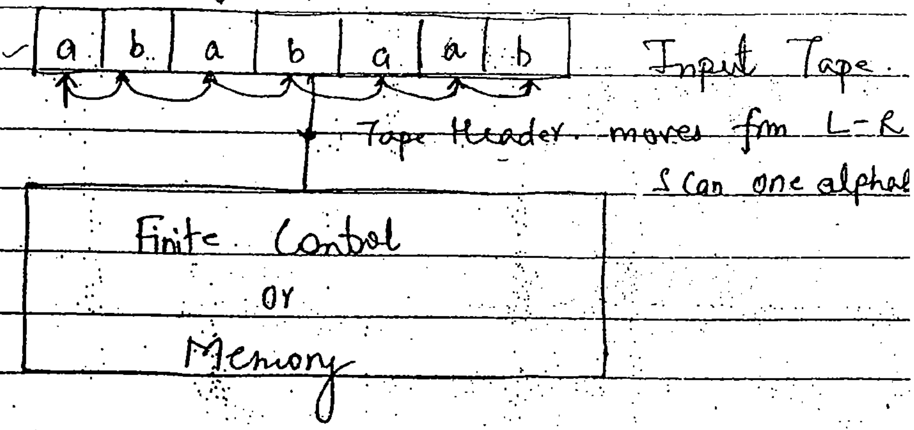
which respond
l, without
h & every I/P
very state.
e. give transition to.
|Σ| = No. of I/P symbol

from Σ DFA
one state.

Transition Table.

a	b
q0	qf
kf	q0.

Architecture of F.A. (Block Diagram).



F.A has 3 physical components.

1. Input Tape.
2. Taple Header.
3. Finite Control / Memory.

1 a) Divided into cells & each cell is Capable of holding only 1 symbol.

) At any point of time i/p tape contains only 1 I/P string.

2) Tape Header :- Reads I/P symbol from I/P tape starting from leftmost end.

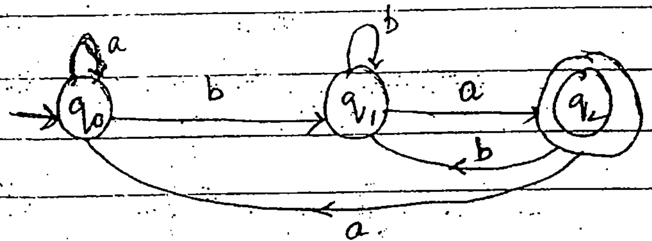
Every time it will read only 1 I/P symbol & moves exactly one cell towards Right side, i.e movement is from L-R But Not from R-L. b/c of this Limitation F.A. can be used as scanner not as Writer.

3) Memory / F.C. → It take care of transition of F.A. movement of F.A. depends on I/P symbol from Σ.

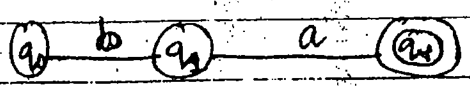
The memory of FA is distributed in form of final & it is static & limited, hence it can't recognized arbitrary large strings.

4 Acceptance by F.A. :-

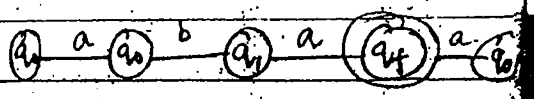
$\Sigma = \{a, b\}$ $\Sigma^* = \{\epsilon, a, b, ab, ba\}$.



a) W = aba.

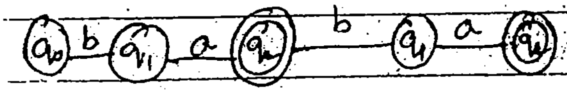


ⓑ W = abaa.

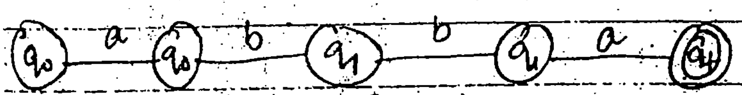


c) W = baba

Accepted



d) W = abba. [redacted] accepted.



L (F.A) = { w / w ends with ba }.

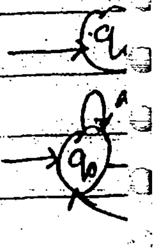
1) Let x is
Correspondi
transition
& ends
X is

2) Set o
Krta
bolte

NOTE :-

1. F.A. a

2. F.A.
empty



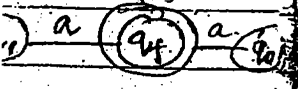
Jisme fi
bhi is

of transition
depends

in form of
limited, hence
large strings.

a}.

baa.



cepted.

ba}.

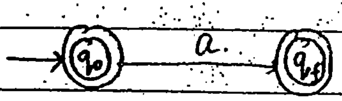
1) Let x is any string over alphabet Σ Corresponding to X if there exist a transition path which states @ initial & ends in final state then then string X is accepted by F.A.

2) Set of all string Jo F.A accept Krta hai usko Language of F.A bolte hai = $\{ n / \delta(q_0, n) = F's \}$.

NOTE :-

1. F.A. accepts every string ∈ if inital = Final

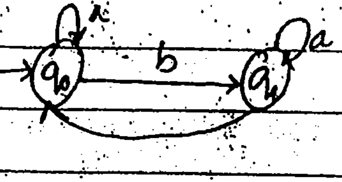
$L = \{ \in \}$



$L = \{ \epsilon, a \}$.

2. F.A. without final states accepts empty language i.e. $L = \emptyset$

$L = \emptyset$



$L = \emptyset$

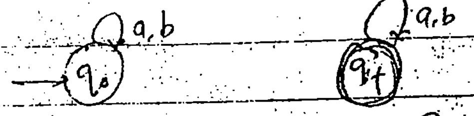
Jisme final states nhi hoti hai wo koi bhi ie $\{\emptyset\}$. language accept ni krta hai.

Minimal

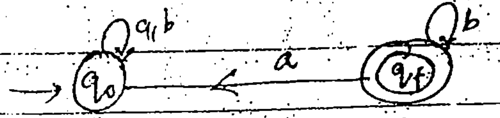
3. The state which can't reach from inital state is called Unreachable state.

4. F.A. in which all accepting / final state is Unreachable accepts Empty Language {∅}

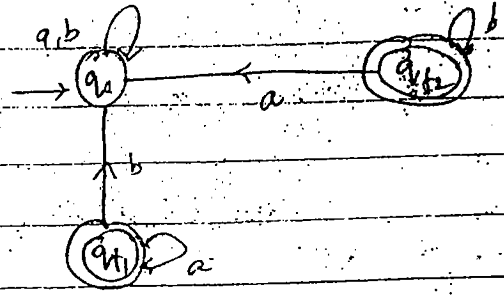
a)



L = ∅

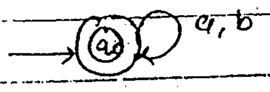


L = ∅

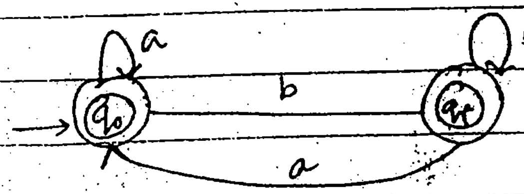


L = ∅

5. F.A. if all states are final states accepts Universal Language Σ*.



L = Σ*



L = Σ*.

starting me.

fom initial
ble state.

inal state
ipty Language{$\emptyset$}

$\emptyset$

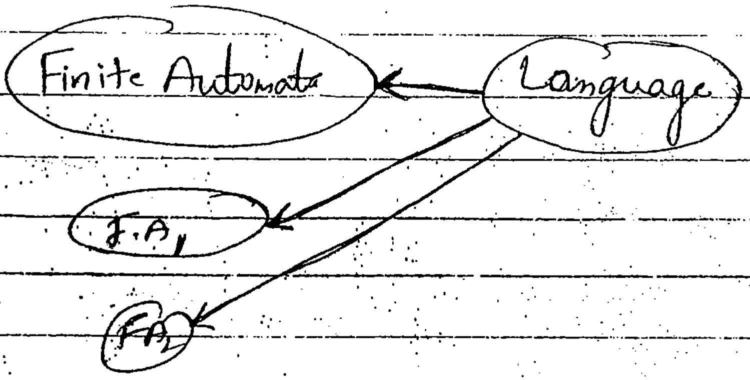
: $\emptyset$

$\emptyset$

final states
Σ^*.

$L = \Sigma^*$

$L = \Sigma^*$.



4-11-

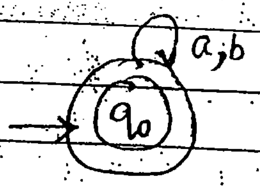
Construction of minimal expression for Infinite Language.

① Construct minimal F.A. accept all string of a & b in cluding ϵ.

Sol:- $\Sigma = \{a, b\}$

$L = \Sigma^* = \{\epsilon, a, b, ab, ba \ldots\}$.

for Construting that sabse pehle hum ϵ Ke liye Banayege.



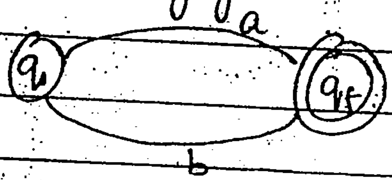
No. of state = 1.

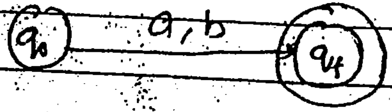
② Cons. min F.A. that accept all string of a,b excluding ϵ.

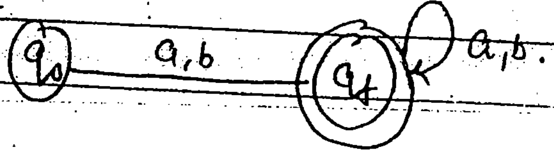
Sol:- $\Sigma = \{a, b\}$.

$L = \Sigma = \{\underbrace{a, b}, ab, ba \ldots\}$.

Min string a aur b hai to hum sabse pehle inke liye Banayege.

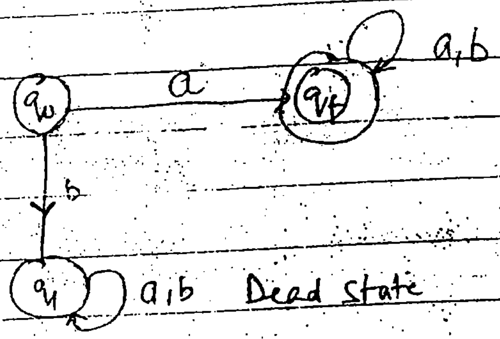






③ All string of {a,b} & every string start with a.

Sol:-

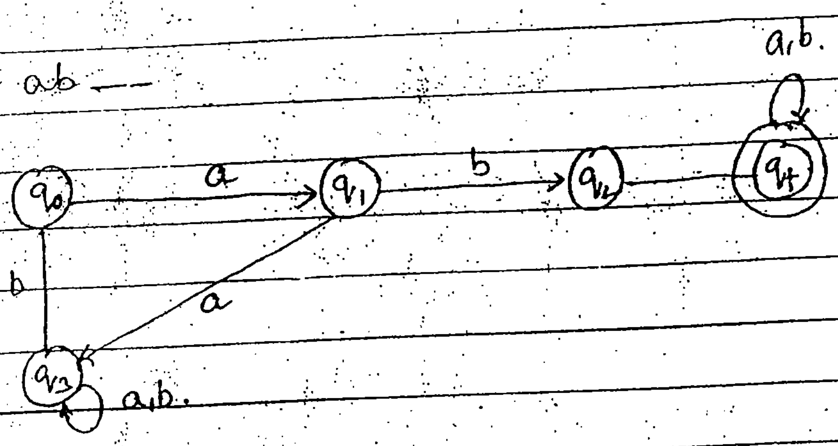


Σ = {a,b}
W = {a, ab, aa, aab}.

No. of state = 3. Hume dead ~~state~~ ko bi lenge.

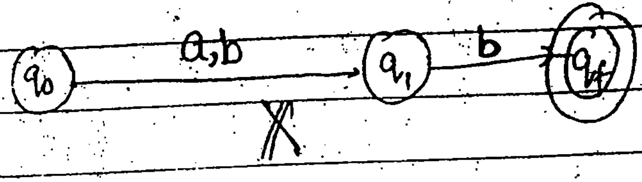
④ All string of {a,b} where every string start with ab.

Sol :- L = ab ---



⑤ Cons Min. F.A that accepts all string of {a,b} ~~except~~ where every string end with b.

Sol :-



⑥ Co
{ c

Sol :- W

{ a ba,

⑦ Com
wh

Sol :-

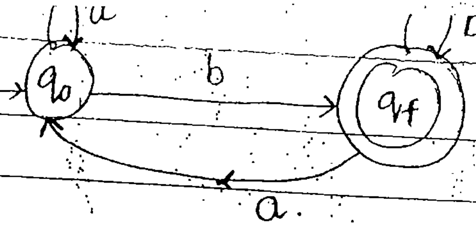
⑧ Al

Sol :-

start with a.

a,b}

ab, aa, aab}.



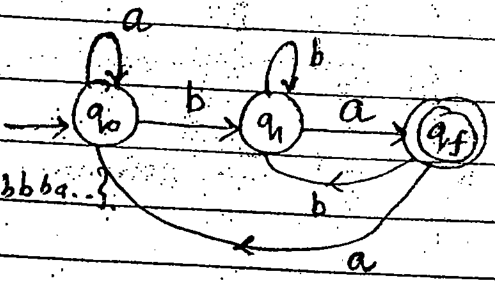
Ans.

⑥ Construct min F.A. accept all string of {a,b} every string end with (b,a).

bilenge.

Sol :- W = X ba

{aba, ba, bba, aaba, bbba..}

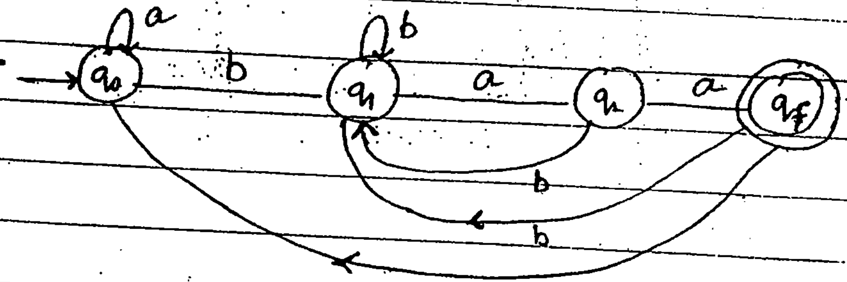


string start with

⑦ Con minimal F.A. that accept all string of {a,b} where every string ends with {baa}.

a,b.

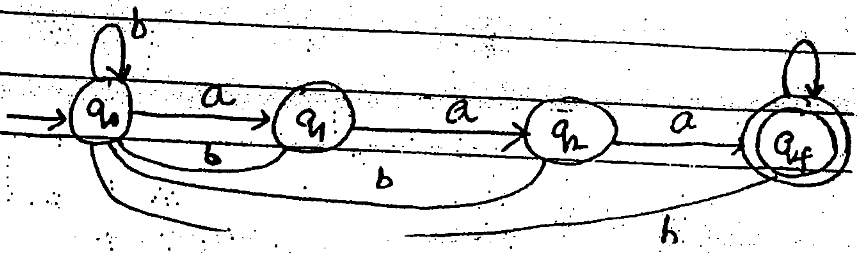
Sol :- W = X baa.



ring of
string end with.

⑧ All string of {a,b}. where each string end with aaa.

Sol :-



Deadstate

ax = 2+1 = 3   xa = 2

abx = 3+1 = 4   xba = 3

abbx = 4+1 = 5   xabb = 4

babax = 5+1 = 6   xbaba = 5.

Dead State :- The state from which we can't come back is called dead state.

a) In DFA D.S. is non accepting state.

b) If DFA contain dead state than we hv to make D.S. in min. DFA also.

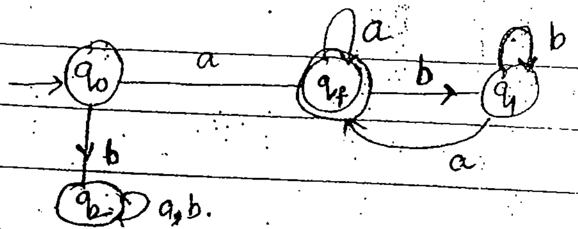
* D.S. can be hide from construction lekin jab hum min state count karenge to usko bi include krna padge.

c) Hide frm cons but to include all in No. of state.

d) A D.F.A can hv. atmost one Dead state.

Q:- Cons. Min F.A. accept all string of {a,b} where every string starts and ends with a.

Sol:- W = axa , a

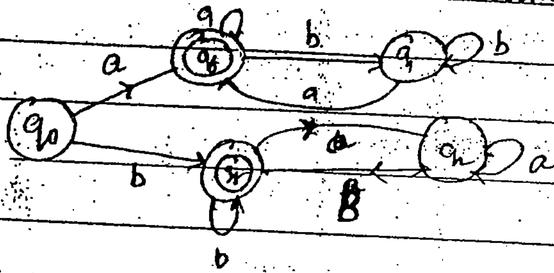


Q:- Accept all state of {a,b} where every string start & ends with same symbo

re Can't

e.

Sol:-



state

hv to make

in jab

to usko

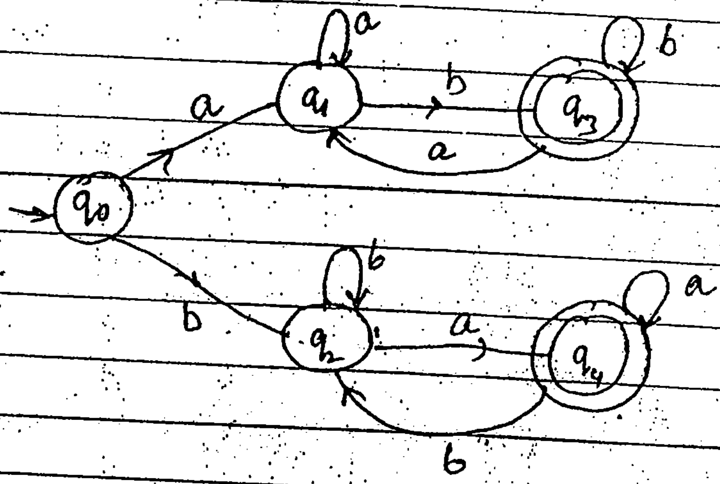
Q:- Accept all string of {a,b} where every string start & end with diff. symbo

Sol:- W = axb = aab, abb, aabb,

bxa = baa,

in No. of state.

Dead state.

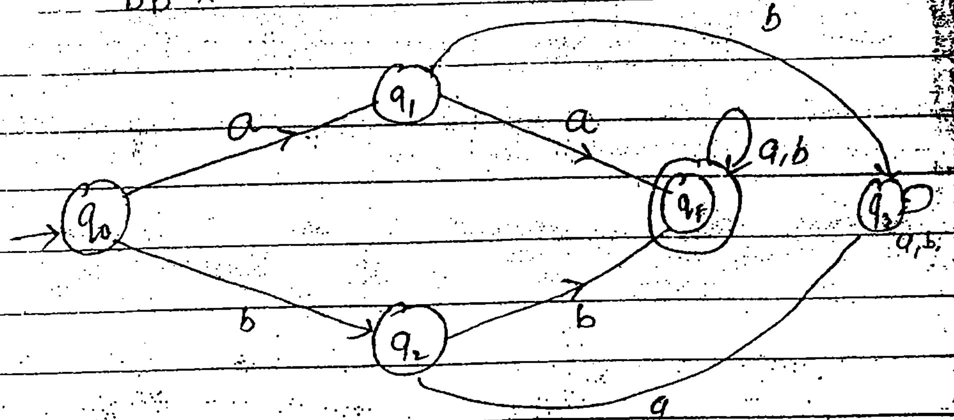


{a,b} where

a.

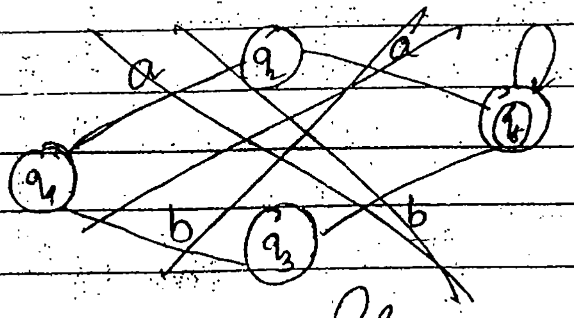
Q:- Cons. min F.A. start with aa or bb.

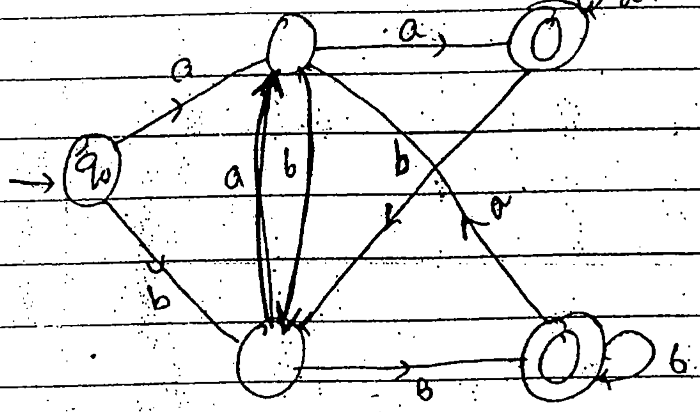
Sol:- W = aa x } (aa + bb) x
bb x



Q:- Min FA accept all string of {a,b} where every string ends with aa or bb.

Sol:-





Q:-
ev
b

Sol:-

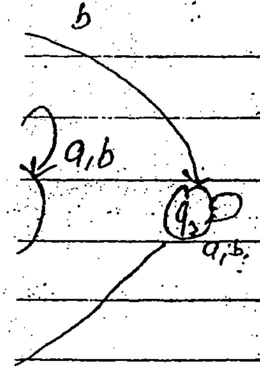
Q:- M
S

Sol:-

Q:- l

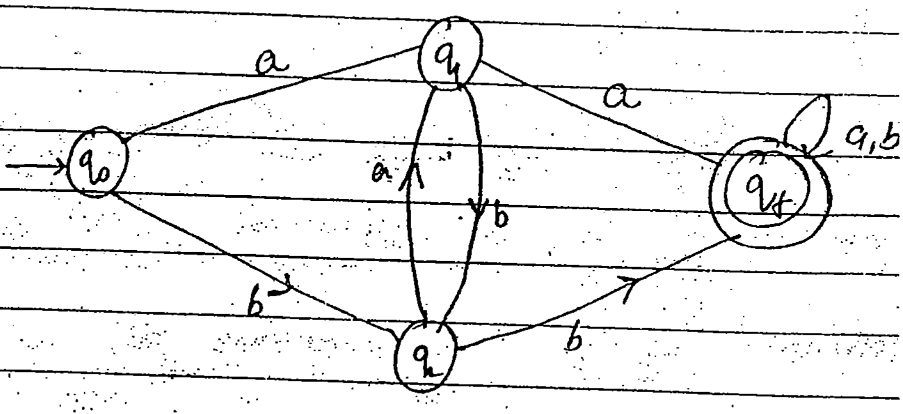
Sol:-

66.



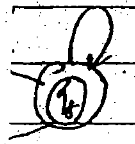
Q:- Cons min F.A accept all string of {a,b}. every string contain digit { which contain aa or bb as substring }.

Sol:-

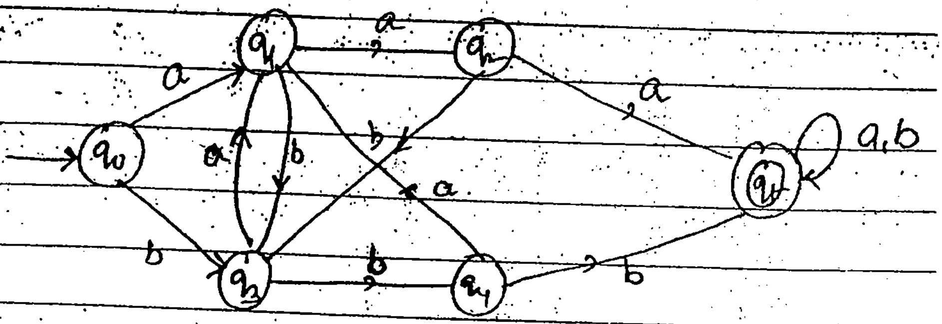


ther every

Q:- Min F.A accept all string of {a,b} where every string Contains tribit { which Contain aaa or bbb } as sul

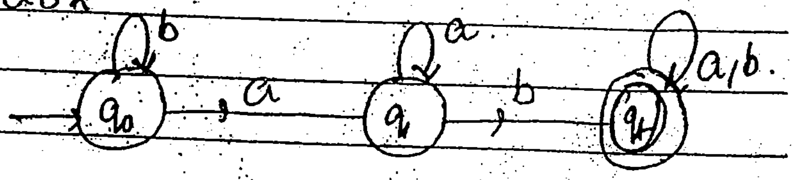


Sol:-



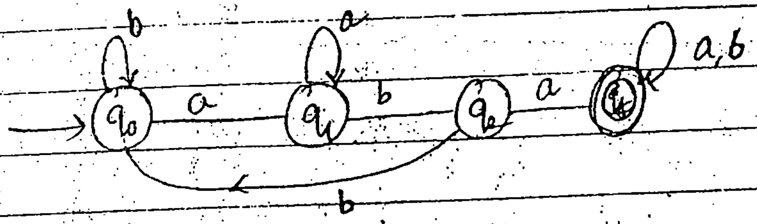
Q:- Where every string Contains {ab} as substring

Sol:- W = x ab x



1Q:- Contains aba as substring.

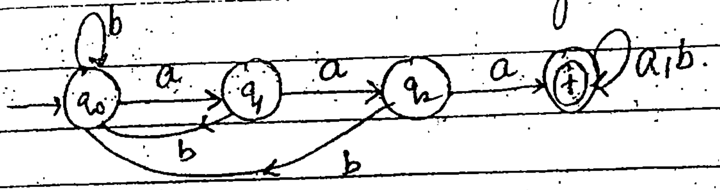
Sol:- W = X aba X



Q:- Contains aaa.

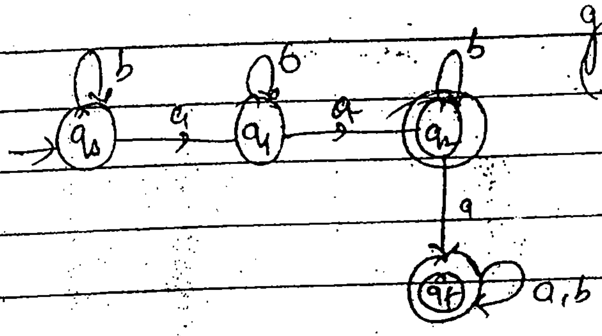
Sol:- W = X aaa X.

When we don't have any restriction on Grammer we get string + 1 States.



Q:- Contain exactly aa

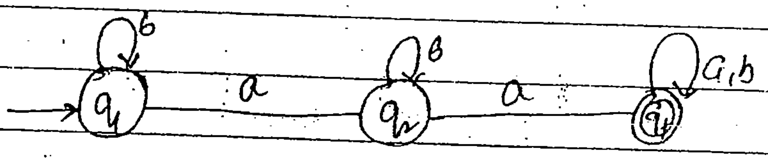
Sol:-



When we restrict grammer we get (string+1) + Dead State

Q:- Contains atleast 2 a's.

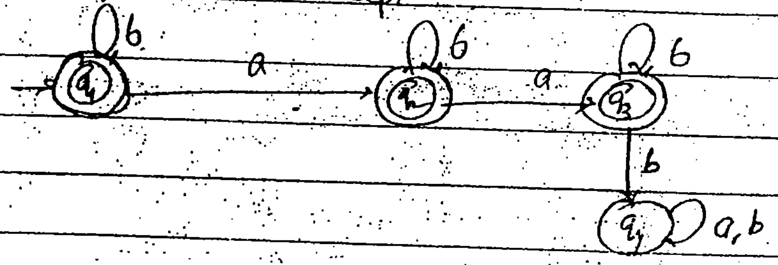
Sol:-



Q:- Contains atmost 2 a's.

Sol:- 0, 1, 2 a's are accept.

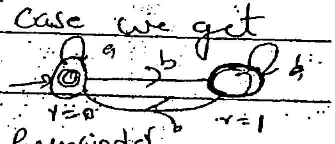
't have
on on
get
states



Q:- Contains even No. of b's.

or In that case we get

Sol:- 0, 2, 4, 6 . . . . remainder = 0, 1



Remainder

↑

0 (mod 2) → Quotient

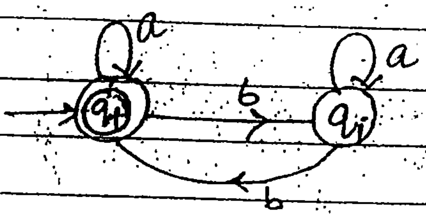
↓

starting from 0 then

0+2

0+2+2

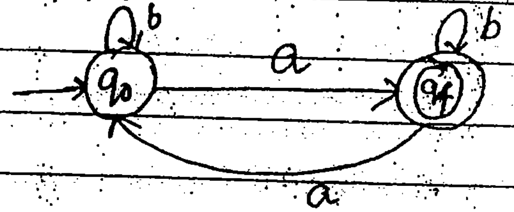
0+2+2+...



restrict
we get
+ Dead
ate

Q:- Odd No. of a's.

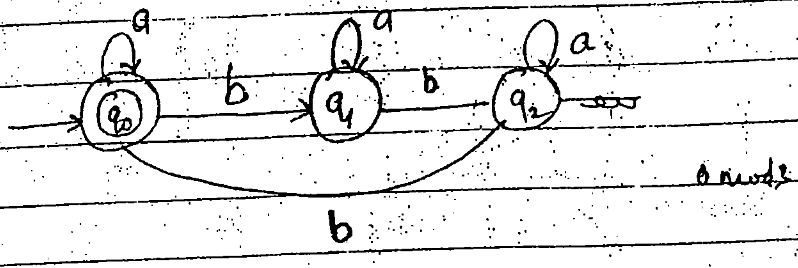
Sol:- a = 1, 3, 5, 7



1(mod 2).

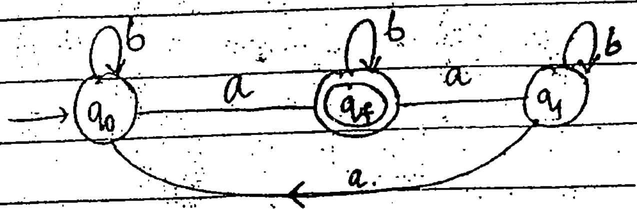
Q:- where No. of b's divisible by 3.

Sol:- b = 0,3,6,9,12



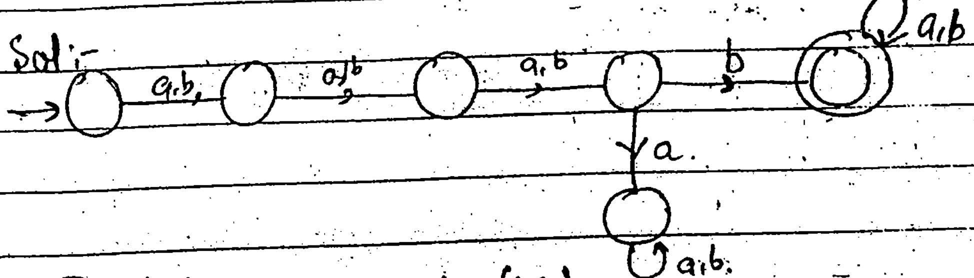
Q:- No. of a's in string is Congruent to (1 mod 3).

Sol:- $(W|_a \equiv 1 \bmod 3 = 1,4,7,10,13.$



Q:- fourth Symbol from L.H.S is always ~~to~~ b.

Sol:-



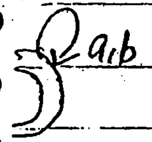
From Left	No of state (Min)
4th	4+2
5th	5+2
6th	6+2

y^3.

mode

nt to (1mod3).

ba b.



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

$L = \{ n_a(w) = r \bmod n$

$n_b(w) = r \bmod \underline{\underline{n}} \}$ then.

Min No. of states in F.A. is $\underline{\underline{N}}$.

Min. F.A. where Left frm N^{th} pos is fixed Contains exactly

$N+2$ states

Min states in a F.A. which ~~right~~ value frm R.H.S is fixed at N^{th} pos. Contains only exactly

2^{N} states

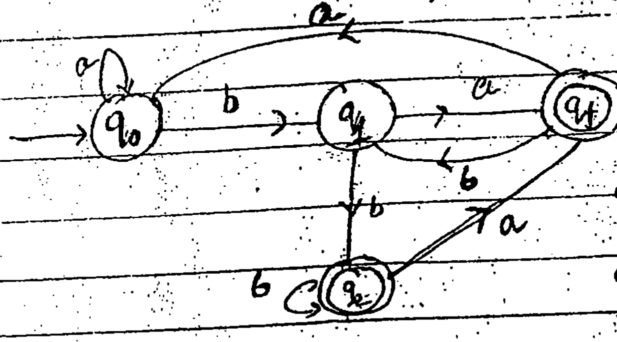
$2^{\overset{k}{n}-1}$ final states

Q:- Second symbol from R.H.S is always B.

Sol:- $\Sigma = \{a, b\}$

W = ... bx   bb
               ba.

write all odd state & even state in ascen-
Not fixed   fixed



	a	b
q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_1
q_3	q_2	q_3

Q:- Where 3 Symbol from Right end is always a.

Sol:-

	a	b
q_0	q_1	q_0
q_1	q_3	q_2
q_2	q_5	q_4
q_3	q_7	q_6
q_4	q_1	q_0
q_5	q_3	q_2
q_6	q_5	q_4
q_7	q_7	q_6

Minimal
Contains

Q:-

Sol:-

length

Q:- (
Whe

Sol:-

q_0

W

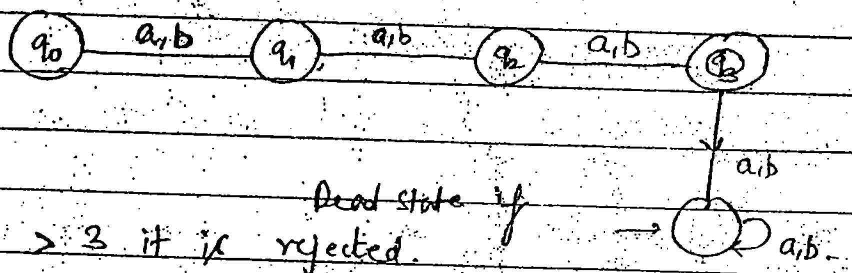
Q:- (

contains $n+2$ states.

Q:- Cons minimal FA accept all strings of a's & b's where length of string is exactly 3.

Sol:- $|w| = 3$

$w = xxx$



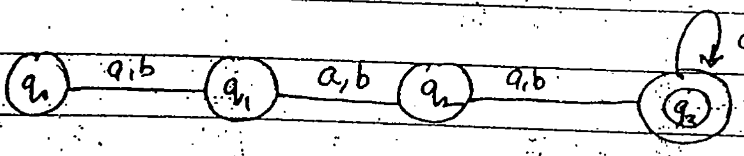
~~Dead state~~ if length is > 3 it is rejected.

$|w| = 3 + 2$ $w_5 = 5+2$.

$|w_4| = 4+2$ $w_n = (n+2)$

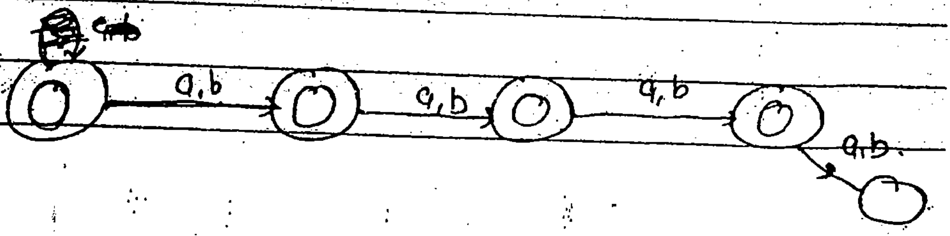
Q:- Cons minimal FA accepts all string of a, b. where length of string is atleast 3

Sol:- $|w| \geq 3$



When length is $w \geq n$ ~~acc~~ Contains 'n+1' Stat

Q:- Const. min FA accept all strings of a, b length of string is ≤ 3.



write all state & even states is ascending fixed

fixed

b

q_4

q_3

q_1

q_3

is always a

b

q_0

q_2

q_4

q_6

q_0

q_2

q_4

q_6

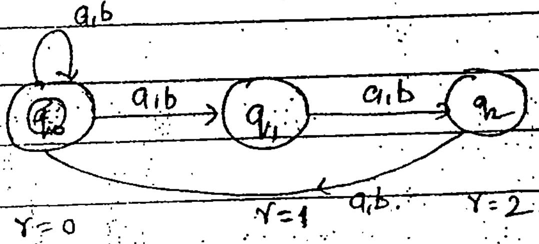
Minimal F.A. that accept all string of a,b of length ≤ 'n' contains 'n+2' states.

No. of string accepted by above F.A. is $2^{n+1}-1$.

$$2^0+2^1+2^2+\cdots 2^n = \frac{2^{n+1}-1}{2-1} = \boxed{2^{n+1}-1}$$

Q:- Construct a minimal F.A. that accept all string of a, b length of string divisible by 3

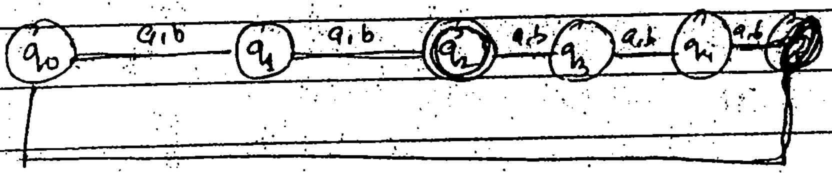
Sol:-



No. of

Q:- String of of a'b Length of string is congrent to 2 mod 5
→ states

Sol:- 2, 7, 12, 17, 22 . . . .



F.A.

Minimal F.A. that accepts all strings of a,b where Congruent of r mod n Contains exactly N states.

-1

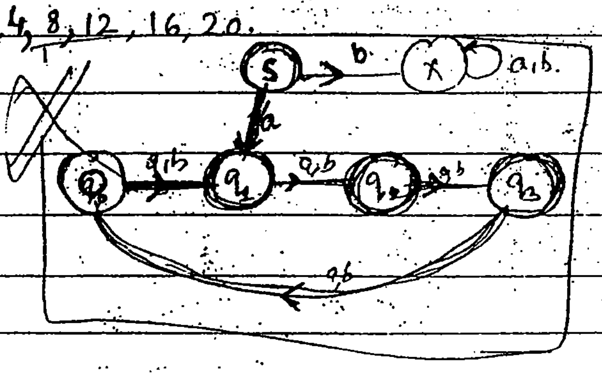
t all

ng divisible

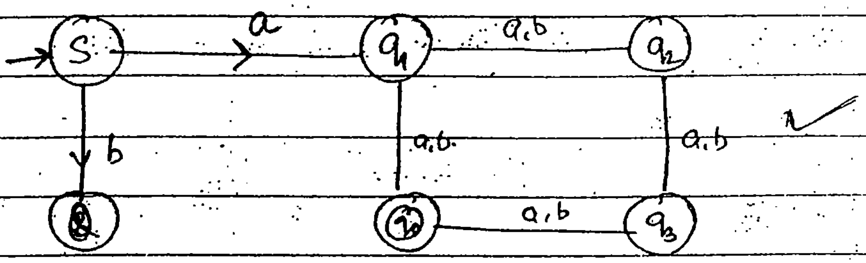
Q:- Cons. minimal F.A. that accept all string of a,b. String start with a & length is congruent to 0 mod 4.

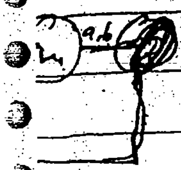
w = ax
↓
1 variable
so
2 states
+
4 states.

Sol:- 0, 4, 8, 12, 16, 20.



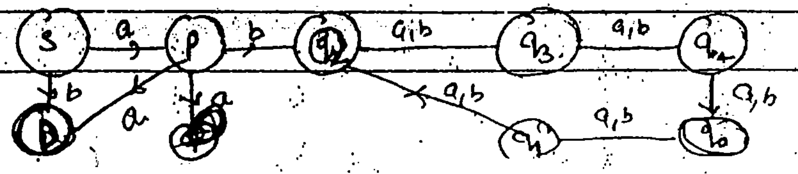
ing if





Q:- Find Number min every string start with a,b length is Congruent of 2 mod 5

Sol:- |W| = $\boxed{ab}$ → 3 states   2, 7, 12, 17   0, 1, 2, 3, 4, ~~5~~
↓
final
+
5 sta



Q:- No. of states in FA accept all string of a & b where string start with aba & length is congruent to 99 mod 100.

$$\text{Sol:-}\quad |w| = \underbrace{aba}_{\text{4 states}}X \quad \& \quad \underbrace{|w| = 99 \ (\text{mod } 100)}_{\text{100 states.}}$$

$\Rightarrow$ 104 states.

#

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

$W = \underbrace{|s|}_{m}X \quad \& \quad |w| = r \text{ mod } n$

Then total No. of states is

$$\boxed{m + 1 + n}$$

s is a substring of length m.

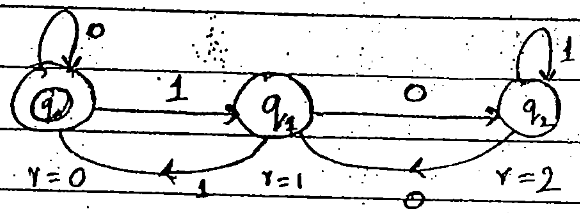
r of a & b
nt to

PROBLEMS On Binary Strings.

Q:- Cons. a minimal FA accept string of 0's & 1's the integer equivalent divisible by 3.

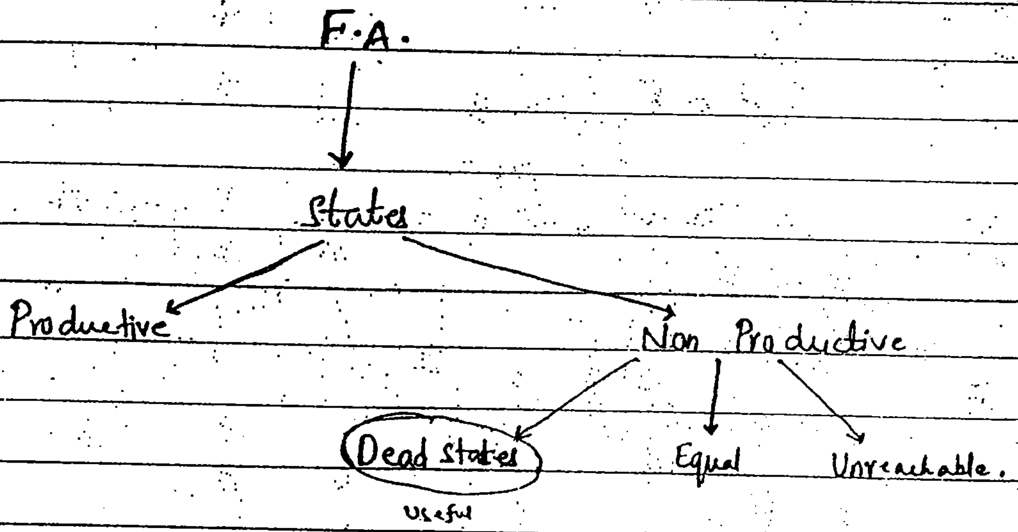
100)
↓
00 states.

Sol:-



0,3,6,9.
"

States is 3.



1) Can't Compare final with other states,
2) Same type of states is Compared. (i.e Non F to Non F, Final to F.
3) Make gps acc. to final & non final states.

$\{q_0\}$ $\{q_1, q_2\}$
G_1 G_2
↓
Break into 2 Gps.

	0	1
q_1	G_2	G_1
q_2	G_2	G_2

↓ Non final
↓ final + Non final so we split this

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$.

Q:- States in Min FA accept Binary strings integer eq. congruent to 1 mod 4.

$n = 4$

Sol:- 1, 5, 9, 13, 15 (101, 1001, 1101)

0,1,2,3

Direct δ

δ	0	1
→ q_0	q_0	q_1
(q_1)	q_2	q_3
q_2	q_0	q_1
q_3	q_2	q_3

(q_0) $q_0 \equiv q_2$

They are also in same group. So we merge them into one

$\Rightarrow 2^2 + = 2+1$ states

δ	0	1
← q_0	q_0	q_1
(q_1)	q_2	q_3
q_3	q_2	q_3

We can't compare final states with the other states.

Min No. of states is 3

Q :- Binary strings Congrent to 2 mod 6.
$\downarrow$ 0, 1, $\textcircled{2}$, 3, 4, 5

Sol:- $\frac{6}{2} + 1$

4 states

δ	0	1
q_0	q_0	q_1
q_1	q_2	q_0
$\rightarrow \textcircled{q_2}$	q_4	q_5
q_5	q_1	q_6

δ	0	1
$q_3 = q_0$	q_0	q_1
$q_4 = q_1$	q_2	q_3
$\rightarrow \textcircled{q_2}$	q_4	q_5
q_3	q_0	q_1
q_4	q_2	q_3
q_5	q_4	q_5

4 States.

Q :- 0 Mod 10
$\downarrow$ 0 to 9 states are formed.

Sol :-

δ	0	1
$\rightarrow \textcircled{q_0}$	q_0	q_1
$q_6 = q_1$	q_2	q_3
$q_7 = q_2$	q_4	q_5
$q_8 = q_3$	q_6	q_7
$q_9 = q_4$	q_8	q_9
✓ q_5	q_0	q_1
α q_6	q_2	q_3
α q_7	q_4	q_5
α q_8	q_6	q_7
α q_9	q_0	q_9

δ	0	1
$\rightarrow \textcircled{q_0}$	q_0	q_1
q_1	q_2	q_3
q_2	q_4	q_5
q_3	q_1	q_2
q_4	q_3	q_4
q_5	q_0	q_1

No. of state = 6

$\frac{10}{2} + 1$ States.

L

q_1

q_2

↓ final + Non final

so we split the

strings.

n = 4

1

q_1

q_3

q_4

q_3

So we

= 2+1 states

nt

final

with

states.

Q:- $\Sigma = \{0, 1\}$

1 Mod 8

Solⁿ:-

	δ	0	1
q_4 =	$\rightarrow q_0$	q_0	q_1
	$\boxed{q_1}$	q_2	q_3
q_6 =	q_2	q_4	q_5
q_7 =	q_3	q_6	q_7
	q_4	q_0	q_1
	q_5	q_2	q_3
	q_6	q_4	q_5
	q_7	q_6	q_7

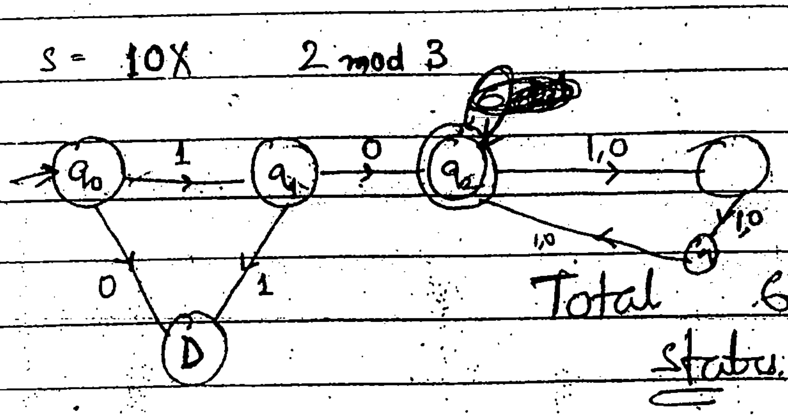
δ	0	1
q_0	q_0	q_1
$\rightarrow \boxed{q_1}$	q_2	q_3
q_2	q_0	q_5
q_3	q_2	q_3
q_5	q_2	q_3

$4 \Rightarrow q_0, q_1, q_2, q_3$ **Ans**

Q:- All Binary strings start with {1,0} & Congrent to 2 mod 3.

↓ 0,1,2

Sol:- s = 10X 2 mod 3



Total 6 States.

Odd	even.	Not expressed in power of m then.
# $3 \Rightarrow 3$	$4 = 2^2 \Rightarrow 3$	
$5 = 5$	$8 = 2^3 \Rightarrow 4$.	$6 = \frac{6}{2} + 1 = 4$
$7 \Rightarrow 7$	$16 = 2^4 = 4+1=5$	$10 = 5+1 = 6$
⋮ n = n states.	2^m = m + 1 states	$14 = 7+1 = 8$ $m = \frac{n}{2} + 1$ state

1

q_1
q_3
q_5
q_3
q_3]

n = Odd	n states
$n = 2^m$	m + 1 states
$n \neq 2^m$	$n/2 + 1$ states

Q:- Find no. of states in minimal F.A. accept start with 101 & congruent to 10 mod 12

Sol:- 4 states + 7 states

11 states.

$\frac{12}{2} + 1$ states

1,0 k

10

6

stes.

18/09/2010

$L = \{ a^m b^n \}$ $m \geq 0$ $n \geq 0$

$m \geq 0$ $n \geq 0$

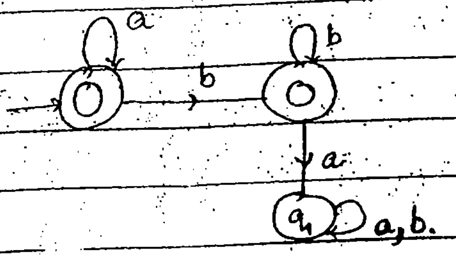
$m \geq 1$ $n \geq 0$

$m \geq 0$ $n \geq 1$

$m \geq 1$ $n \geq 0$

Q :- find minimal F.A. & no. of state for for $L = \{ a^m b^n \mid m, n \geq 0 \}$.

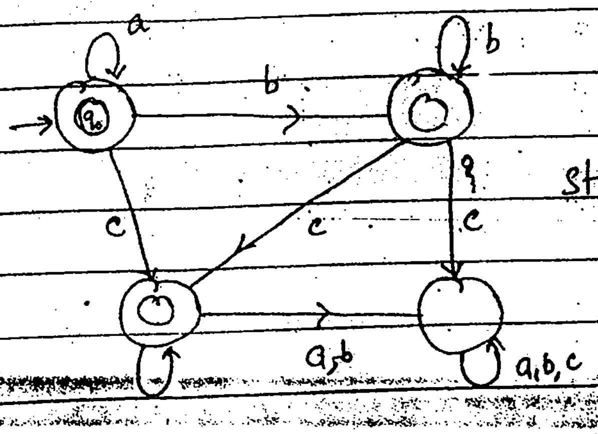
Sol :- $a^m b^n = \{ \epsilon, a, aa, aaa, b, bb, bbb, ab, abb \ldots \}$



No. of symbols 2

No. of states = 2+1

Q :- $L = \{ a^m b^n c^p \mid m, n, p \geq 0 \}$.



Sy. = 3

states = 3+1.

Q :- L

No. o

NOTE :-

∴ If

Q :-

m

Q :- l

q1

Q:- $L = \{a^m b^n c^p d^q \mid m, n, p, q \geq 0\}$.

Symbol = 4

No. of states = 5

NOTE:-

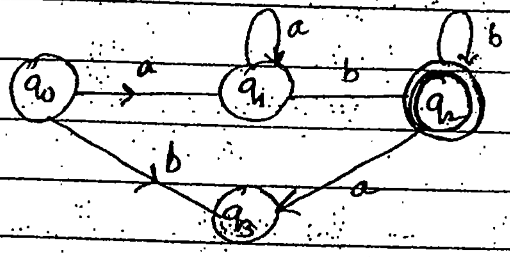
∴ If there are 'N' no. of symbols then the No. of states is 'N+1'.

ates for

Q:- $L = \{a^m b^n \mid m \geq 1 \quad n \geq 1\}$.

ab, abb...}

m = 1 n = 1 min FA a, b ⇒ ab.



Min Length = 2

No. of state = 4.

{ Symbols 2

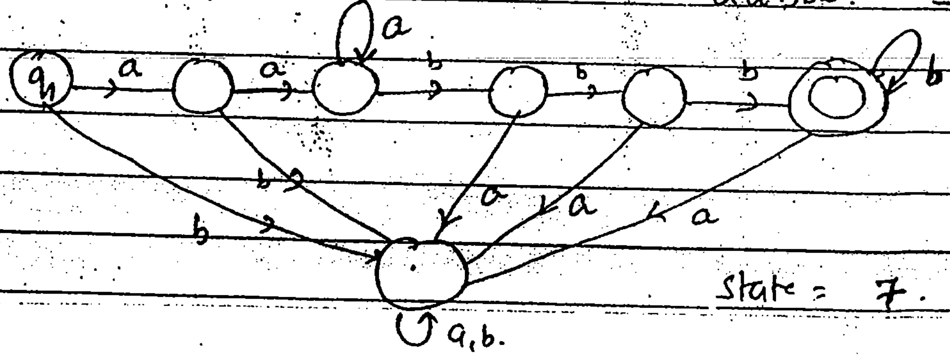
state = 2+1

Q:- $L = \{a^m b^n \mid m > 2 \quad n \geq 3\}$

aabbb = 5

= 3

= 3+1.



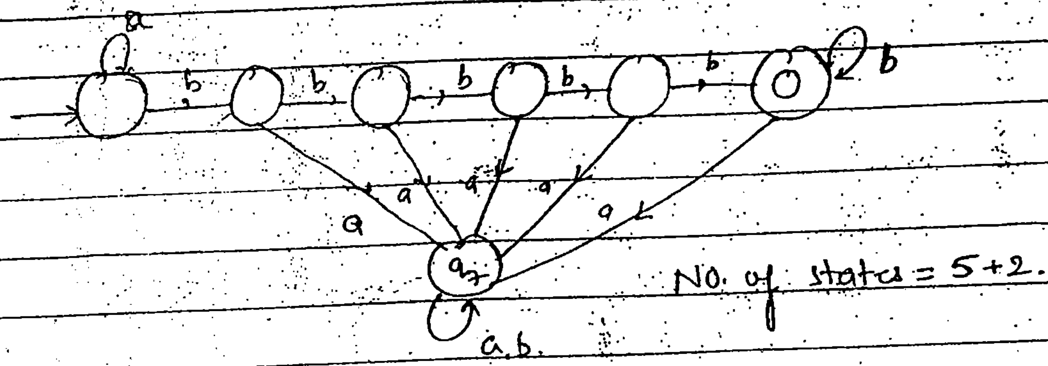
State = 7.

NOTE :-

For Above type of Que. If Min Length of string is 'N' then there No. of states in Minimal Finite Automata = N+2.

Q :- $L_1 = \{ a^m b^n / m \geq 0, n \geq 5 \}$.

bbbbb. = 5



No. of states = 5+2.

Q :- $L = \{ a^m b^n / m \geq 0 \ \ n \geq 7 \}$.

Min length of string = bbbbbbb.

No. of states = 7 + 2 ⇒ $\underline{\underline{9}}$

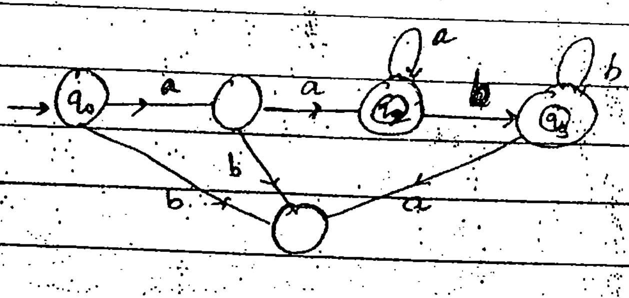
Q:- Min FA for $L = \{a^m b^n / m \geq 2, b \geq 0\}$.

f Min Length

= N+2.

aa. = 2

states = 5



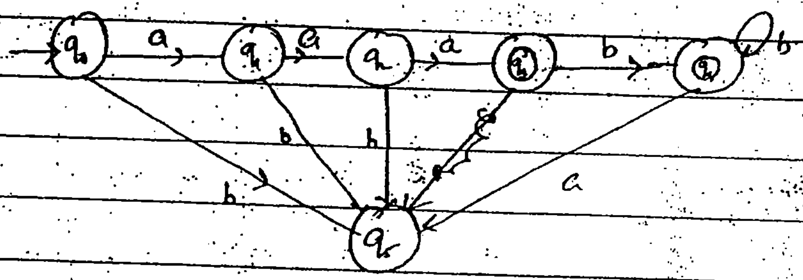
= 5

Q:- $L = \{a^m b^n / m \geq 3, b \geq 0\}$.

?b

aaa = 3.

states = 6.



ata = 5+2.

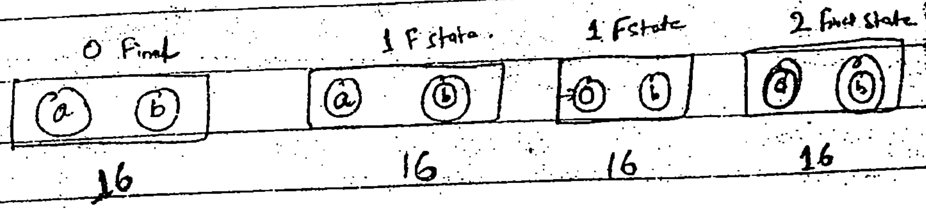
NOTE :-

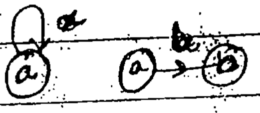
If Min Length of string is 'N' then

No. of states in Minimal FA = N+3.

Q:- How many 2 states DFA with designated initial state can be constr over the alphabet $\Sigma = (a, b)$.

Sol :-





δ	a	b
x	2	2
y	2	2

$= 2^4$

$2^4 \times 4 = \underline{\underline{64}}$ Ans

Q:- How many 3 state DFA cons. with designated initial state can be cons. over alphabet $\Sigma = (a, b)$.

Sol:- $\Sigma = (a, b)$

$Q = \{x, y, z\}$

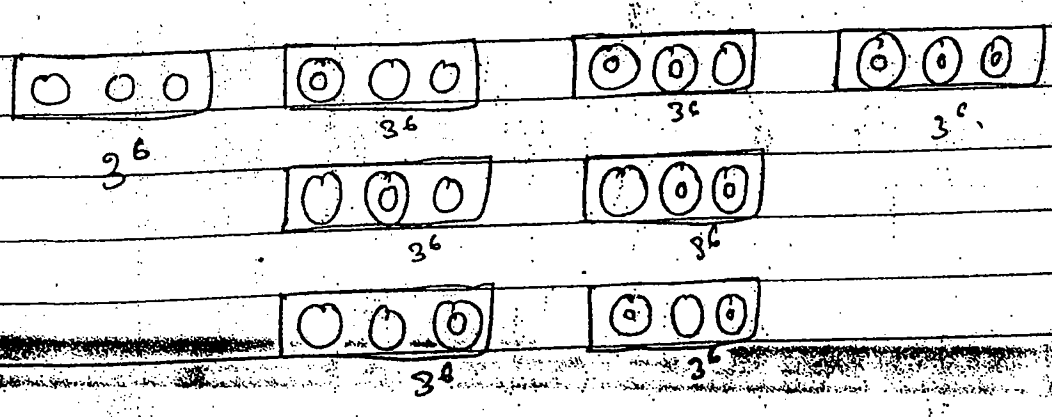
x ↓ Initial

0's = 1

1 final = 3

2 final = 3

3 final = 1

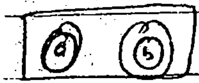


designated
or the

δ	a	b
x	3	3
y	3	3
z	3	3

→ either x or y or z

2 final state.



$$\Downarrow$$

$$3^6 \times 2^3 = 5832$$

16

Generalization

1 Ans

m = No. of Symbols. Σ

n = No. of states.

No. of Type = {0 Final, 1 Finals, 2 Final} = 2^N.

re. with Cons.

No. of each Type = $n^{m \times n}$

∴ Total No. of DFA = $2^N \times n^{m \times n}$.

1

= 3

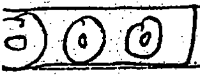
= 3

= 1

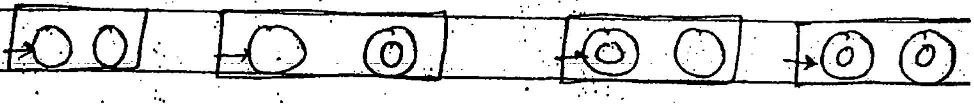
Q:- How many 2 State DFA with design initial states can be Cons. that accept empty lang. $2^2 = 4$

φ.

Soln:-

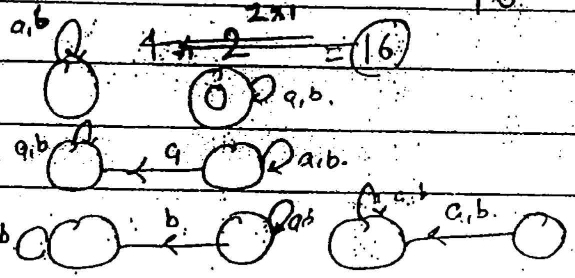


3^c.



16

If final state is Unreachable then the F.A. accepts ϵ.



Q :- How many 2 state D.F.A. Cons over alphabet $\Sigma = \{a, b\}$ that accept Σ^*.

$\Sigma^* = \{ \epsilon, a, b, ab, ba \ldots \}$

Sol : $2^n \times n^{m \times n}$

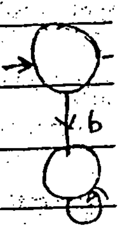
$4 * 2^{2 \times 2} = 64$

All 2 Final states = 16

1 Final states accepts 4 = 20 Ans

COMP

L =



By

NOT

1.)

2)

3) No

4) I

5)* C

6) (

Q

over

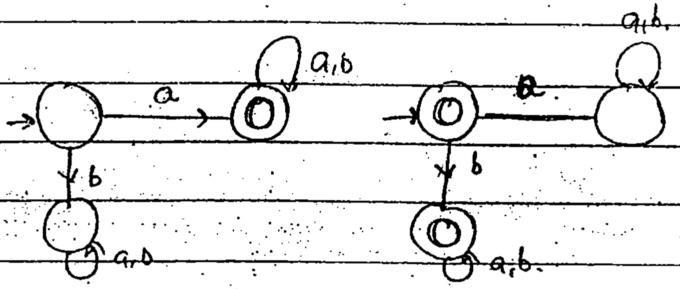
*

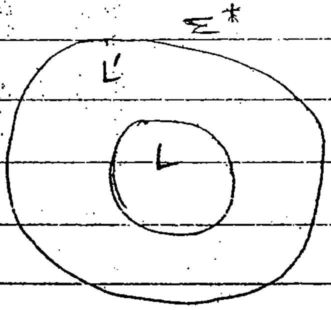
b, ab, ba...}

ni

COMPLEMENT OF FINITE AUTOMATA.

$L = ax$ then $L' = bx$.





$\Sigma^* - L = L'$

By interchaning Final & NonFinal states, we get the complement of F.A.

NOTE :-

1) $L(F.A) \cup L(F.A') = \Sigma^*$

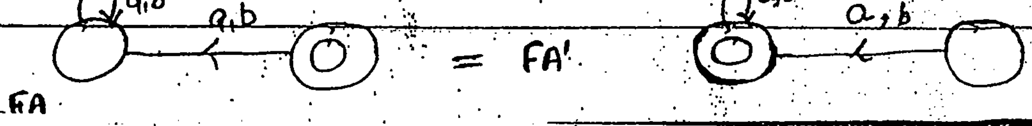
2) $L(F.A) \cap L(FA') = \emptyset$

3) No. of states in FA & FA' are equal.

4) If dead state is non final in F.A then it becomes final state in F.A'.

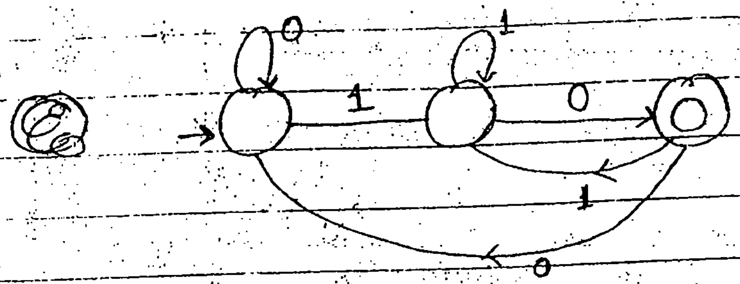
5)* Complement can be defined only for D.F.A. not for NFA.

6) Unreachable states in F.A & FA' ~~of~~ remains same.

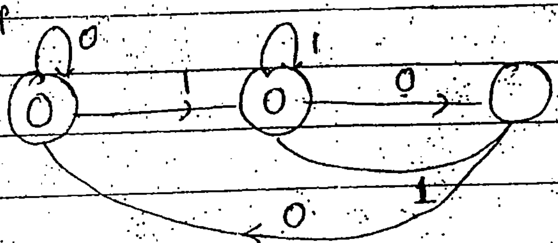


Q:- accepts all string of {1,0} every string don't end with 10.

Accepts 1,0

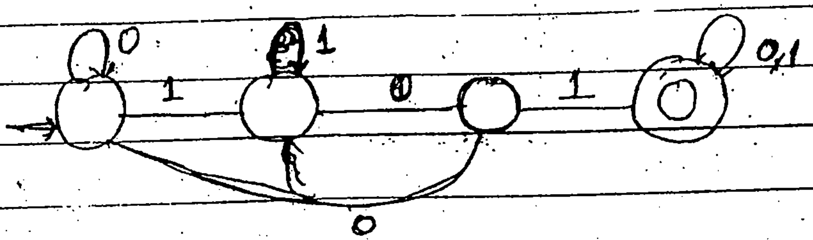


Take comp.

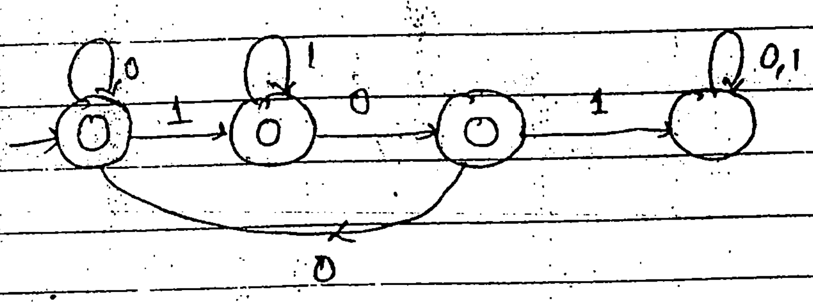


Q:- All string of {0,1} where donot contain 101 as substring

FA'



FA'

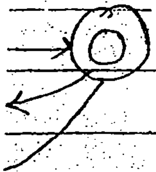


Q:- C

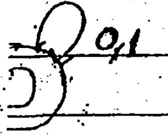
F.A.

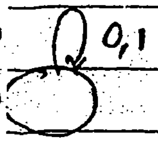
FA'

0} every
10.



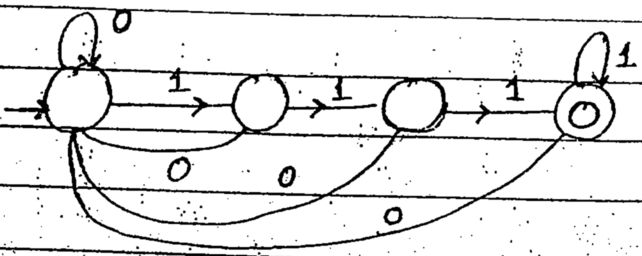
not Contain



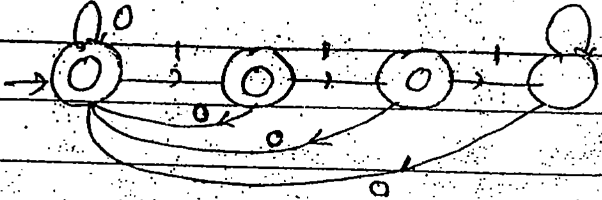


Q:- Comp. Σ = {0,1} donot end with 111.

F.A.



FA'



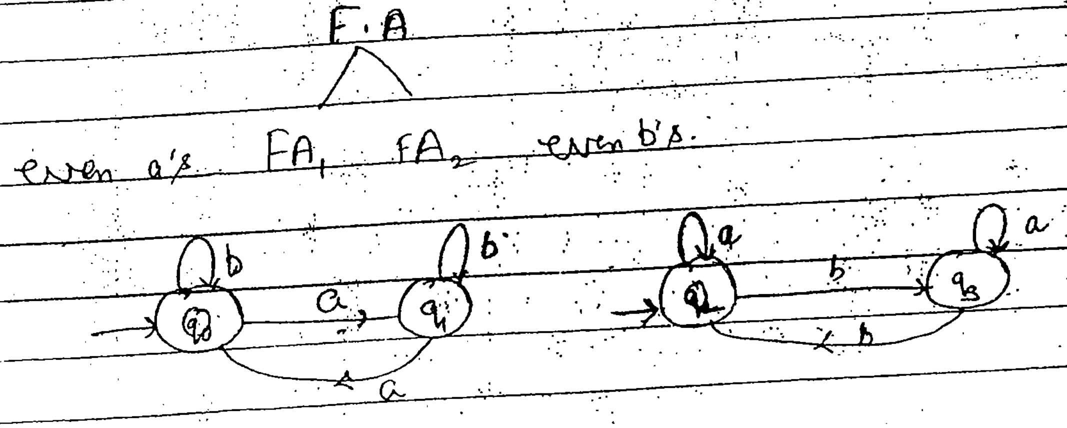
18/09/2010

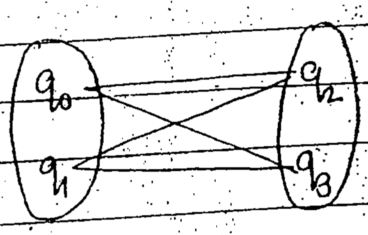
Compound Automata.

i) Cons Minimal F.A. accept all string of {a,b} where each string contains

a) even No of a's X even No. of b's

b) " OR " " "

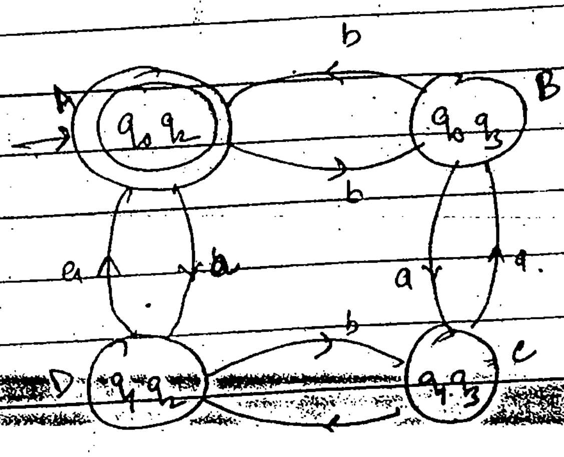




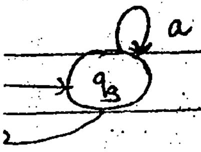
final state of Both DFA becomes final of NEW DFA.

$Q = \{q_0q_2, q_0q_3, q_1q_2, q_1q_3\}$

$\delta = \{q_0q_2, a\} = \delta(q_0, a\} \cdot \delta\{q_2, a\}$



string of.
ring Contains
No. of b's
" "

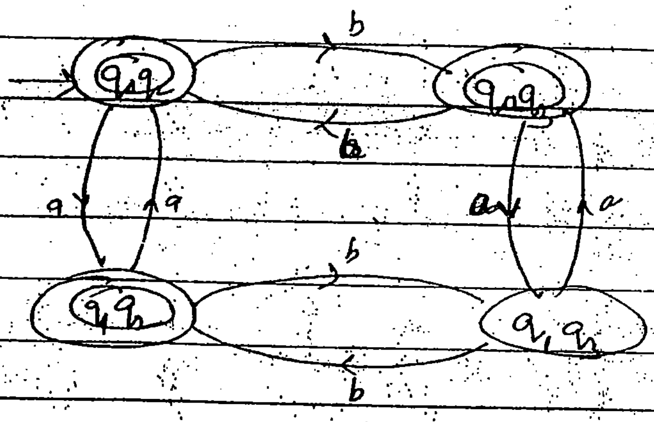


of Both DFA
of

$\{q_2, a\}$

Ⓑ $FA_1 \cup FA_2$

If Any state Contain final state of FA_1 or FA_2 then that state becomes final in Automata.



Minimization:-

(A) (B,C,D)

(A) (D,B) (C)

Ye 2No direct Contact me Hai Final state ke.

A (B,D) (C)

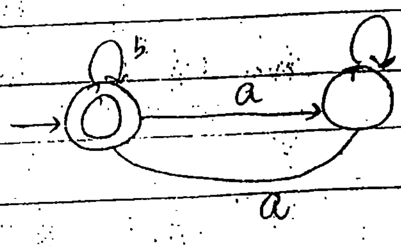
G_1 G_2 G_3

δ	a	b
B	G_3	
D	G_1	

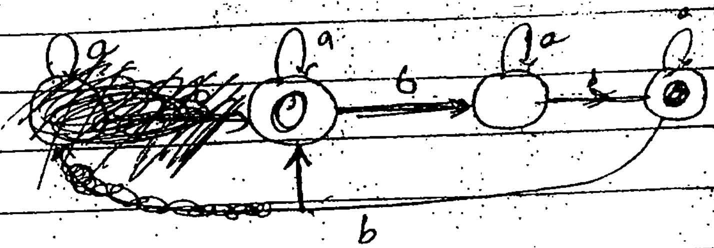
A, B, C, D

Q :- Σ = {a,b} ~~every~~ each string a is divisible by 2 & b's div. by 3.

Sol :- FA₁ =



FA₂ =



q1
q2
0

q3
q4
q5

21/09/10

Non

The
stat

NFA
whe

q0 =
δ =

1. Ca

2. NF
mo

3. Cons

4. Un

5. NF
no

6. No

7.* N

8. E

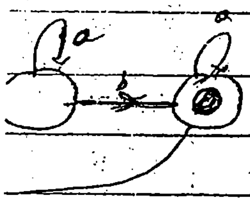
21/09/10. TOC

is divisible by

Non Deterministic Finite Automata. } Powerful than DFA

The F.A which has 0, 1 or more transition from any state for any I/P symbol is called NDFA.

NFA is a 5 tuple $M = \{Q, \Sigma, \delta, q_0, F\}$.
where Q = set of all states.
Σ = Input alphabets
q_0 = Initial state F = set of final state.
$\delta = (q \times \Sigma \rightarrow 2^Q)$ transition f^n.



1. Capability of NFA & DFA is same. L(NFA) = L(D

2. NFA is more powerful than DFA But DFA is more efficient than NFA.

3. Cons. of NFA is easier than DFA.

4. Understand Business logic is easier in NFA.

5. NFA take care of only valid Cons. no need to take care of invalid Cons.

6. No Concept of Dead State in NFA.

7.* No Complement Concept.

8. Every DFA is NFA. If we remove Dead state DFA becomes NFA.

9) NFA Converted in DFA.

10) NFA is Kind of || Computing where U Can run multiple threads Concurrently {Multithreading Concept}.

If any symbol from the left side is fixed at n^{th} position the total No. of states = (N+1).

↑ Also applicable if from R.H.S.

Q:- C

Sol:-

2) Cons.

Sol

3) S

4) S

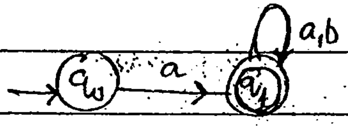
(

where
ads Concurrently

eft side
ion the
(N+1).

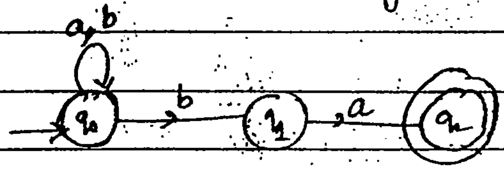
Q.- Cons NFA accepts all string of {a,b} every string start with a.

Sol:- Σ = {a,b}.
W = ax



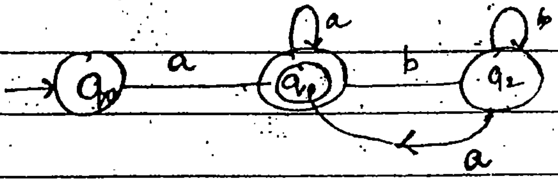
2) Cons. NFA where ending with ba.

Sol.

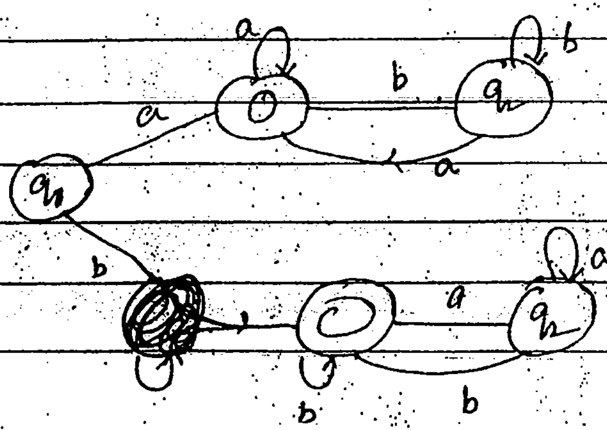


3) Start & end with a.

w = axa, a.



4) Start & end with same symbol.



w = axa, bxb, a, b

NFA = DFA

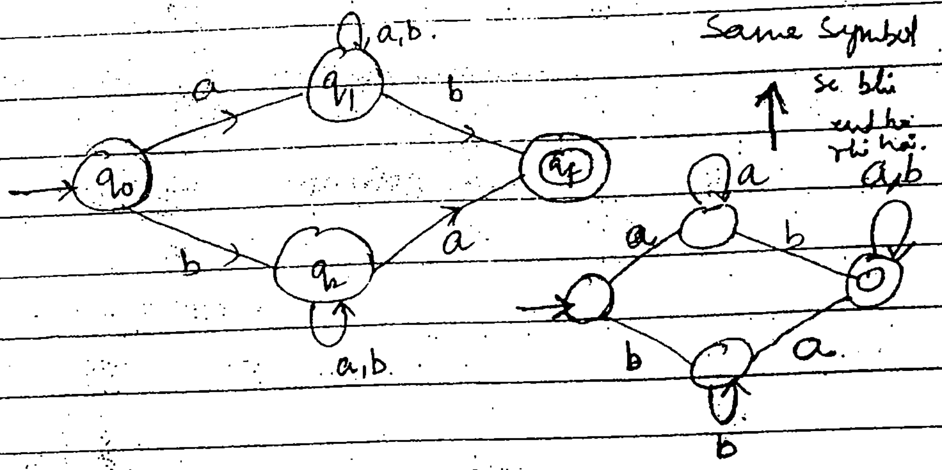
minimal AFA.

5) every string start & end with diff symbol

Sol:- axb, bxa, ab, ba

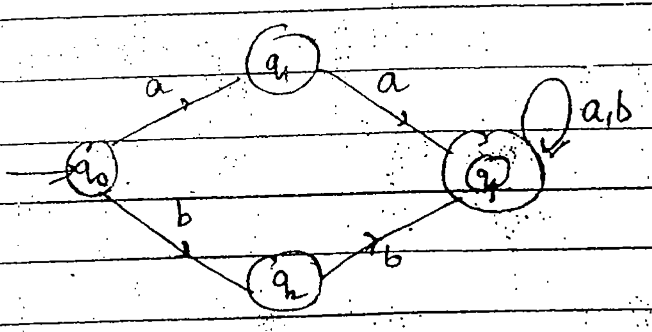
string is me ~~last~~ Same symbol

↑ se bhi end ho rhi hai.

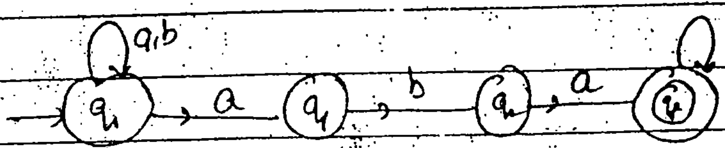


6) every string start with aa or bb.

aax, bbx, aa, bb. = (aa+bb)*



7) Each string Contains a,b,a as substring



8) Exa

Sol:-

9) Wh

10) E

11) thi

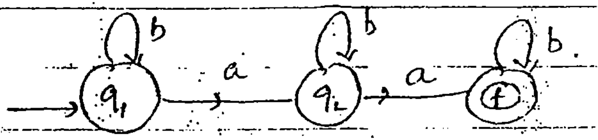
12). L

Sol:-

In diff.

8) Exactly 2 a's

Sol:-

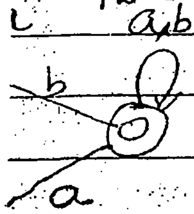


e ~~last~~ string

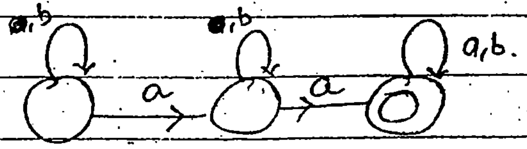
me symbol

↑ se bhi end ho rhi hai.

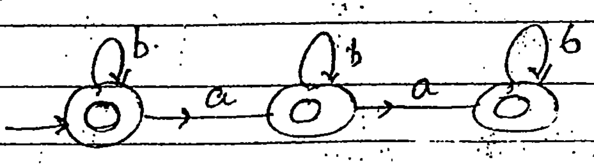
L



9) Where each string contains atleast 2 a's.

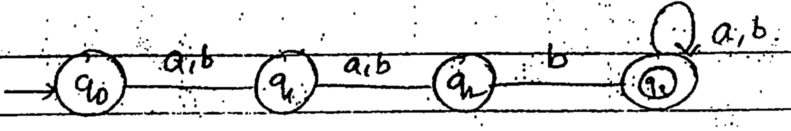


10) Every string contains atmost 2 a's.



*

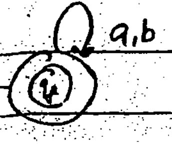
11) third symbol from Left hand is always b

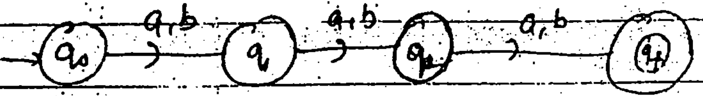


12). Length of string is exactly 3.

Substring

Sol:- W = XXX

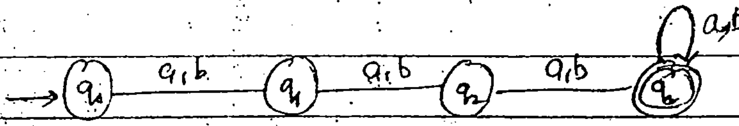




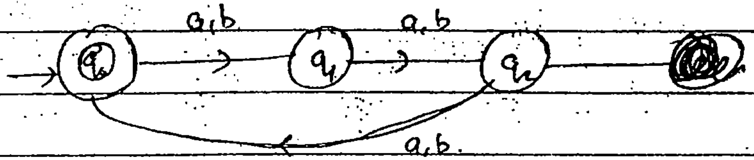
when length of

greabthor than
or less than → state

13) Length of string is ≥ 3.



14). Length of String is divisible by 3. {0, 3, 6, 9, 12, 15}



Incase of Divisibility No. of states in NFA = DFA.

S.No
1. st
2. "
3. Le
4 L
5. L
6. L
7. n
8. nth
9.

Equiv

The
is

NF
n sta

If
1)
2)
3)

S.No.	Language	DFA	NFA.
1.	start & end with same sym.	5	5
2.	" " with Diff symbol.	5	4
3.	$\|w\| = n$	$n+2$	$n+1$
4	$\|w\| \leq n$	$n+2$	$n+1$
5.	$\|w\| \geq n$	$n+1$	$n+1$
6.	$\|w\| = x \bmod n$	n	n
7.	n^{th} sym. fm Left	$n+2$	$n+1$
8.	n^{th} fm Right	2^n	$n+1$
9.	$\|n\|_a = r \bmod n$ $\|n\|_b = r \bmod n$	n	n.

0, 3, 6, 9, 12, 15?

Fee in NFA - DFA.

Equivalence b/w NFA & DFA.

The process of Conversion of NFA into DFA is called subset Construction.

$$\text{NFA} \Longrightarrow \text{DFA}$$

n states $\boxed{1 \leq m \leq 2^n}$ m states.

If

1) $m = 1 \longrightarrow$ Best Case
2) $1 \leq m < n \longrightarrow$ Average Case.
3) $m = 2^n \longrightarrow$ Worst Case.

procedure :-

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

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

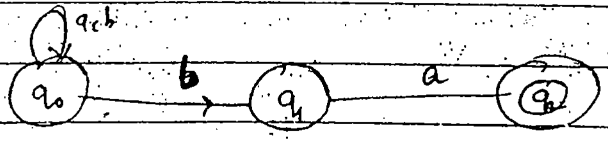
step:-

i) Initial state :-

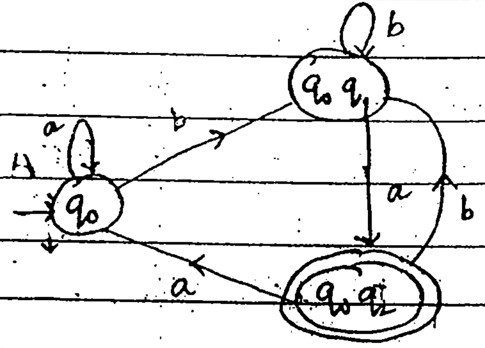
No change in initial state.

$q_0' = q_0$.

ii) start cons of δ' with inital state & Continue process for every new state that appear in Input Column & terminate the process whenever no new state apper in I/P Column.



δ'	a	b
$\rightarrow q_0$	q_0	q_0q_1
q_0q_1	q_0q_2	q_0q_1
(q_0q_2)	q_0	q_0q_1



iii) final state :- every subset which contain final state is final state in DFA.

(2). N

③

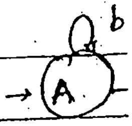
δ'

$\rightarrow q_0$

q_0q_1

q_0q_2

$q_0q_1q_2$



Q:-

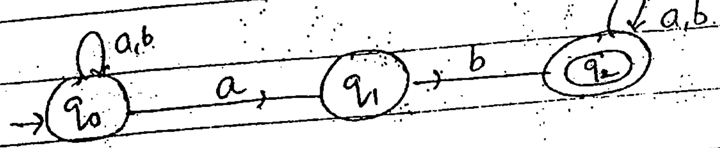
Sol.

(2). NFA

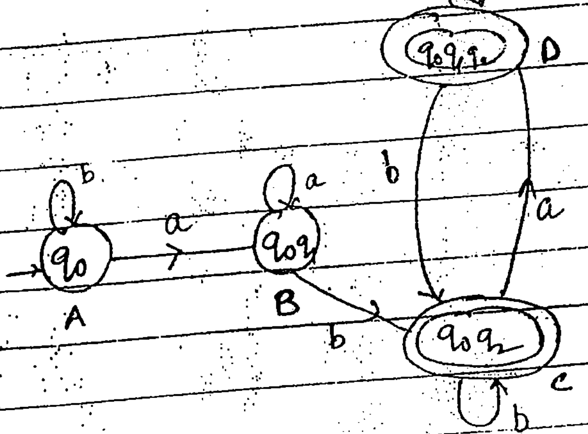
δ	0	1
→ q_0	q_0	q_0q_2
q_1	q_2	—
q_2	q_0q_1	q_2

δ'	0	1
→ q_0	q_0	q_0q_2
q_0q_2	q_0q_1	q_0q_2
q_0q_1	q_0q_2	q_0q_2

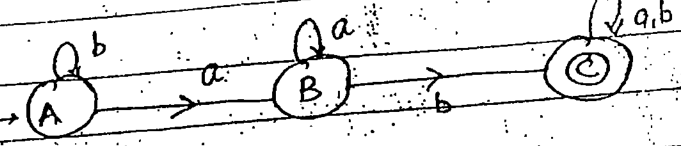
③

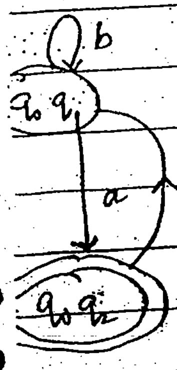


δ'	a	b
→ q_0	q_0q_1	q_0
q_0q_1	q_0q_1	q_0q_2
q_0q_2	$q_0q_1q_2$	q_0q_2
$q_0q_1q_2$	$q_0q_1q_2$	q_0q_2



k
is state
mu: k
er no new
emu.

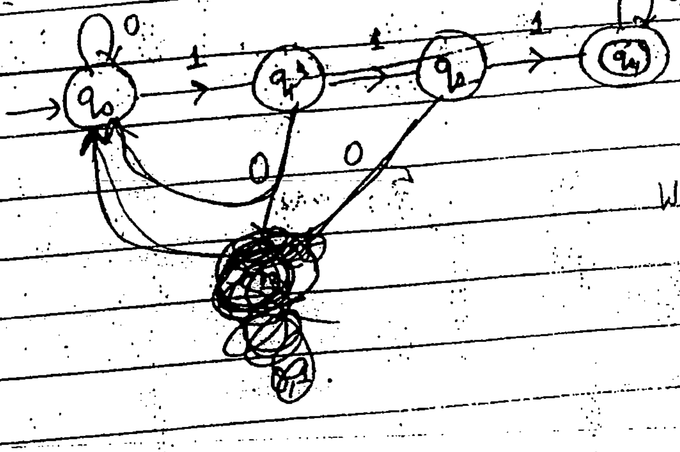




Contains final DFA.

Q:- Minimal DFA of 0&1 3 ones Consecutively.

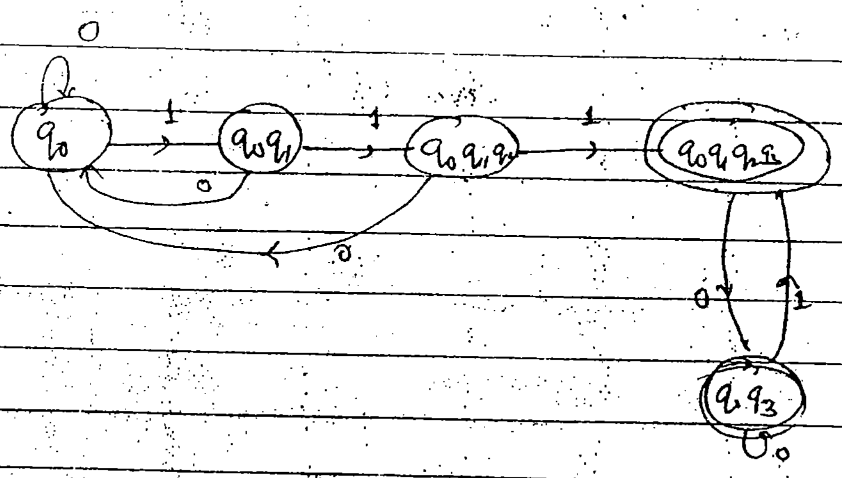
Sol.

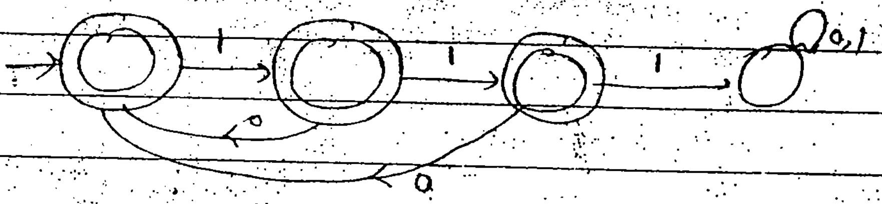


NFA.

Why it is NFA

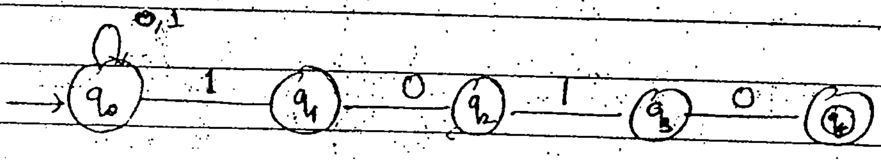
DFA :-

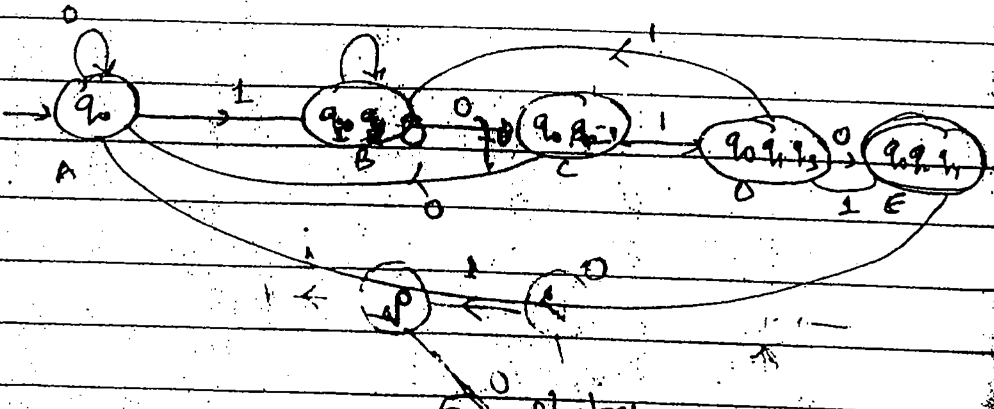




15) Accepts where every string donot end with 1010.

sol:-





5 states

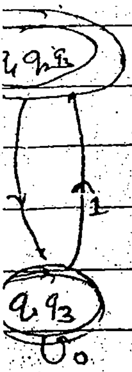
Convert Non final to final & final to Non final states.

16)

M1
↓
L1

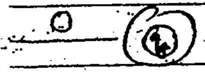
a) L1
b) l
~~c)~~
d)

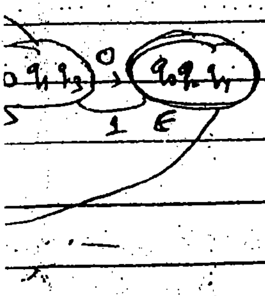
x



)0,1

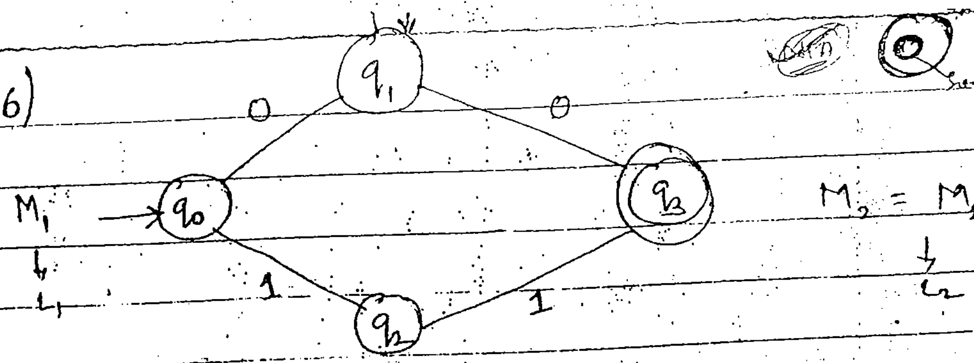
ot end with 1010.





final to
states.

16)



a) $L_1 \cap L_2 = \emptyset$

b) $L_1 = L_2$

c) $L_1 \subset L_2$

d) $L_2 \subset L_1$

~~10/01~~ 01/10/2010. TOC

ϵ NFA or NFA with ϵ moves :-

The NFA which has a transition even for empty string ϵ is called ϵ NFA.

OR

ϵ NFA 5 Tuple $M = (Q, \Sigma, \delta, q_0, F)$

Q = Set of states
Σ = Input alphabet
q_0 = Initial state
F = Final state
$\delta : Q \times \Sigma \cup \{\epsilon\}$ is a transition f^n.

→ Capablities of NFA, DFA, ϵNFA is same.
L(ϵ NFA) = Lang (DFA) = L(NFA).

→ Inclusion or Exclusion of ϵ transitions in NFA will not affect the language of NFA.

→ ϵ NFA will provide programming conveinance

→ ϵ NFA Can be Converted in NFA & fm NFA to DFA.

OR

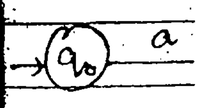
Direct fm ϵ NFA to DFA.

Ex :-

② L

③

④



even for
NFA.

$q_0, F)$

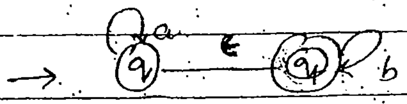
is same
L(NFA).

transitions
the language

2g conveinence

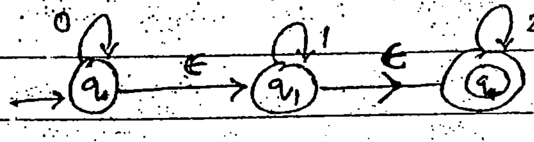
n NFA x

Ex :- $L = \{ a^n b^m / \underset{m}{n} \geq 0 \}$.



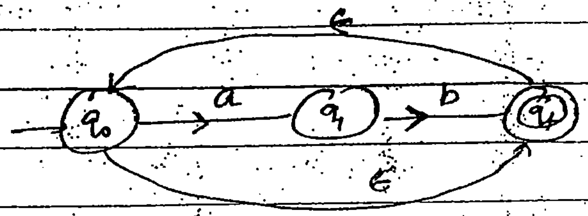
$\epsilon(q_0) = \{q_0, q_1\}$
$\epsilon(q_1) = \{q_1\}$

② $L = \{ 0^m 1^n 2^p / m, n, p \geq 0 \}$



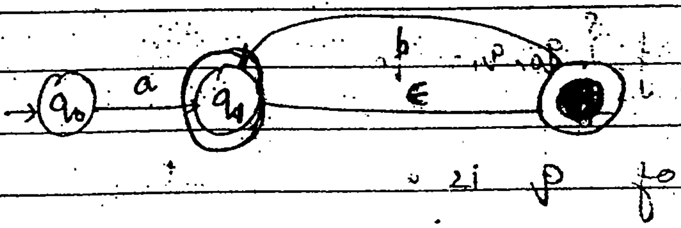
$q_0 = q_0 q_1 q_2$
$q_1 = q_1 q_2$
$q_2 = q_2$

③ L = $L = \{ (ab)^n / n \geq 0 \}$. ab, abab, ababa



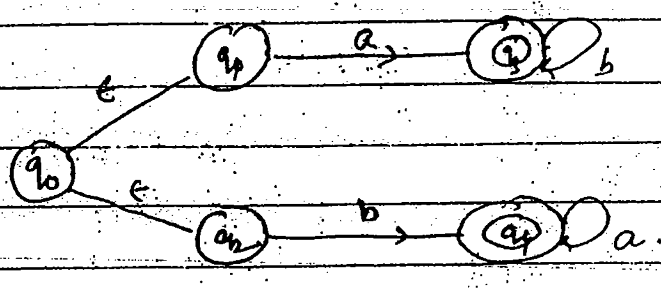
$q_0 = q_0, q_2$
$q_1 = q_1$
$q_2 = q_0 q_2$

④ $L = \{ ab^n / n \geq 0 \}$ a, ab, abb, abbb.



$q_2 = q_0$
$q_1 = q_1 q_2$
$q_2 = q_2$

⑤ $L = \{ ab^n \cup ba^n / n \geq 0 \}$.



$q_0 = q_0 q_1 q_2$
$q_1 = q_1$
$q_2 = q_2$
$q_3 = q_3$
$q_4 = q_4$

ϵ Closure (q) :-

The set of all states which are at 0 distance from state q is called ϵ closure of q.

OR.

set of all states that can be reached from state q along ϵ labelled Transition path.

⇒ Every state is at 0 distance from itself.

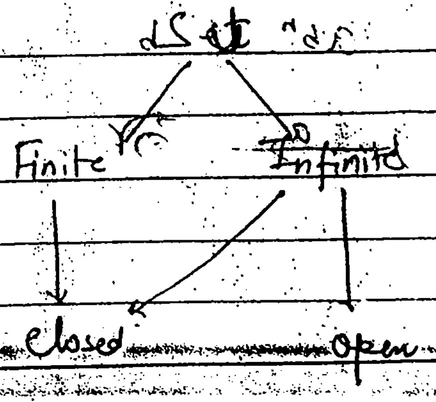
$\delta(\epsilon, q) = q.$

→ ϵ Closure of q is a Non-empty finite subset of Q.

→ ϵ closure of $\phi = \phi$

→ ϵ Closure of $\{q_0, q_1, \ldots q_n\} = \bigcup_{i=0}^{n} \epsilon$ Closure of q_i

→ ϵ Closure of q is a closed set.



Equivalance B/w NFA & ε NFA.

→ No change in the initial state.

→* No change in total No. of states.

→ May be change in the final states.

Procedure :-

$$\text{Let } M = (Q, \Sigma, \delta, q_0, F) \rightarrow \epsilon \text{ NFA}$$
$$M' = (Q', \Sigma', \delta', q_0', F') \rightarrow \text{NFA.}$$

1) Initial state :-

No change in initial state: $q_0' = q_0$

2) Cons. of δ' :-

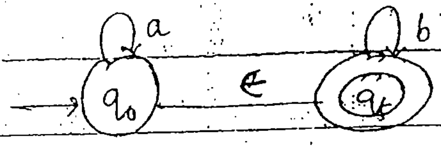
$$\delta'(q, x) = \epsilon \text{ closure} \{ \delta(\epsilon \text{ closure}(q), x) \}$$

3) Final state :-

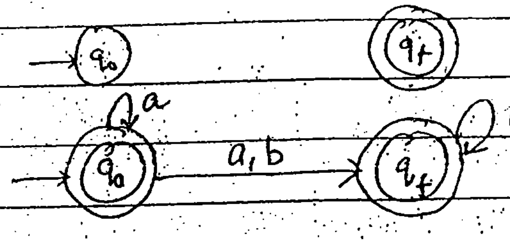
Every state whose ε closure contains final state of ε NFA is a final state in NFA.

are at
is called
be reached
labelled
from itself.
n empty
ε closure of q_i
set.

Q :-



Sol :- ①



$\delta''(q_0, a) = \epsilon \text{ Closure } \{\delta(\epsilon \text{ clos}(q_0), a\}.$

q_0 ~~Read~~ ∈ dene par kiya state Milenge

$= \epsilon \text{ Clos } \{\delta((q_0, q_1), a)\}.$

$= \epsilon \text{ clos } \{q_0 \cup \emptyset\}$ → In done state par a dene par kya Hoga

$= \epsilon \text{ Clos } \{q_0\} = (q_0, q_1)$

q_0 par ∈ dene par Kaya Milega.

$\delta'(q_0, b) = \epsilon \text{ Closure } \{\delta(\epsilon \text{ clos}(q_0), b\}.$

$= \epsilon.c \{\delta(q_0, q_1), b\}.$

$= \epsilon\text{-}c \{\emptyset \cup q_1\}$

$= q_1$

$\delta'(q_1, a) = \epsilon \text{ Clo}\{\delta(\epsilon \text{ Clo}(q_1), a)\}$

$= \epsilon \text{ clo } \{\delta(q_1, a)\}$

$= \emptyset$

$\delta'(q_1, b) = \epsilon \text{ Clo } \{\delta(\epsilon \text{ clo}(q_1), b/$

$= \epsilon\text{cl}(\delta(q_1, b))$

$= q_1$

② →

Sol :- ∈

① q_0 par ∈

to humko

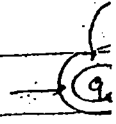
milega + j

nikal rahe

bhi include

② Uske Ba

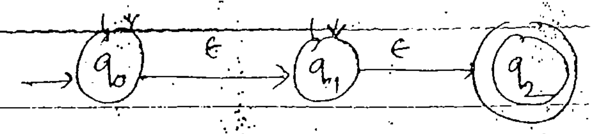
③ Jo st



$q_1 \xrightarrow{\epsilon}$

$q_2 \xrightarrow{\epsilon}$

②



①

sol :- $q_0 \xrightarrow{\epsilon} q_0 q_1 q_2$

$q_0 q_1 q_2$	$q_0 q_1 q_2$	$q_0 q_1 q_2$
↓	↓ 1	↓ 2
0 - ②	q_1	q_2
↓	↓ ε	↓ ε
q_0		
↓ ε - ③	$q_1 q_2$	q_2

① q_0 par ε denge to humko $q_1 q_2$ milega + jiske liye nikal rahe hai wo bhi include karenge. q_0, q_1, q_2

② Uske Baad Hume sare transition ke states Nikalenge.

③ Jo state milegi uspar phir se ε transition ke liye chek karenge.

↓

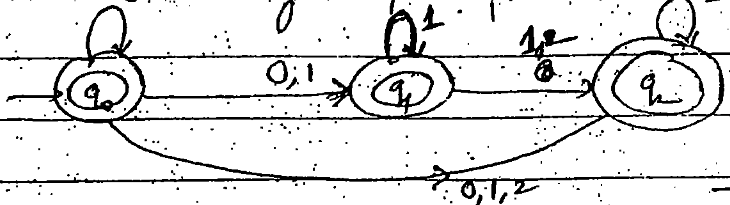
Jis state par ε de rhe hai wo to aayegi + Additional states

a }.
par kya state Milengi

Done state par done par kya Hoga

q_1)
milega.

2 }.



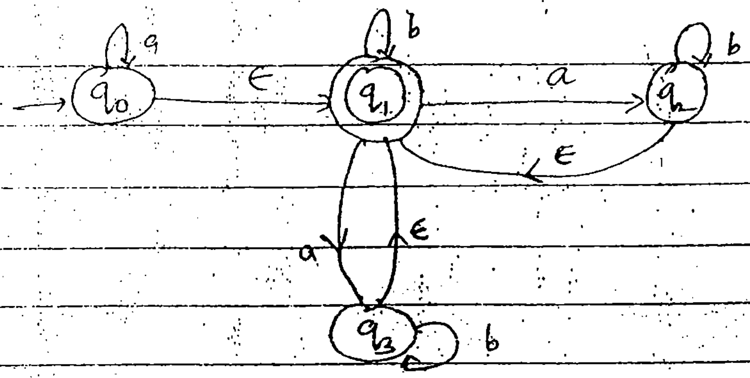
$q_1 \xrightarrow{\epsilon} q_1 q_2$

$q_1 q_2$	$q_1 q_2$	$q_1 q_2$
↓ 0	↓ 1	↓ 2
∅	q_1	q_2
↓ ε	↓ ε	↓ ε
∅	$q_1 q_2$	q_2

$q_2 \xrightarrow{\epsilon}$

q_2	q_2	q_2
↓ 0	↓ 1	↓ 2
∅	∅	q_2
↓ ε	↓ ε	↓ ε
∅	∅	q_2

Q :-

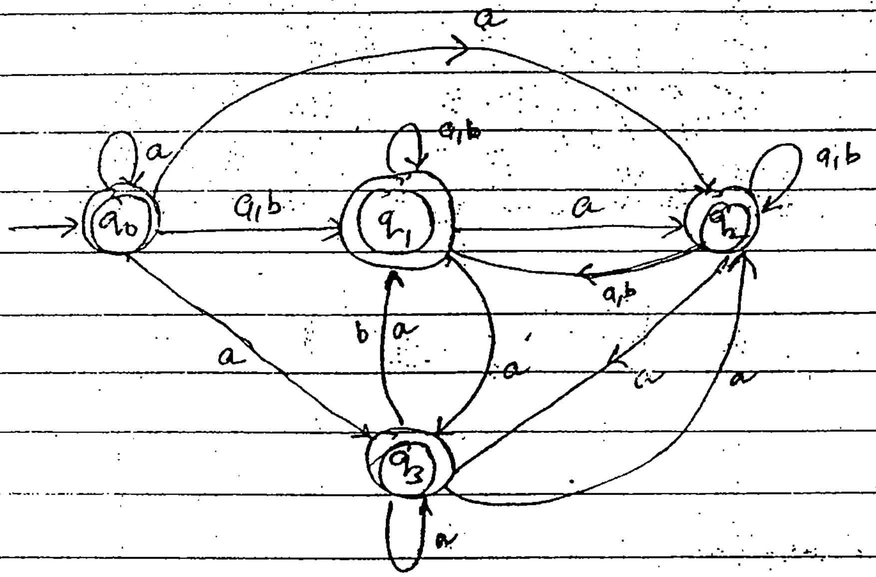


$q_0 = q_0 q_1$

$q_1 = q_1$

$q_2 = q_2, q_1$

$q_3 = q_3, q_1$



$$\begin{tikzcd} q_2 \arrow[r, "\epsilon"] & q_2 q_1 \arrow[d, "a"] & q_2 q_1 \arrow[d, "b"] \\ & q_2 q_3 \arrow[d, "\epsilon"] & q_2 q_1 \arrow[d, "\epsilon"] \\ & q_1 q_2 q_3 & q_1 q_2 \end{tikzcd}$$

Ea

→ May

→ May

→ May

Procedure

Let

1. Initi

2. Cons.

δ'	0	1
→q0q1	q0q1	q
q1	Ø	
Ø	Ø	

3. Sta

K

State

Column

new

Equivalence B/W ϵ NFA & DFA.

→ May be change in initial state.

→ May change in Total No. state.

→ May be Total No. of final states.

Procedure :-

Let $M = (Q, \Sigma, \delta, q_0, F)$. → ϵ NFA

$M' = (Q', \Sigma', \delta', q_0', F')$ → DFA.

q,b

1. Initial state.

$$q_0' = \epsilon \text{ Closure } (q_0).$$

2. Cons. of δ'

δ'	0	1
$\rightarrow q_0q_1$	q_0q_1	q_1
q_1	$\emptyset$	q_1
$\emptyset$	$\emptyset$	$\emptyset$

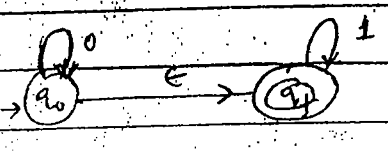
$$\delta'(q, x) = \epsilon \text{ Closure } \{\delta(q, x)\}.$$

3. start the Cons. of δ' with Initial state & Continues the process for every new state that appear on the input Column & end the process when no new state appear in Input Column.

③ Final State :-

Every Subset which Contains F state of ∈ NFA is final state in DFA

①

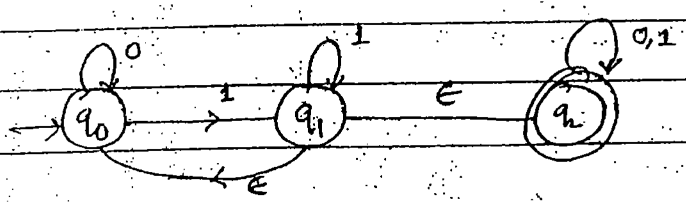


Initial state ∈ (Initial of ∈ NFA)

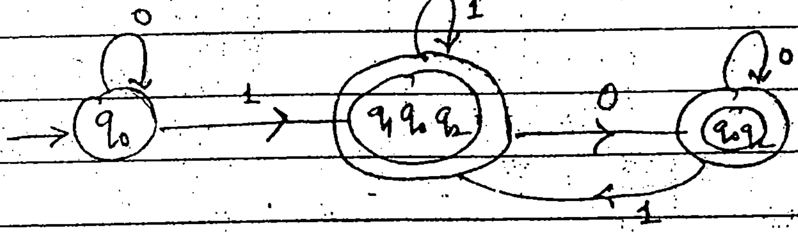
	0	1
→ q0 q1	q0 q1	q1
q1	∅	q1
∅	∅	∅

$q_0 \xrightarrow{\epsilon} q_0 q_1$ = Initial state

②



DFA



$q_0 \xrightarrow{0} q_0 \xrightarrow{\epsilon} q_0$

$\xrightarrow{1} q_1 \longrightarrow q_1 q_0 q_2$ (New state)

$q_0 q_1 q_2 \xrightarrow{0} q_0 q_2 \xrightarrow{\epsilon} q_0 q_2$ (New state

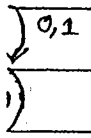
$\xrightarrow{1} q_1 q_2 \xrightarrow{\epsilon} q_0 q_1 q_2$

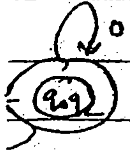
ns F state

state in

state ∈ (Initial of ∈ NFA)

$q_0 q_1$ = Initial state



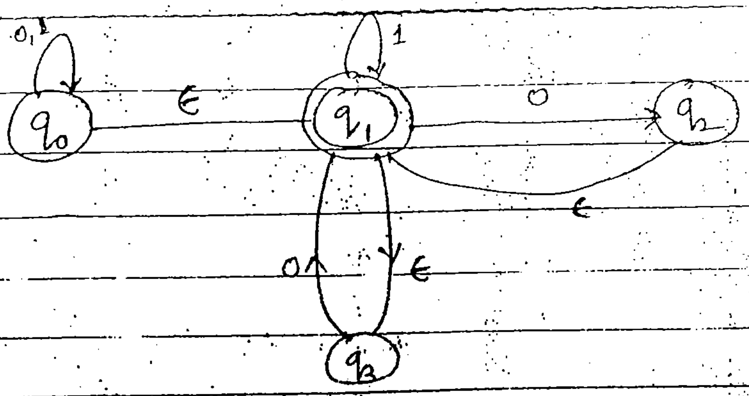


(New state).

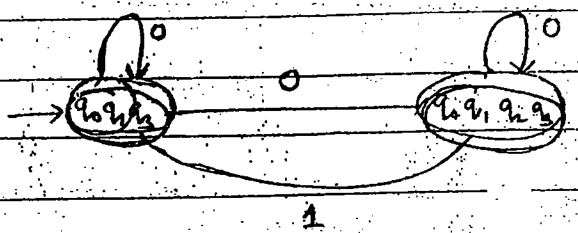
$q_0 q_2$ (New state

$q_0 q_1 q_2$

③

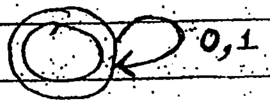


DFA :-



If all states are final then it will accept all Languages.

MDFA



Initial state = $\in(q_0)$ = $q_0 q_1 q_3$

q_0

Decision Properties.

1) Finiteness / Infiniteness

2) Emptyness / Non Emptyness. — No final state accept empty language
↓ If final state exists.

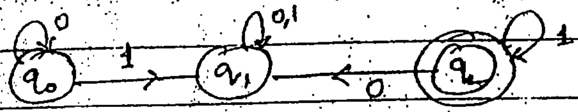
3) Equalness.

4) Membership

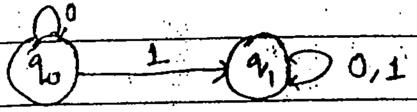
1:- EMPTYNESS :-

a) Select the state which can't be reached from initial state (Unreachable) X delete them.

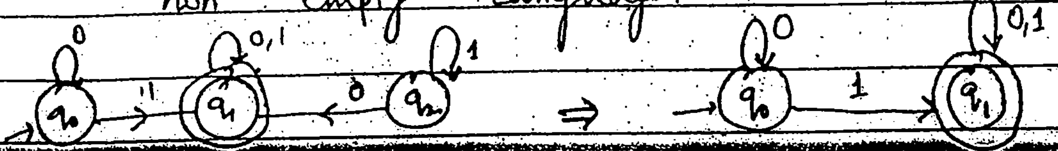
Ex.



b) If resulting m/c ~~machine~~ free from final state then it will accept empty language.



c) If resulting machine ~~mission~~ contains atleast one final state then ~~mission~~ will accept non empty language.



---

2. FINI

a) Select initial

b) Selec Can them

*c) If X fini

d) Con mis

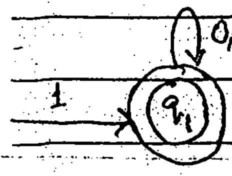
Ex①

d state accept
mpty Language
exists.

tate which
ial State

inal state
language.

st One
vill accept



2. FINITENESS :-

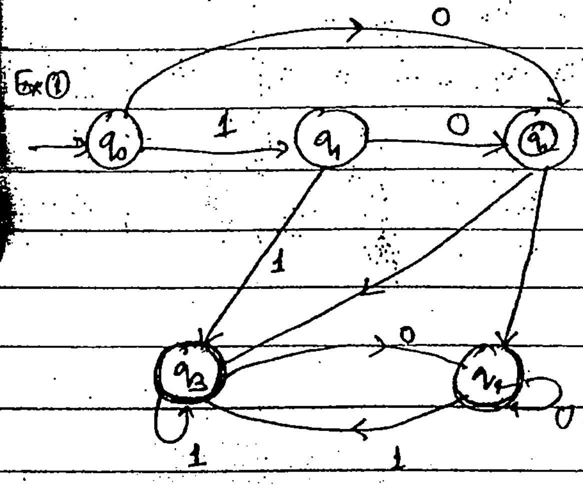
a) Select state which can't be reached by initial state & Delete them.

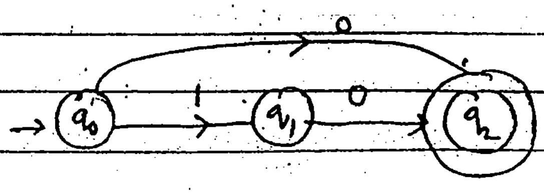
b) Select those states from which we can reach final state & deleted them. (Dead State)

*c) If resulting $\overset{m/c}{\text{~~mission~~}}$ is free from cycles & self loops then mission accept finite Language.

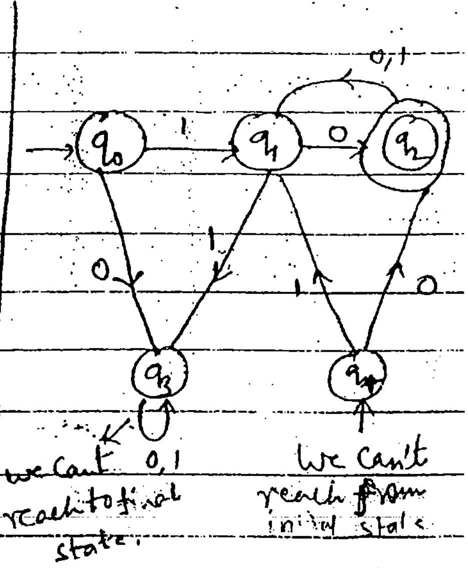
d) Contains self loops or cycles then mission will accept Infinite Lang.

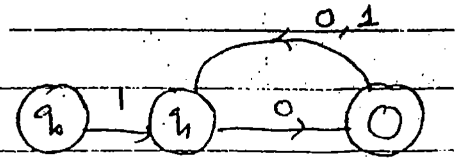
Ex(1)





q3 & q4 are dead states





27/10/10.

Equal

is

accpet

∴

NOTE :-

1) Eq

sta

2) Tw

to

Co

(b) I

(c) (

if

then

(3) T

if

(

(A

$f(a) :$

$f(b) =$

27/10/20

Equalness :- 2 finite states missions M_1 & M_2 is said to be equal if both M_1 & M_2 accepts same set of strings.

$$\therefore \quad M_1 = M_2 \iff L(M_1) = L(M_2)$$

NOTE :-

1) Equal missions no need to contain same no of states.

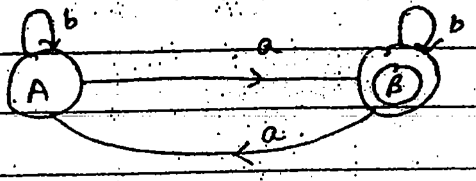
2) Two mission M_1 & M_2 said to be isomorphic to each other if both of them (a) accepts same language

(b) If contains same No. of states.

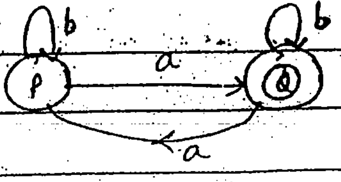
(c) Have the same properties.

if by replacing state we get same F.A then they are c/d Isomorphic.

(3) Two mission are isomorphic to each other if & only if one can be obtained by other by replacing the states.



M_1



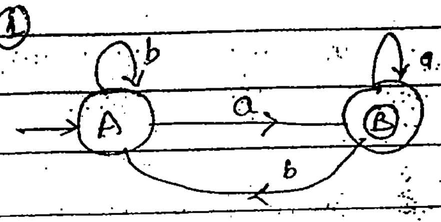
M_2

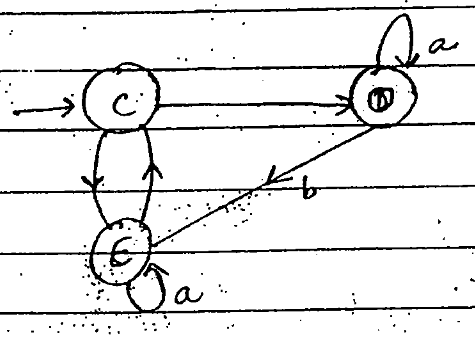
f(a) = P   If there is no edge b/w P, Q then also no edge b/w AB
f(b) = Q

4. Isomorphic missions are equal but equal missions are not need to be isomorphic.

Comparision → Examples

①



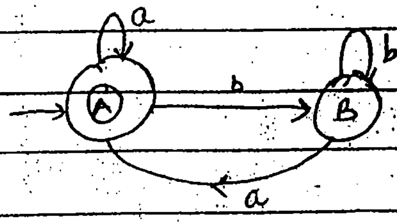


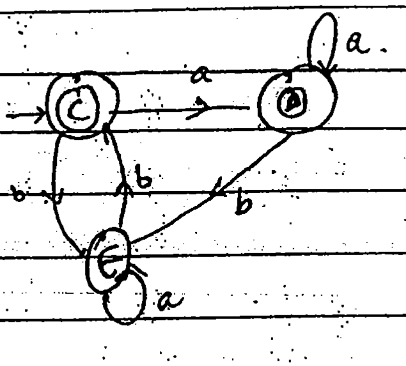
take initial state

δ	a	b
A, C	(B, D) F F	(A, E) N, NF
B, D	(B, D) F F	A, E) NF NF
A, E	(B, E)	

F, NF So it is not equal.
↓
B, C either Both has to final or Non Final

②





δ

(A,C) (

(A,D) (,
(F

(B,E) (,
(F

Compa

1) M,

a) Co
of
x

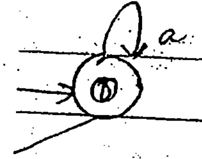
b) Sta
in

c) In
p
d
e

d) Co
pai

but
to

δ	a	b
(A,C)	(A,D) (F F)	(B,E) (N,F)
(A,D)	(A,D) (F,F)	(B,E) (NF,NF)
(B,E)	(A,E) (F,NF) → Not equal	



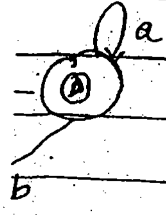
Comparision Theory

1) M_1 & M_2 are 2 missions.

a) Construct transition table that contains pair of states (P,Q) where P is from M_1 & Q is from M_2

or Non Final

b) Start con of table with the pair of initial state.



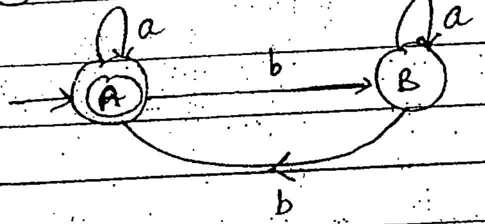
c) In the cons process if we get any pair of form (Final, NF), (NF, F) then stop cons of table & declare that 2 missions are not equal.

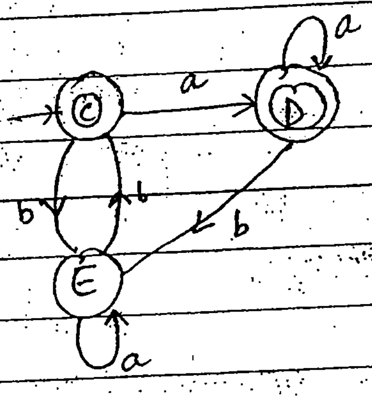
d) Continue to cons the table for every new pair of form (F,F), (NF, NF) that appear in Input Column & terminate the process

Whenever no such new pair comes in the Input Column.

5) If transition table contains all the pairs of form (NF, NF) or (F,F) then the 2 missions are equal.

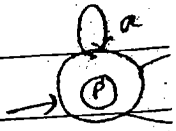
Ex ③.





1. δ	a	b
(A,C)	(A,D) F F	(B,E) NF, NF
(A,D)	(A,D) F F	(B,E) NF, NF
(B,E)	(B,E) NF,NF	(A,C) F F

So these are equal Missions.

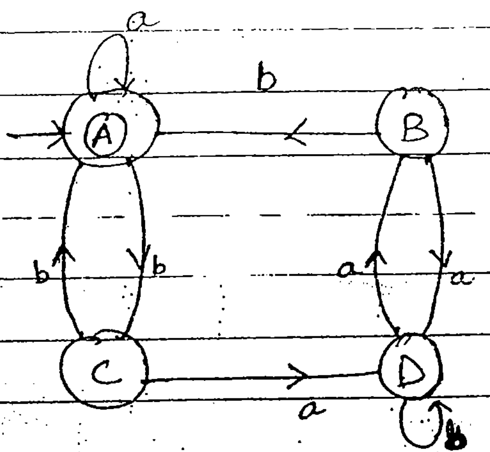


Sol^n :-

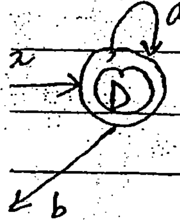
(A,P)
(C,Q)
(D,R)
(B,R)

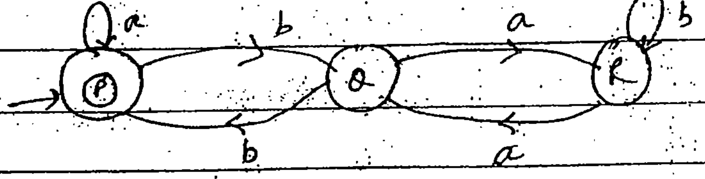
the Input

④



pairs of the





Sol^n :-

δ	a	b
(A,P)	(A,P)	(C,Q)
(C,Q)	(D,R)	(A,P)
(D,R)	(B,R)	(D,R)
(B,R)	(D,Q)	(A,R) ⟶ Not equal. F,NF

ans.

Equal states :-

Let P, Q are 2 states. then P, Q is said to be equal if both

$\delta(P,x)$, $\delta(Q,x)$ goes to final or Non final for every Input string x.

$$\begin{tikzcd}
 & & NFs & \\
\therefore\ P = Q \iff & \delta(P,x) \arrow[ur, no head] \arrow[dr, no head] & & \delta(Q,x) \arrow[ul, no head] \arrow[dl, no head] \\
 & & Fs &
\end{tikzcd}$$

Equal states are also c/d as Non Distinguishable states.

Unequal or Distinguishable states :-

2 states P, Q said to be distinguishable if $\delta(P,x)$ & $\delta(Q,x)$ goes to Final & Non final states for same I/P string x or vice versa.

$$\begin{tikzcd}
 & Fs & NFs \\
P \neq Q \iff & \delta(P,x) \arrow[u] \arrow[d] & \delta(Q,x) \arrow[u] \arrow[d] \\
 & NFs & FS
\end{tikzcd}$$

Note :-

1. Ev

2. Fina
sta

3. Tw
sta
i.e.
of

K E

cla
fin
:

δ

NOTE
Let

tates. then
both

n final for
x.

$\delta(Q,x)$

s Non

P,Q said
x) k $\delta(Q,x)$
al states
vice versa.

NFS
↑
$\delta(Q,x)$
↓
FS

Note :-

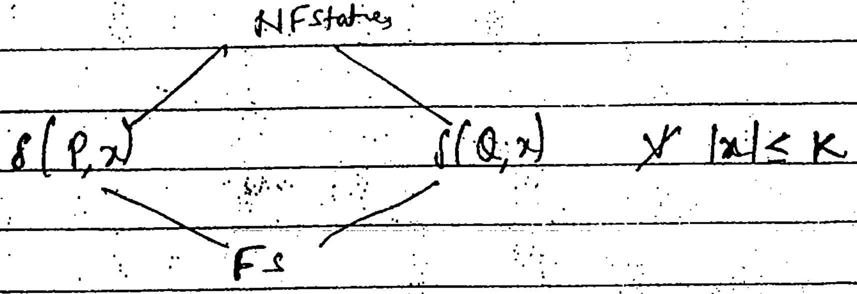
1. Every state is equal to itself.

2. Final state can't be equal to Non final state.

3. Two final states can be equal or 2 NF states can be equal.
i.e. the equality relation exist b/w pair of final state (or) pair of Non Final states.

K Equivalance Class → 2 states P,Q are said to be in K equivalance class if $\delta(P,x)$ x $\delta(Q,x)$ goes to final or non final for every $|x| \le K$

∴ P, Q ∈ K equivalance class if

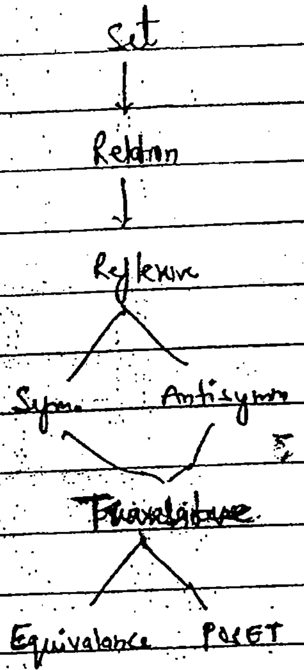


NOTE :-
Let $Q = \{q_0, q_1, q_2 --- q_n\}$ be the set of states of F.A.

R → "Equality relation"

$q_i R q_i \Leftrightarrow q_i = q_j$

$R = \{ (q_i, q_j). \mid q_i = q_j \}$.



then

Relation R is reflexive, Symm, Transitive then

⇒ R is an Equivalance R^n.

⇒ The set Q Can be partitioned into mutually exclusive subsets & those subset are equivalance classes.

No. of equivalance class is equal to No. States in Minimal F.A.

28/10/10

Optimi:

If
b

The p
whose
the
Optimi:

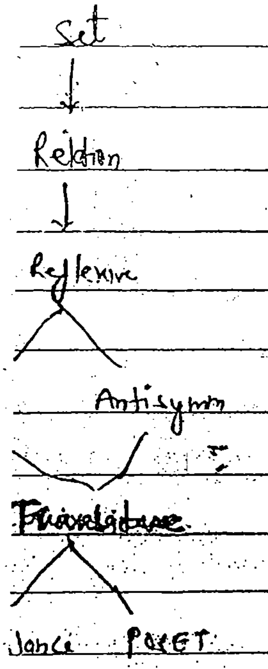
Note:

1. The
Un
wi

2* We
mini

3. FA
sta

4. If
it i
accep



l into

& there

lassete

qual to No.

28/10/10

Optimization or Minimization F.A :-

If we removed Dead state frm DFA it becomes NFA.

The process of deduction & elimination of states whose presence or absence will not affect the lang. of mission is ~~cta~~ called Optimization or Minimization of FA.

Note:-

1. The presence of absence of dead state, Unreachable states & equal states will not affect language of Mission.

2* We need to maintain dead state in minimal DFA if DFA contains Dead state.

3. FA which is free from equal & Unreachable state is called minimal FA.

4. If minimum no. of states to Con. F.A. ~~is~~ to accept language L is M then it is not possible to any other F.A to accept same L with less than 2 states.

5. Every language is accepted by only one Minimal F.A.

minimal FA is Unique But DFA is not Unique.

Method of optimization:-

Best method

1. Gate State Equivalence Method:-

Let Q is set of all states of F.A.
Let R is relation "equality of the state on the set Q.

$q_i R \; q_j \iff q_i = q_j$

$\iff \delta(q_i, x)$ & $\delta(q_j, x)$ goes to final or Nonfinal for every x.

then the relation R is Reflexive, Equivalence & Transitive.

⇒ R is equivalence Relation defined on Q.

⇒ Q can be partitioned into Mutually exclusive set & these sets are c/d equivalence class.

only

is not

F.A.

the states

o final or

ivalence &

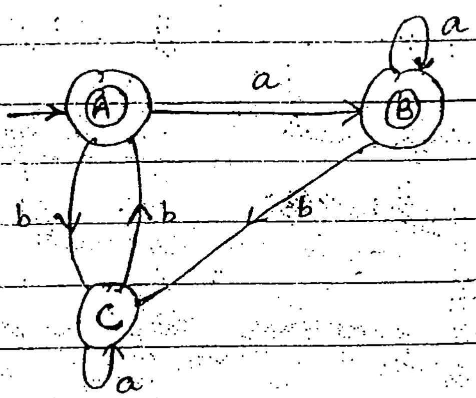
a.

ly enclusive

equivalance

⇒ No. of equivalance classes of q = No. of states in Minimal F.A.

Q :- Ex 1 :-

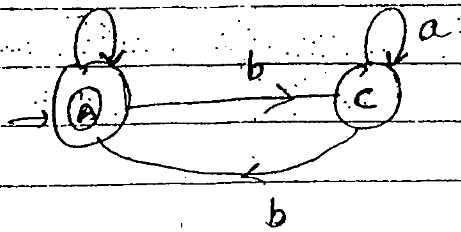


Sol:- Divide into final x Non final states.

Q_{π_0} = {(A, B), C}.
G_1 G_2

δ	a	b
A	G_1	G_2
B	G_1	G_2
	↓	↓

When the column are same the we say that 2 states are same.



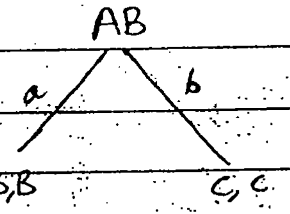
2. reading Table filling Algo :-

	A	B	C
A	AA	AB	AC
B	BA	BB	BC
C	CA	CB	CC

AA, BB, CC → equal By Reflexive.

AB = BA → Symmetric.

ⓑ		
C	X	X
	Ⓐ	ⓑ

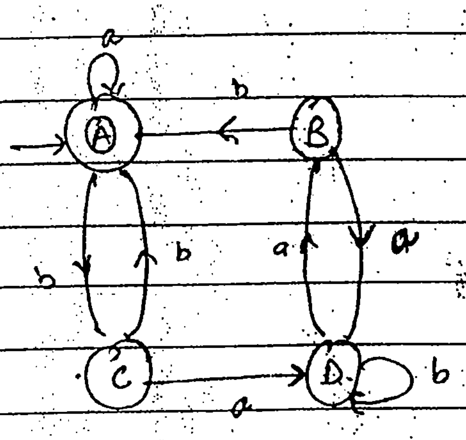


A is final & C is Non final so we put X in CA.
B " " " " " X " CB.

(B,B) & (C,C) are equal by reflexive property, so A & B also be equal.

⇒ A = B.

Q :-



No. of states in MFA is ③

1. Stat

Q =

2.

equal By Reflexive
metric.

C, C

in CA.
" CR

flexive
equal.

states
MFA is

③

1. State Equivalance State

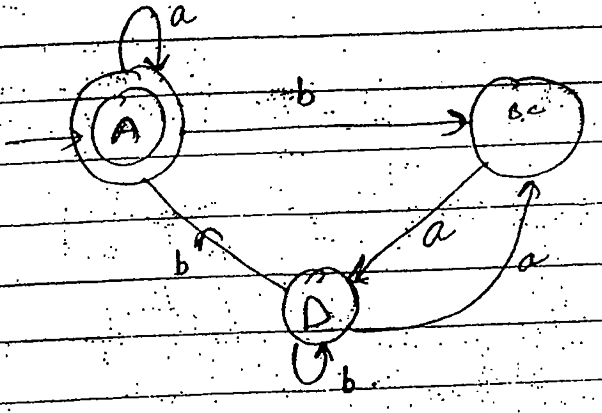
$Q = \{A, B, C, D\}$.

$\Pi_0 = \{(A), (B, C, D)\}$.

$= \{(A)(B, C), (D)\}$ → B/C B & C mapping to A But D doesnt map to A.
G_1 G_2 G_3

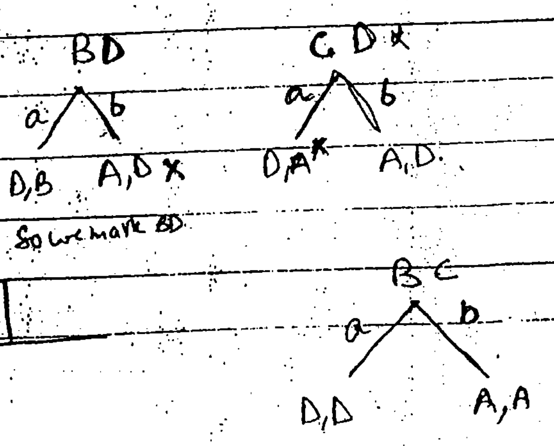
δ	a	b
B	G_3	G_1
C	G_3	G_1

So B = C



2. Table filling Method

B	X		
C	X		
D	X	X	X
	Ⓐ	B	C



So we mark BD

Q :-

δ	a	b
→ 1	6	4 ←
2	5	6
③	4 ←	5
④	3 ←	2
5	2	1.
6	1	3 ←

find which state is goes to final ~~state~~ & sep. them

↑

$\pi_0 = \{ (3,4)\ (1,2,5,6) \}$

$G_1 \quad G_2$

$\pi_1 = \{ (3,4)\ (1,6)\ (2,5) \}$

$G_1 \quad G_2 \quad G_3$

i.e 3 = 4

1 = 6

2 = 5

	a	b
(6) 1	1	3
(5) 2	2	1
(4) 3	3	2

No. of states = 3.

2	X				
③	X	X			
④	X	X	=		
5	X	=	X	X	
6	=	X	X	X	X
	1	2	③	④	5

c is goes
~~[illegible] sep. ther.~~
↑

1,2,5,6)}
G_2

(2,5)}.
G_3

Q :-

δ	a	b
$\rightarrow q_0$	q_1	q_5
q_1	q_6	q_2
$\textcircled{q_2}$	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

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

$\Pi_1 = \{\underset{G_1}{\{q_2\}}\ \underset{G_2}{\{q_3 q_5\ q_1\ q_7\}}\ \underset{G_3}{(\overset{q_0}{q_4}, q_6)}\}$

$\Pi_1 = \{\{q_2\}\ (q_1, q_7)\ (q_5, q_3)$
$(q_0, q_4, q_6)\}.$

$\Pi_1 = \{(q_2)\ (q_1\ q_7)\ (q_5, q_3)$
$(q_0, q_4)\ (q_6)$

δ	a	b
q_1	G_3	G_1
q_3	G_1	G_3
q_5	G_1	G_3
q_7	G_3	G_1

$q_1 = q_7$

$q_3 = q_5$

$q_4 = q_0$

δ	a	b
q_0	G_2	G_3
q_4	G_2	G_3
q_6	G_3	G_1

	a	b
q_4 q_0	q_1	q_3
q_7 q_1	q_6	q_2
q_2	q_0	q_2
q_5 q_3	q_2	q_6
q_6	q_6	q_0.

Q:- No. of states in minimal FA.

δ	a	b
$\rightarrow q_0$	1	0.
q_1	0	2
q_2	3	1
$\textcircled{q_3}$	3	0
q_4	3	5
q_5	6	4
q_6	5	6
q_7	6	3

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

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

$\quad G_1 \qquad G_2 \qquad G_3$ ↑

δ	a	b
q_2	G_1	G_2
q_4	G_1	G_2
q_7	G_2	G_1

δ	a	b
q_0	G_3	G_3
q_1	G_3	G_2
q_5	G_3	G_2
q_6	G_3	G_3

is wale gp ke Liye G_1 me Check Karne ki jarurat nhi Hai.

$\Rightarrow q_2 = q_4$

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

In minimal

δ	a	b
q_6 q_0	q_1	q_0
q_5 q_1	q_0	q_2
q_4 q_2	q_3	q_1
q_3	q_3	q_0.
q_7	q_0	q_3

, q_2) }

q_5, q_6) }

q_3 ↑

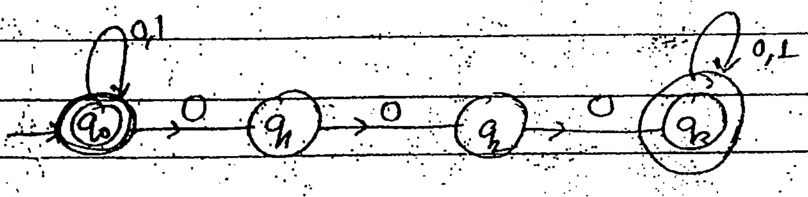
is wale gp ke Liye q_1 me check karne ki janurat nhi Hai.

{ q_1, q_5 } }.

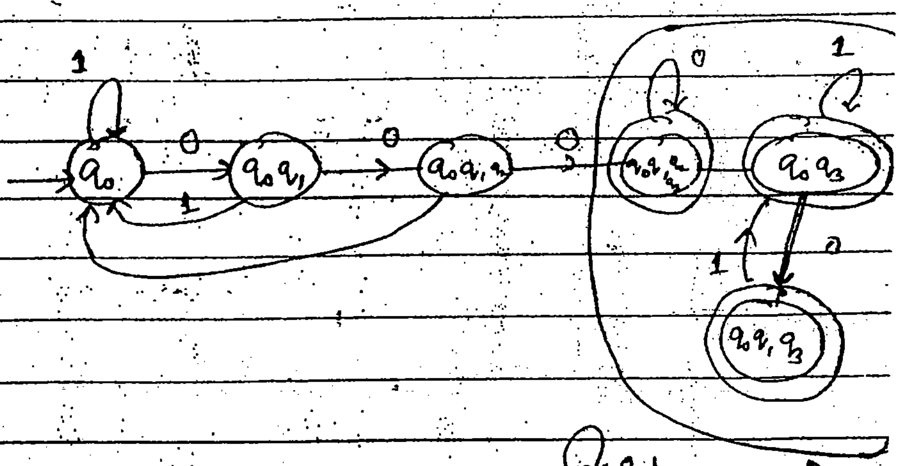
Q :- Find no of states in minimal F.A. that accepts all strings of 0 & 1 each string Contains 3 0's Consecutively

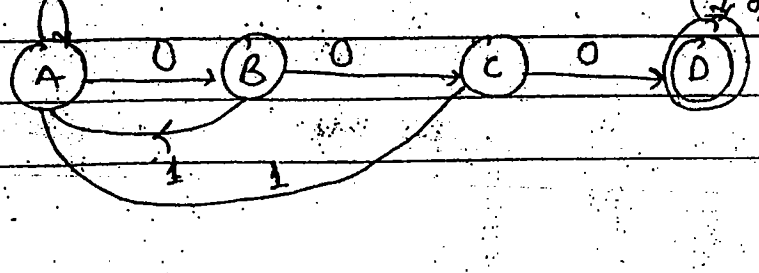
Sol:-

NFA



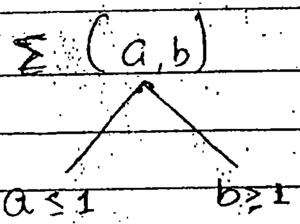
D.F.A

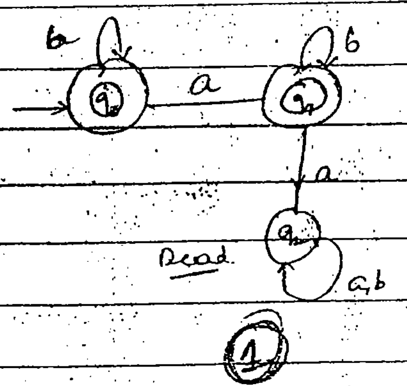


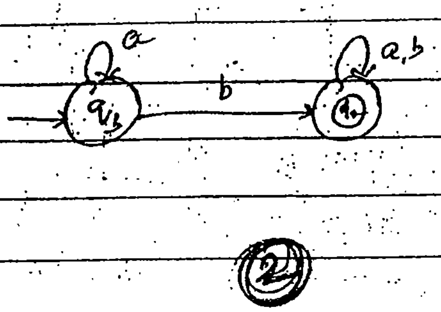


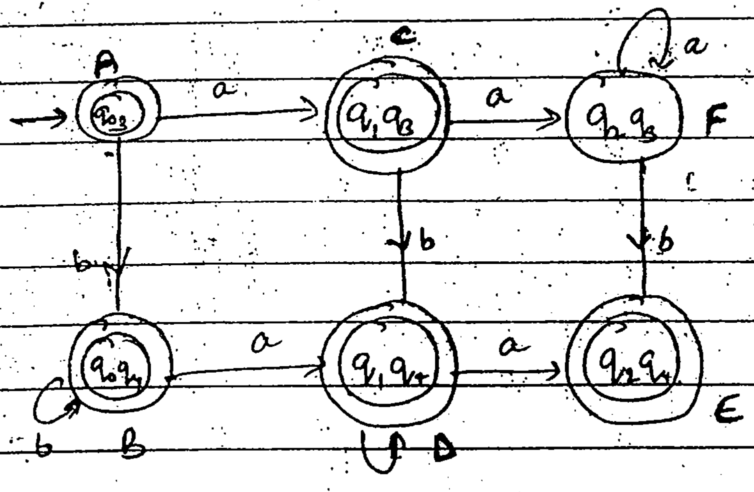
Q:- string Contains atmost One a or atleast 1 b.

Sol:- b, ab, abb, abbb.







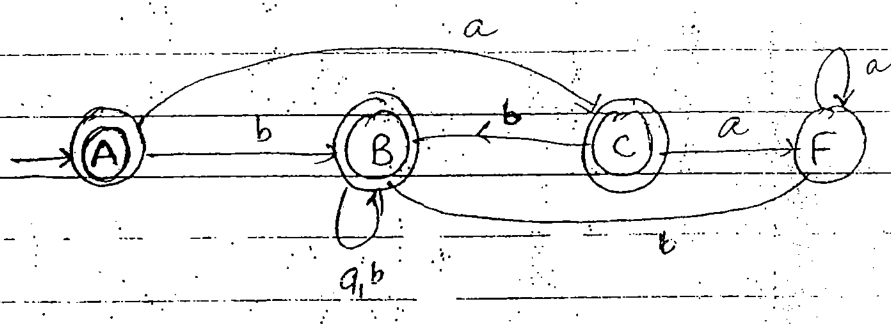


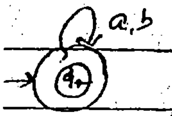
(A, B, C, D, E, F) $\pi_0 = \{A, B, C, D, E\}\ \{F\}.$

$\{BDE\}\ \{A\}, \{C\}\ \{F\}$

④ States

or atleast





Q:- Accepts all strings of a's & b's each string ~~strength~~ strats with ab & length is concurrent to 2 mod 5.

Sol:- W = ab X & |W| = ~~8~~ 2 (mod 5)

↓ 3 ↓ 5

8 states.

F

Q:- Where No. of a's divisible by 6 & No. of b's divisible by 8.

E

Sol:- $\Sigma = \{a, b\}$

|W| = 0 mod 6 0 mod 8

↓ 6 ↓ 8

48 states.

c,D,E} {F}.

}, {C} (F}

Q:- Accept string of a's, b's, c's where No. of a's div. by 2 or b's by 3 & c's not divisible by 5.

2 (~~15~~)

Sol:- $\Sigma(a, b, c)$

No. of a's $0 \bmod 2$ — $0 \bmod 3$ — $|w| \neq 0 \bmod 5$

$\downarrow$ 2 — $\downarrow$ 3 — $\Downarrow$ FA'_3 $\downarrow$ 5 states

No. of states $= 2 \times 3 \times 5$

$= 30$ Ans.

28/10/10

Acceptor .

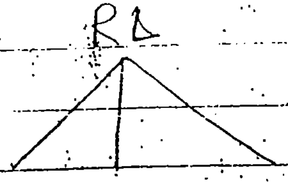
1. Th
or

1.
2.
3.
4.
5.
6.
7. L
8. L
9. L
10.

div. by
divisible

28/10/10

Regular Language



Acceptor ← FA = RE = RG

$\downarrow$ Generator $\downarrow$ Generator

75'

1. The language which is accepted by FA or generated by RE or generated by Regular Grammer is called Regular Language.

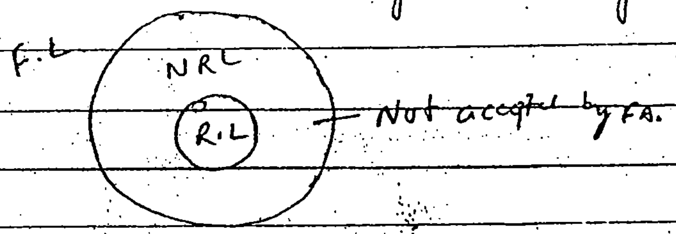
od5

states

1. $L = \emptyset$ RL
2. $L = \Sigma^*$ RL
3. Every finite language is Regular.
4. $L = \{a, b\} \rightarrow$ RL
5. $L = \{ab, aa, ba, bb\}$ Regular.
6. $L = \{a^n \mid 1 \leq n \leq 2^{10 lakhs}\}$ Regular.
7. $L = \{a^n b^n \mid n = 1, 2 \text{----} 2010^{2010 lakhs}\}$
8. $L = \{a^m b^n \mid 1 \leq m \leq 20 \quad 1 \leq n \leq 50\}$.
9. $L = \{a^n b^n \mid$ n is 10 digit prime No$\}$
10.

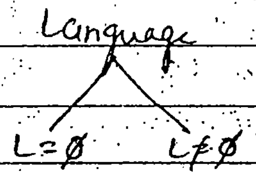
Jote :-

- A language not accepted by FA are c/d as Non regular language.



- Every language is either Regular or Non Regular.

3· No language is both regular & Non Regular



4. Every finite lang. is regular But a regular lang. need not to be finite.

5. Every Non regular lang. must be ~~Regular~~ infinite But an infinite lang. can be Regular or Non Regular.

Q:- Wh

a) $L =$

b) $L = \{$

c) $L = \{$

d) $L = \{$

e) $L = \{$

f)* $L=\{a^m b^n$

g)* $L = \{ a$

h)

i) $L = \{$

j) $L =$

k) $L =$

l) $L =$

m) $L =$

n) $L =$

o) $L =$

p) $L =$

q) $L =$

r) $L =$

s)

T).

FA are language.

by FA.

egular or

* Non Regular

But a

t to be

ust be ~~Regular~~

lang. can be

n.

Q:- Which of following is Regular.

a) $L = \{a^m b^n / m \geq 0, n \geq 0\}$ RL

b) $L = \{a^m b^n / m \geq 2, n \geq 3\}$. RL

c) $L = \{a^m b^n / 1 \leq m < 10 \quad 2 \leq n \leq 10\}$. RL

d) $L = \{a^m b^n / mn$ is finite $\}$ RL

e) $L = \{a^m b^n / m+n$ is finite $\}$ RL

f)* $L = \{a^m b^n / m+n /$ Odd $\}$ R.L.

g)* $L = \{a^m b^n / m+n$ even $\}$. R.L

h) $1 \leq m < n < 2\}$. R.L.

i) $L = \{a^m b^n / m \textcircled{=} n\}$ NRL

j) $L = \{a^m b^n / m \neq n\}$ NRL

k) $L = \{a^m b^n / m < n\}$ NRL

l) $L = \{a^m b^n / m > n\}$ NRL

m) $L = \{a^m b^n / m = 2n\}$ NRL

n) $L = \{a^m b^n / m = r \bmod n\}$. NRL

o) $L = \{a^m b^n / \gcd(m,n) = 1\}$ NRL

p) $L = \{a^m b^n /$ m divided by m $\}$. NRL

q) $L = \{a^m b^n c^p / m = n = p\}$. NRL

r) $L = \{a^m b^n c^p / m \leq n \; p \geq 1\}$ NRL

s) $\{a^m b^n c^p / m \geq n = p\}$ NRL

T) $\{a^m b^n c^p / m + n = p\}$ NRL

29/10/10

Properties :-

1. Every subset of a regular set need not be regular.

2. Every finite subset of regular set is Regular.

3. Every subset of NRL set need not be Non Regular.

4. Every fininite subset of NRL set, is Regular.

5. A superset of Regular set need not be Regular.

6. A superset of NR set need not be Non Regular.

7. Every language has a Regular subset.

8. Finite $\left\{ \begin{matrix} \text{Union} \\ \text{Intersection} \end{matrix} \right\}$ of Regular sets is always Regular.

Infinite $\left\{ \begin{matrix} \cup \\ \cap \end{matrix} \right\}$ of Regular sets need not be Regular.

Jha Bhi finite Word ka use Hoga wo Regular Hoga.

$*$

$+$

$\cdot$

Regular

regul

An

3 of

expr

Ex:- ①

NOTE:-

1. If

g

ex :-

2. Ever

But

tha

one RE

3. Alge

Regu

(0

ed not

t is

not be

is Regular.

need not

not be

lar Subset.

ts is

need not

e Hoga wo

---

* Kleen Closure → Regular expressions always

\+ Union (starts from 0). generates Regular

. Concatination. Language.

Regular Expressions :-

An expression that represent regular language is called R.E.

or.

An expression of strings. which is constructed using 3 operators {*, +, .}. is called Regular expression.

Ex :- ① $r = a+b$, ② $r = ab$. ③ $r = a^* + b$.

NOTE :-

1\. If r is Regular Exp. then L(r) is language generated by r.

ex :- $r = a+b$ then $L(r) = (a, b)$.

RE → LANG

RE ↗ ↑ ↑

RE RE

2\. Every R.E. generates Only one language. But A language Can be generated by more than One form of R.E. {R.E. is not Unique

One RE one Language But these language may be same.

3\. Algebric exp. rep. numerical value while Regular expression always represents Regular Language.

$(0+1)0 = 0$
A.E.

$(0+1)0 = \{00, 10\}$
R.E

Regular Operators :-

$*$, $+$, $\cdot$ are called regular Operators.

$*$ → Highest, $+$ → Lowest

e.g. $(a + \underset{③}{b} \cdot \underset{②}{c})^{*}$ — ①

Types of Regular Expression :-

RE

- Restricted $(*, +, \cdot)$ → Directly rep. into FA.
- Semi ~~unrestricted~~ restricted $(*, +, \cdot, \cap)$
- Unrestricted. $(+, +, \cdot, \cap, \sim)$ ($\sim$ → Complement)

Semi restricted and Unrestricted → first we have to Convert into restricted then into FA.

Ex.	Regular Exp.	R.L.
1.	$r = \phi$	$L = \{\ \}$.
2.	$r = \epsilon$	$L = \{\epsilon\}$.
3.	$r = a$	$L = \{a\}$
4.	$r = \{a+b\}$	$L = \{a, b\}$.
5.	$r = \{a+b+c\}$	$L = \{a, b, c\}$.
6.	$r = \{w_1 + w_2 + w_3 + \text{--} w_n\}$	$L = \{w_1, w_2 \text{ --- } w_n\}$.
7	$r = \{ab\}$	$L = \{ab\}$.
8	$r = (a+b)a$	$L = (aa, bb)$

1–3: reg. Exp without Regular operators.

1. If r is a R.E then r^*, r^+ are also regular expression.

$$L(r^*) = [L(r)]^*$$
$$L(r^+) = [L(r)]^+$$

$*$ Kleen closure

$+$ +ve Closure.

2. If r is any R.E. then $r^* = \{\epsilon, r, rr, rrr \cdots$
& $r^+ = \{r, rr, rrr \ldots\}$.

$* = \{0, 1, 2, 3 \ldots\}$, $+ = \{1, 2, 3, 4 \ldots\}$.

3. If r is any R.E. then $r^*.r = r^+ = r.r^*$

$$r^* r^+ = r^+ = r^+ r^*$$
$$r^* r^* = r^*$$
$$r^+ r^+ = r^+$$
$$(r^+)^* = r^* \qquad r^{*^{*^{*\cdots}}} = r^* \quad r^{+^{+^{+^{+\cdots}}}} = r^+$$
$$(r^*)^+ = r^*$$

$$r^{*^{+^{*^{+}}}} = r^* \qquad r^{+^{*^{+^{*}}}} = r^*$$

4. If $r = \phi$ then

$\{\epsilon, \phi, \phi\phi, \phi\phi\phi\}$ $r^* = \epsilon$ $\qquad \epsilon \cup \phi \cup \phi \cup \phi = \epsilon$

$\{\phi, \phi\phi, \phi\phi\phi\}$ $r^+ = \phi$ $\qquad \phi \cup \phi$.

5. If $r = \epsilon$ then

$r^* = \epsilon$ $\qquad \{\epsilon, \epsilon\epsilon, \epsilon\epsilon\epsilon, \ldots$

$r^+ = \epsilon$ $\qquad (\epsilon, \epsilon\epsilon, \epsilon\epsilon\epsilon, \ldots$

n/s.

stricted.

+, ;, ∩, ~)

Complement

to Convert into

then into FA.

}. } reg. Exp

ε}. without

{a} Regular operators.

}

}.

, w_1 -- w_n}.

ab}.

aa, bb}.

ba.

r^* & r^+ represents infinite languages except for $r = \emptyset$ or $r = \epsilon$.

If r is a regular expression such that $r = r^+$ then

$$r = \emptyset ; \epsilon$$

→ If r is regular exp. such that $r^* = r^+$ then

$$r = \epsilon$$

→★ If r_1, r_2 are 2 regular expressions then both $r_1 + r_2$ & $r_1 . r_2$ are also regular expression.

$$L(r_1) + L(r_2) = L(r_1 + r_2)$$
$$L(r_1) . L(r_2) = L(r_1 . r_2).$$

★ If r_1, r_2 be 2 R.E. than

$$(r_1 + r_2)^* = (r_1^* + r_2^*)^* = (r_1^* + r_2)^* = (r_1 + r_2^*)^* = (r_1^* r_2^*)^*.$$

Shifting

1. If

Q:- Wh

a) Regul

b) R.E.

c) Regul

d) Non

Q:- Which

a) RE (

b) Every

c) R.E.

d) R.E.

Q:- Whi

a)

b)

c)

(d)

ages except

uch that

t

expressions
r_2 are

k.

Shifting Prop: Imp

1. If r_1, r_2 are 2 R.E than.

$$(r_1 r_2)^* r_1 = r_1 (r_2 r_1)^*$$

Q:- Which of the following statement is Correct.

a) Regular Exp. always generates a Language.

b) R.E. Can represent a Non Regular L.

c) Regular exp. Can rep. Non Regular or Regular L.

d) None of the above. (✓) R.E. represent Regular Lang.

Q:- Which of following statement is Correct.

a) RE Can generates more than One R.L. x
b) Every R.E generated by only One R.E. x
c) R.E. is Unique.
d) R.E. represents Unique regular Language (✓)

Q:- Which of following statement is True.

a) $r^* = r(*)$
b) $r^* = \bigcup_{i\geq 1} r^i$
c) $r^+ = \bigcup_{i\geq 0} r^i$
d) $r^*(r^+)r^* = r^*$

Q :- find identical R.E.

[✓ a)] 0^* — $\epsilon, 0, 00, 000$
b) $(00)^*$ — $\epsilon, 00, 0000,$
[✓ c)] $0^*(0+\epsilon)$ — $0^*0 + 0^*\epsilon = 0^+ + 0^* = 0^*$
d) $(00)^*0$ — $0, 000, 00000$

$\Rightarrow \boxed{a = c}$ Ans

Q :- Correct statement is.

a) $(0\cdot1)^* 1 = 0(11)^*$
[✓ b)] $(01)^* 0 = 0(10)^*$ By property shifting
c) $(01)^* 0^* = 0(10)^*$
d)

Q :- Correct statement.

a) $r_1^* + r_2^* = (r_1 r_2)^*$
b) $(r_1^* + r_2)^* = (r_1 r_2)^+$
c) $(r_1 + r_2^*)^* = (r_1^* r_2)^+$
[✓ d)] $(r_1^* + r_2)^* = (r_1^* r_2^*)^*$

NOTE :-
ϵ =

① Σ^*
② Σ^+
③ Σ^n

Examples

Q :- Co
① incl
② ex

Sol :-

Q :- R.

Sol :-

$0^* = 0^*$

y shifting

NOTE :-

$\epsilon = \{a, b\} = a+b$

① $\Sigma^* = \{a, b\}^* = (a+b)^* = \{\epsilon, a, b \ldots\}.$

② $\Sigma^+ = \{a, b\}^+ = (a+b)^+ = \{a, b, aa, ab \ldots$

③ $\Sigma^n = (a+b)^n = \omega \in \{ab\}^* \; |\omega| = n\}.$

Examples :-

Q :- Cons R.E. that generates all strings of a's
① including ϵ
② excluding ϵ.

Sol :- $\Sigma = \{a, b\}$ → Generate all string of a & b.

$\Rightarrow \Sigma^* = (a+b)^*$ $\quad \Sigma^+ = (a+b)^+$

Q :- R.E. generate all string of a'b every string starts with a.

Sol :- $W = a \underset{(a+b)^*}{\underset{\downarrow}{X}}$.

$r = a(a+b)^*$.

Q:- All string of a, b, where every string ends with ba.

Sol:- $W = X\,ba$

$(a+b)^* ba$ Ans

Q:- R.E. Containing all string of a, b where every string start & end with a.

Sol:- $W = aXa$, a — b/c a also start & end with a

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

Q:- Where every string start & end with same symbol.

Sol:- $W = aXa$, bXb

$r = \{a(a+b)^*a + a\} + (b(a+b)^*b + b$

Q:- Where every string start & end with diff. symbols.

Sol:- $W = axb$, bxa

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

string

re every

with same

$)^* b + b$

nd with

x Ans

Q :- All strings of a,b where each string contains exactly 2 a's.

Sol :- $W = X a X a X$

$r = b^* a b^* a b^*$

Q :- Where each string contains atleast 2 a's.

Sol :- $\Sigma = \{a, b\}$

$W = X a X a X$

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

Q :- Which contains atmost 2 a's.

Sol :- $W \Rightarrow a = 0, 1, 2$

$W = X (a+\epsilon) X (a+\epsilon) X$

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

Q :- Where each string contains No. of a's in string is even.

$|w|_a = 0 \pmod 2 = 0, 2, 4, 6, 8, 10, 12$

start frm 2.

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

2,4,6,8 a's.

↳ b/c b, bb, bbb also contains 0 a's

Q:- Where No. a's is Concurrent to 1mod2

Sol:- No. of a's is Odd.

$|W|_a = 1 \ (\text{mod } 2).$

$= 1, 3, 5, 7, 9.$

$r = (b^* a b^* a b^*)^* b^* a b^*$

$(b^* a a b^*)$ kyoi lenge to sari string Generate ni kar Payega.

Q:- Where ~~nay~~ ~~ax~~ 4th Symbol from left end is always b.

Sol:- $\Sigma = \{a, b\}.$

$W = XXXb\ \text{-------}$

$\downarrow$

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

$(a+b)^3 b\, (a+b)^*$

Q:- Where 4 Sym from Right end is always a.

Sol:- $W = \text{-------}\, a\, XXX$

$= (a+b)^* a\, (a+b)^3$

NOTE :-

Long

1. n^{th}

2. n^{th} S

Q:- Whe

Sol:-

If

Q:- Wh

Sol:-

Q:- Le

Sol:-

If 11

NOTE :-

Language $\Sigma = (a,b)$	R.L.
1. n^{th} Symbol frm Left end.	$(a+b)^{n-1}\left(\frac{a}{b}\right)(a+b)^*$ $\overset{\wedge}{}$ a or b jo bhi Que me pucha ho,
2. n^{th} Symbol frm Right end.	$(a+b)^*\left(\frac{a}{b}\right)(a+b)^{n-1}$

Q:- Where length of string is exactly 3.

Sol:- $|w| = 3$

$w = x, x\, x$

$= (a+b)(a+b)(a+b) = (a+b)^3.$

$$\boxed{\text{If } |w| = n \text{ then } \quad r = (a+b)^n}$$

$w \geq n$

$r = (a+b)^n (a+b)^*$

Q:- Where length of string is aleast 3.

Sol:- $(a+b)^3 (a+b)^*$

Q:- Length of String is atmost 3.

Sol:- $(a+b+\epsilon)(a+b+\epsilon)(a+b+\epsilon)$

$(a+b+\epsilon)^3.$

If $|w| \leq n$ then $r = (a+b+\epsilon)^n.$

Q:- Where length of string is even.

Sol:- $\Sigma = (a,b)$

$|W| = 0 \pmod 2$

$= 0, 2, 4, 6, 8$

$\downarrow$

$r = \left[(a+b)^2\right]^*$

Q:- Where length of string is odd.

Sol:- $|W| = 1 \pmod 2$

$= 1, 3, 5, 7 \ldots$

Even + Odd = Odd.

$r = (a+b)\left[(a+b)^2\right]^*$

Q:- Length of string is divisible by 3.

Sol:- $|W| = 0 \bmod 3$

$= 0, 3, 6, 9, \ldots$

$\downarrow$

$\Sigma = \left[(a+b)^3\right]^*$

Q:- Len

Sol:-

NOTE:

Q:- Eac

(b)

Sol (a)

r =

Q :- Length of string is Concurrent to 2 mod 3.

Sol :- $|W| = 2 \bmod 3$

2, 5, 8, 11 - - - - -

$$(a+b)^2 \left[(a+b)^3\right]^*$$

NOTE :-

$\Sigma = (a, b)$

$|W| = r \bmod n$

$r = (a+b)^r \left[(a+b)^n\right]^*$

Q :- Each string (a) start with a length is odd.
(b) start with b length is even.

y 3.

Sol (a) $|W| = a, ab, aa$

$= aX$
↳ even

$r = a\left[(a+b)^2\right]^*$

(b) $|W| = \epsilon, ba, bb$

$r = b\left[(a+b)^2\right]^*(a+b)$.

Q:- R.E for following Languages.

1) $L = \{ a^m b^n / m,n \geq 0 \}$ $= a^* b^*$

2) $L = \{ a^m b^n / m \geq 1\, n \geq 0 \}$. $= a a^* b^* = a^+ b^*$

3) $L = a^m b^n / m \geq 0\ n \geq 1 \}$ $= a^* b b^* = a^* b^+$

4) $L = a^m b^n / m \geq 0\ n \geq 3 \}$ $= a^* bbbb^* = a^* bbb^+$

5) $m \geq 1\ n \geq 0 \}$ $a a^+ b^*$

6) $m \geq 3,\ n \geq 2 \}$ $aaa^+ bb^+$

7) $m \geq 1\ n \geq 1 \}$ $a^+ b^+$

8)

9)

10)

Imp

Q*:- $L = \{ a^m b^n / m+n = even \}$.

Sol:- m+n = even.

m = even	m = Odd
n = even	n = Odd.
$m = 2x$	$m = 2x+1$
$n = 2x$	$n = 2x+1$
$a^m b^n$	$a^m b^n$
$a^{2x} b^{2x}$	$a^{2x+1} b^{2x+1}$
$(aa)^x (bb)^x$	$(aa^x)\, a\, (bb)^x b$
$(aa)^* (bb)^*$	$(aa)^* a\, (bb)^* b$

Q:*-

Sol:-

a

(a

(

⇒

Q:-

Sol:-

$a^+ (a$

$$\Rightarrow r = (aa)^*(bb)^* + (aa)^* a (bb)^* b.$$

$= a^+b^*$
a^*b^+
a^*bbb^+
aa^+b^*
$aa^+ bb^+$
$+b^+$

Q:*- $L = \{ a^m b^n \mid m+n = \text{odd} \}$.

Sol:- $m+n=\text{Odd}$

$m = \text{even}$ $\quad$ $m = \text{Odd}$
$n = \text{Odd}$ $\quad$ $n = \text{even}$

$a^{2x} b^{2x+1}$ $\qquad$ $a^{2x+1} b^{2x}$
$(aa)^x (bb)^x b$ $\qquad$ $(aa)^x a (bb)^x$
$(aa)^*(bb)^* b$ $\qquad$ $(aa)^* a (bb)^*$

$$\Rightarrow r = (aa)^*(bb)^* b + (aa)^* a (bb)^*$$

Q:- R.E all strings of a & b. every string start with a & doesn't Consecutive b's.

Why we can't include ε productions in this

Sol:- $a^+ = a, aa, aaa \ldots$
$(ab)^+ = ab, abab, ababab.$

It can't

$a^+(ab)^+ \leftarrow \boxed{(a+ab)^+}$ Ans

aab

Q:- R.E. generates all String of a,b where every string donot Contain 2 Consecutive a's & b's.

Sol:- $(ab)^* = \epsilon, ab, abab \ldots$

$(ba)^* = \epsilon, ba, baba \ldots$

$(b+\epsilon)\ (ab)^*\ (a+\epsilon)$

OR

$(a+\epsilon)\ (ba)^*\ (b+\epsilon).$

OR

$(ab)^* + (ba)^*$

Q:- Reg. expression for $L = \{a^n /\ n$ div. by 2 or 3 or 5$\}$.

$n=5$

Sol:- $n = 0 \bmod 2 \quad = (aa)^*$

$n = 0 \bmod 3 \quad = (aaa)^*$

$n = 5 \quad = (aaaaa)^*$

$$r = \boxed{(aa)^* + (aaa)^* + (aaaaa)^*}$$

$(aa)^* . (aaa)^* . (aaaa)^*$

Q:- Whi

Set

Subs

Sol:- But

Contai

So

mod

is

Q:- S

a) |

b) |

c) (

d)

Q:- Y

then

a)

b)

c)

d)

s =

t =

where

Consecutive

Q:- Which of following R.E. over $\{0, 1\}$ denotes Set of all strings not Containing 100 as substring.

a) $0^*(1+0)^*$ 100

b) $0^*\,101\,0^*$ 100

c) 0^*1^*01 x

d) $0^*(10+1)^*$ x

Sol:- But c doesnot Contain ϵ as substring so most appropriate ans is (d).

Q:- String 1101 doesnt belong to set rep by.

a) $110^*(0+1)$ → 1101

b) $1(0+1)^*101$ → 1101

c) $(10)^*(01)^*(00+11)^*$ x

d) $(00+(11)^*0)^*$ → 1101

div. by 2 or 3 or 5?

$n=5$

$)^*$

$a)^*$

$aa)^*$

$+(aaaaa)^*$

1^*

Q:- $r = 1(1+0)^*$ & $s = (11)^*0$ $t = 1^*0$ then which of the following is true.

a) $L(s) \subset L(r)$ & $L(s) \subset L(t)$.

b) $L(s) \subset L(t)$ & $L(r) \subset L(s)$

c) $L(t) \subset L(s)$ & $L(r) \subset L(s)$

d) $L(t) \subset L(r)$. & $L(s) \subset L(r)$

$s = \{10, 110, 1110, \ldots\}$

$t = \{0, 10, 110, 1110 \ldots\}$.

Q:- $A = (01+1)^*$

$B = ((01)^* 1^*)^*$ Which of following is true.

011, (01+1)(01+1)

01011

0101011

Sol:- a) $A \subset B$

b) $A \supset B$

~~c)~~ $A = B$ { Direct formulae }

d) None.

Q:- $0^*(10^*)^*$ denote same set as.

a) $(1^*0)^* 1^*$

b) $0 + (0+10)^*$

c) $(0+1)^* 10 (0+1)^*$ x b/c above string doesn't Contain 10 as substring.

~~d)~~ None.

Q:- No. of strings of length < 3 generated by Regular expression $(a+ba)^*$

L=0, L=1 L=2 L=3

↓ ↓ ↓ ↓

1 1 2 3.

ϵ a aa, ba, aaa

aba

baa

a) 5

~~b)~~ 7

c) 9

d) 11

e) None

Q:- L

Sol:-

Q:- I

lang

Q:- L ≤ 3 generated by R.E $(a+ab)^* b (a+b)^*$

Sol:- L = 0 = 0

L→ 1 → 1

L⇒ 2 = 3 , ab, ba, bb

L⇒ 3 = 7

a) 7
b) 9
c) 11
d) 12

Q:- No. of states in min F.A accepting the language gen. by

(0+1) (0+1) (0+1) (0+1) - - - - n times.

$$\Downarrow$$

$$(0+1)^n$$

$$\Downarrow$$

By formulae for DFA having n we hav n+2 state.

a) n
b) n+1
c) n+2
d) None.

following

s.

doesn't

ubstring.

generated

i+ba)*

ne

Equivalance b/w F.A. & Regular Expression.

Conversion FA. to RE →

1. Adrin's Lemma {used for DFA, NFA}
2. State elimination Method. (DFA, NFA, ∈NFA, TG transition graph)

ARDEN's Lemma :-

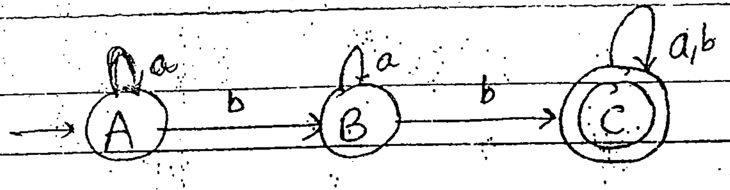
→ Used only for DFA & NFA

→ If P, Q are 2 regular exp. then

a) $\boxed{R = Q + RP}$ has unique sol^n^ if P doesn't contain ∈.

i.e. sol^n^ is $R = QP^*$

b) R = Q + RP have infitely many sol^n^ if P contains ∈.

Ex :- 

Q:-

$A = Aa + \epsilon$ → Initial state ke sath ϵ bhi add kyte hai

Sol:-

$B = Ab + Ba$

$C = bB + C(a+b)$

$$\underset{R}{A} = \underset{R}{A}\underset{P}{a} + \underset{Q}{\epsilon} = \epsilon a^* = a^*$$

$$B = \underbrace{a^*b}_{Q} + \underset{R}{B}\underset{P}{a} = a^*ba^*$$

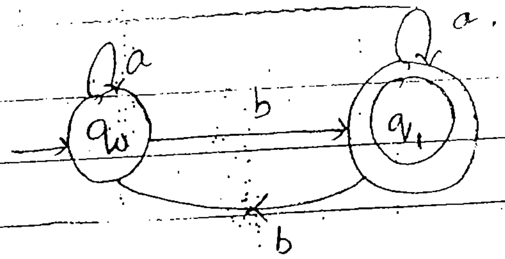
$$C = \underbrace{a^*ba^*b}_{Q} + \underset{R}{C}\underset{P}{(a+b)} = a^*ba^*b(a+b)^*$$

r.e. :

$r = C$ b/c C is final state

$r = a^*ba^*b(a+b)^*$.

Q:-



ε rhi
ite hai

Sol:- $q_0 = \underset{R\,P}{q_0 a} + \underset{Q}{q_1 b + \epsilon}$

$q_0 = (q_1 b + \epsilon) a^*$

$q_1 = q_1 a + q_0 b$

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

$= q_1 a + q_1 b a^* + a^*$

$=$

$q_1 = \underset{R}{q_1} (\underset{P}{a + ba^*}) + \underset{Q}{a^*}$

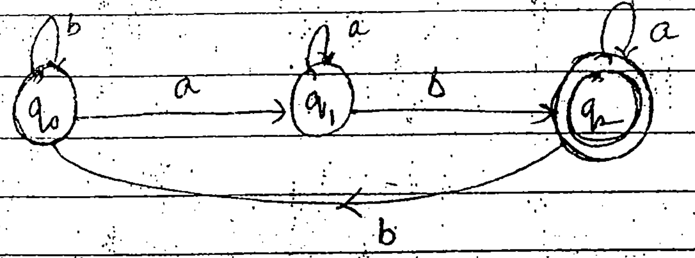
$(a+b)^*$

r.e. $= q_1 = a^* (a + ba^*)^*$ Ans

state

$q_1 = (a + ba^* b)^* ba^*$

Q: ④



Sol:- $q_0 = q_0 b + q_2 b + \epsilon$

$q_0 = (q_2 b + \epsilon) b^*$

$= q_2 b^+ + b^*$

$q_1 = q_0 a + q_1 a$

$= q_2 b^+ a^+ + b^* a^+$

$q_2 = q_1 b + q_2 a$

$q_2 = q_1 b a^*$

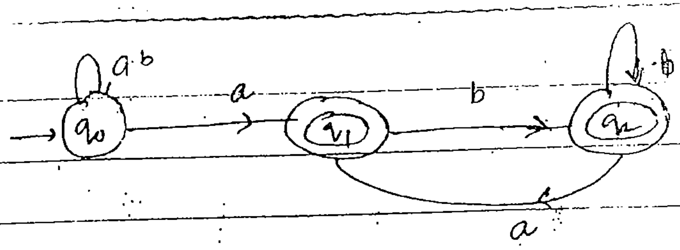
$q_2 = (b^* a^+ + q_2 b^+ a^+) b a^*$

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

Q:-

q

Q:-



$q_0 = q_0 (a+b) + \epsilon$

$q_0 = (a+b)^*$

$q_1 = q_0 a + q_2 a$

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

$q_2 = q_1 b + q_2 b$

$q_2 = q_1 b b^*$

$q_1 = (a+b)^* a + q_1 b^+ a$

$q_1 = (a+b)^* a (b^+ a)^*$

$q_2 = (a+b)^* a (b^+ a)^* b^+$

ba^*

$r = q_1 + q_2$

$a^*)^*$

State elimination Method:- used for FA as well as transition Graph < Any final / Any Initial

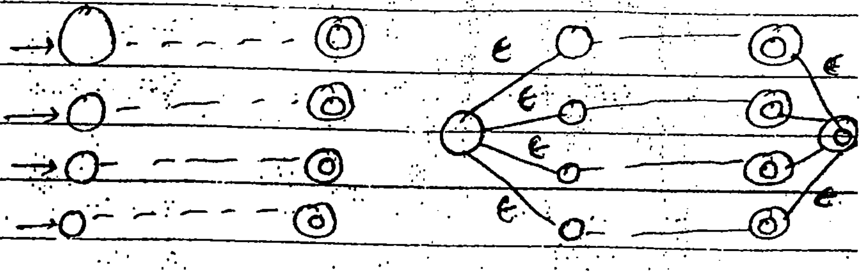
→ TG can have more than one initial state.

→ In TG the edge labels can be from Σ^*.

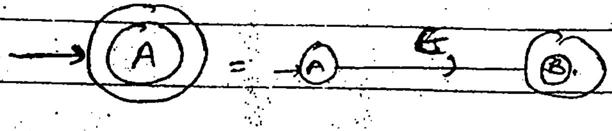
$\delta : q \times \Sigma^* \to 2^Q$ is a transition f^n in TG.

Algo:-

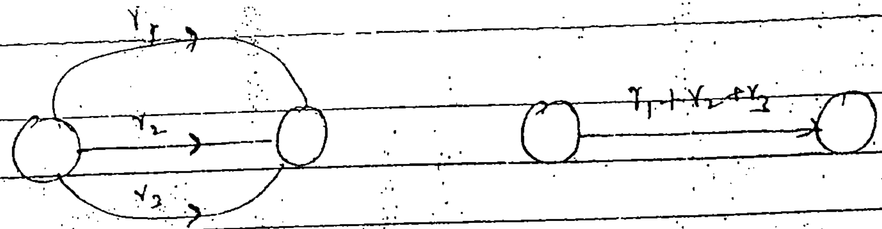
1) Simplify TG such that it has only one initial state & only one final state.



2) Simplify sys. to have diff. initial & final states.

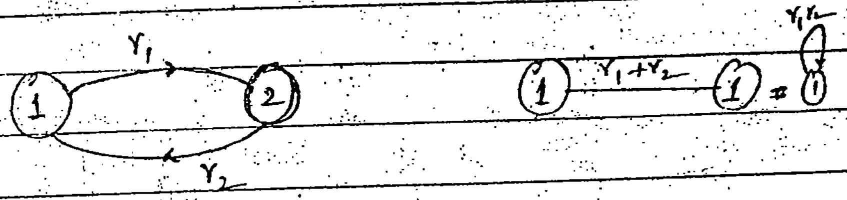


3) If there exist parallel edges b/w pair of states then.

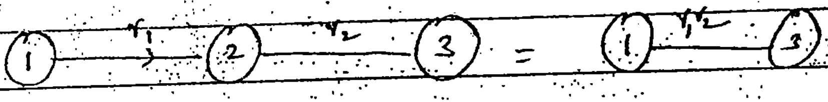


4) eliminate state 2 fm following graphs

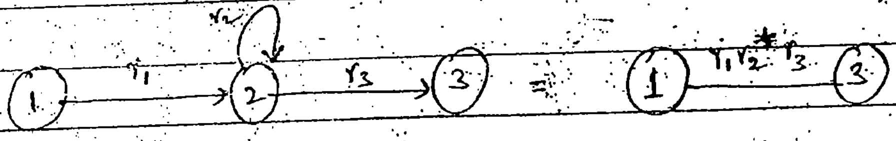
(a)



b)

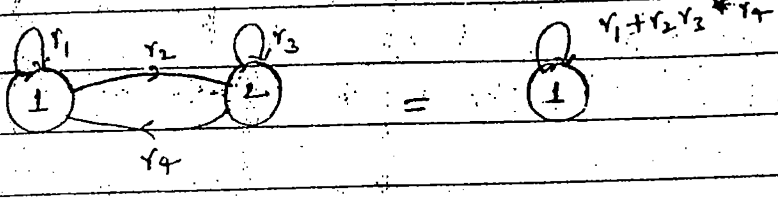


c)

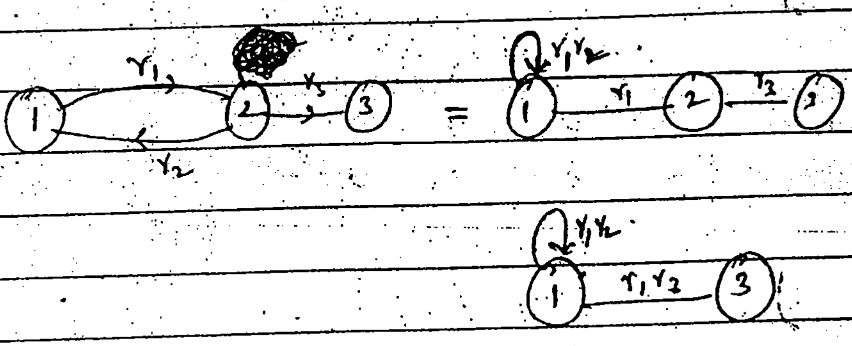


d)

e)



f)



g)

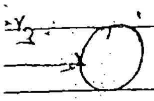
h)

Step 5

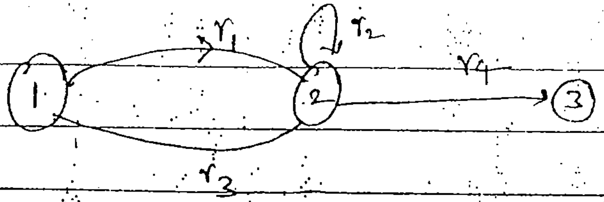
elin

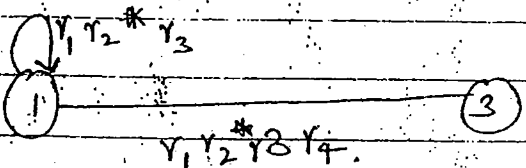
tak

①

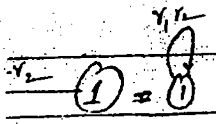


g)

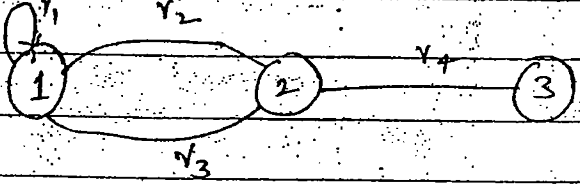


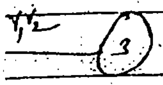


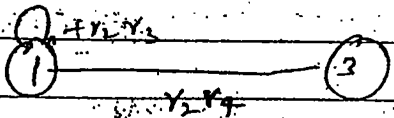
graphs

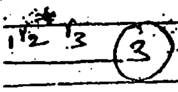


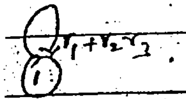
h)









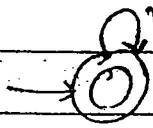


Step 5 :-

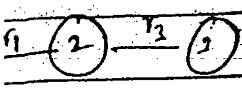
$r_3^* r_4$

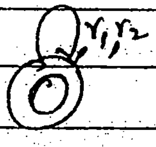
Continue the process of state elimination untill the transition G takes any one of the following form.

①

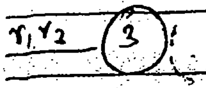


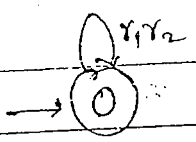
⇒ r^*



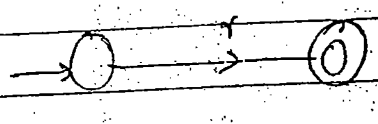


⇒ $(r_1+r_2)^*$

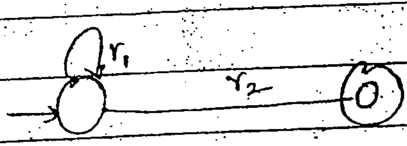


③ 

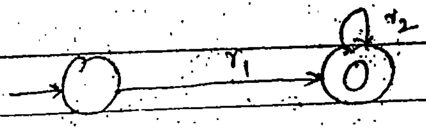
$= (r_1 r_2)^*$

④ 

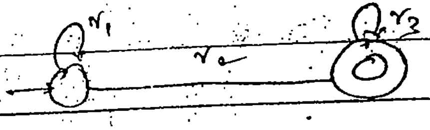
RE = r

⑤ 

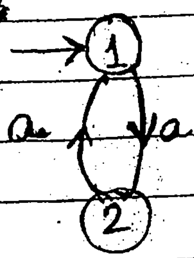
r.e = $r_1^* r_2$

⑥ 

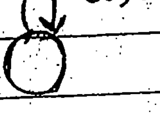
$= r_1 r_2^*$

⑦ 

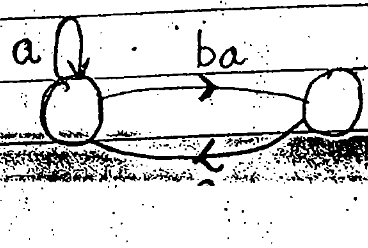
$= r_1^* r_2 r_3^*$

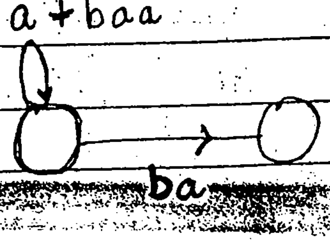
* 

⇒ $(aa)^*$

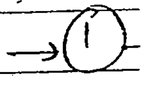
⇒ 

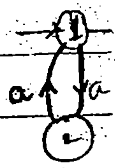
⇒ $(a+b)^*$

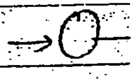
\# 

⇒ 

Examples :-

① 

② 

③ 

④ 

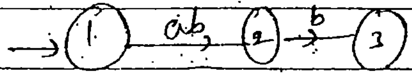
(a+

⑤

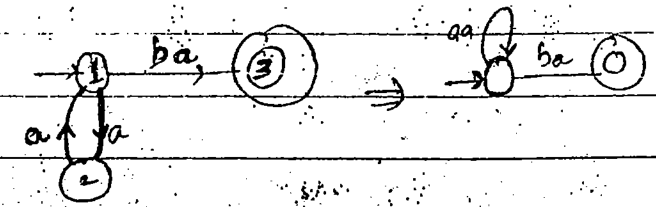
⑥ 

⑦

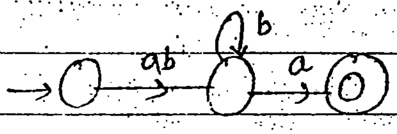
Examples :-

①


$r = abb$

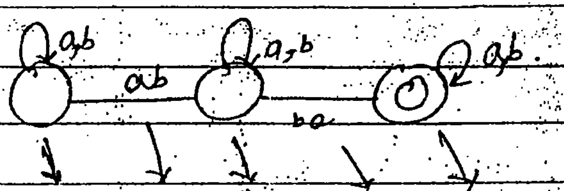
②


$r = (aa)^* ba.$

③


$r = abb^* a.$

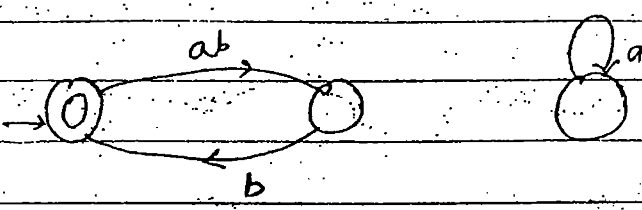
$_1^* r_2$

④


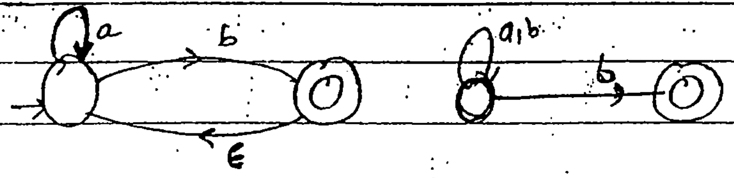
$(a+b)^* \; ab \; (a+b)^* \; ba \; (a+b)^*$

$*$

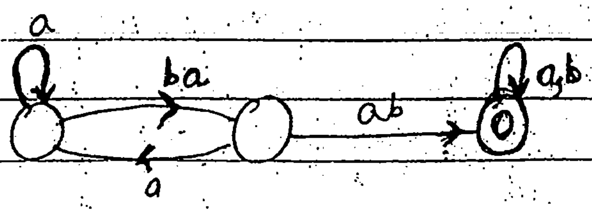
r_3^*

⑤


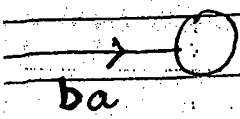
$= (abb)^*$

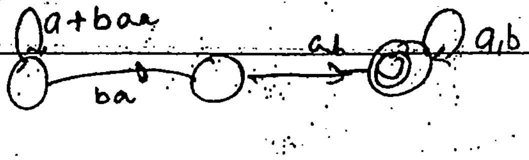
⑥


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

⑦


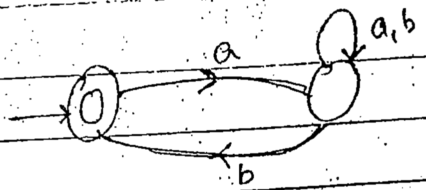
aa

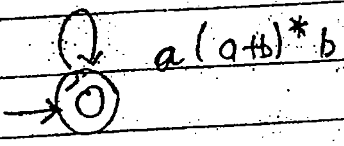




$= (a+baa)^* ba\, ab$
$(a+b)^*$

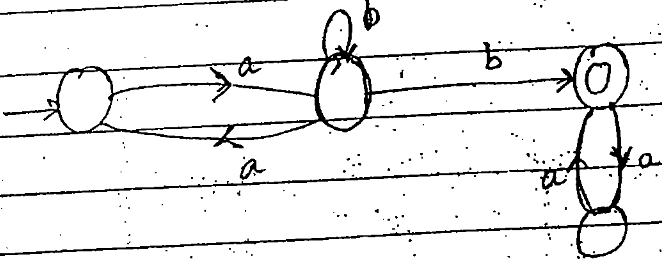
⑧

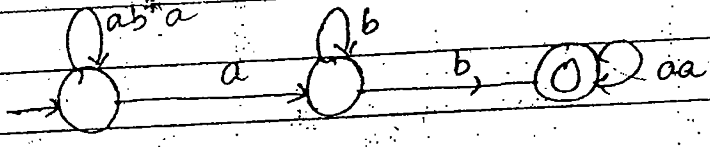




$r = [a(a+b)^* b]^*$

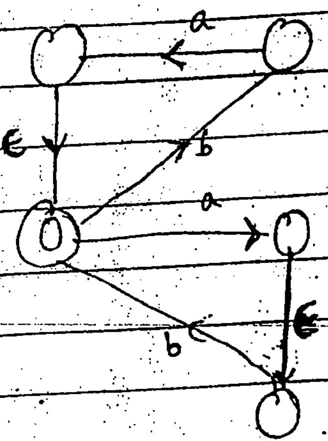
⑨

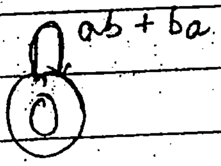




$(ab^* a) a b^* b (aa)^*$

(10)





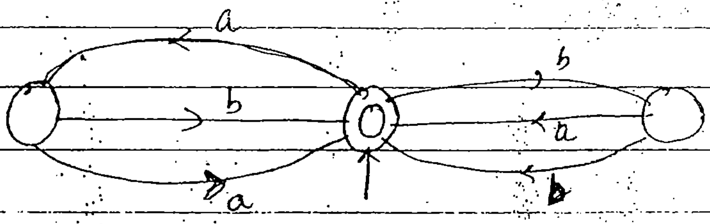
$r = (ab+ba)^*$

11)

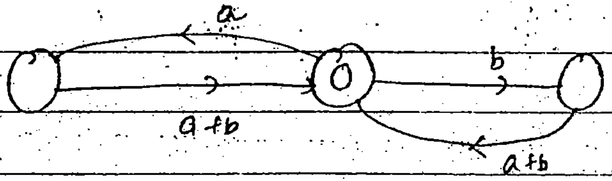
⑫

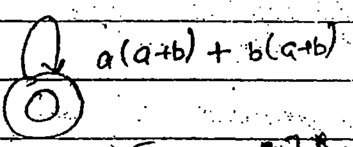
⑬

11)



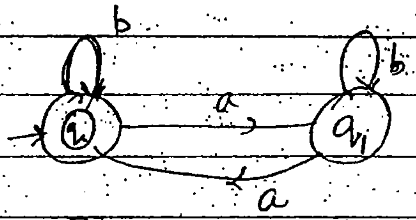
*

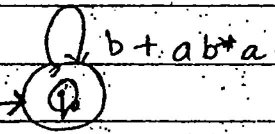




$[(a+b)^*]^*$

⑫

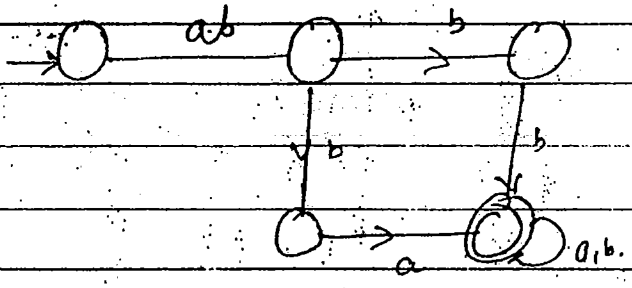




$= (b+ab^*a)^*$

⑬

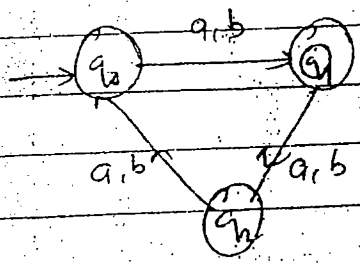
+ba

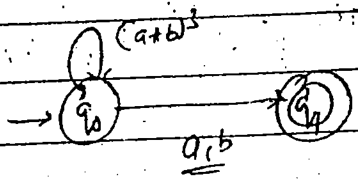


$ab+ba)^*$

$ab(ba+bb)(a+b)^*$

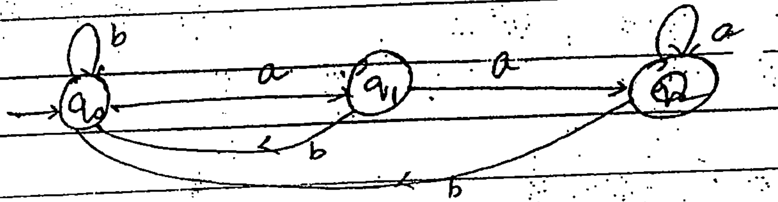
Q

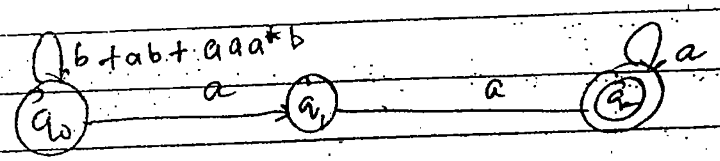




$$\left[(a+b)^3\right]^* (a+b)$$

Q :-





$$r = (b+ab+aaa^*b)\ aaa^*.$$

Regu

1. r

2. r

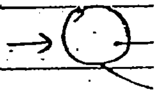
(1) Meth

a) Klee

b) Con

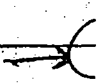
c) Un

a) r^*



b) Conc

c) Uni



Regular Expression to F.A

1. Method of Synthesis

2. Method of Decomposition.

(a+b)

(1) Method of Synthesis

$$\begin{tikzcd}
RE \arrow[r] & \epsilon NFA \arrow[d] \\
DFA \arrow[u] & NFA \arrow[l]
\end{tikzcd}$$

a) Kleen closure

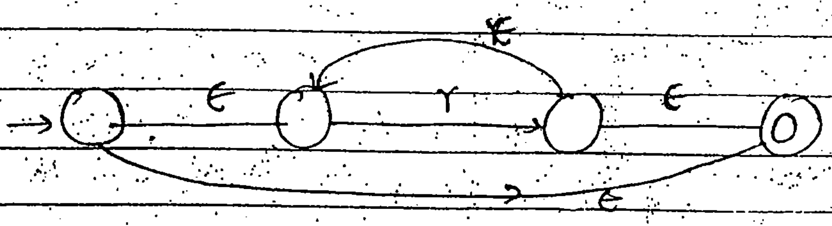
a

b) Concatination

)

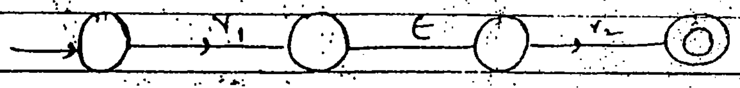
c) Union

a) $r^* = \epsilon, r, rr, rrr \ldots$

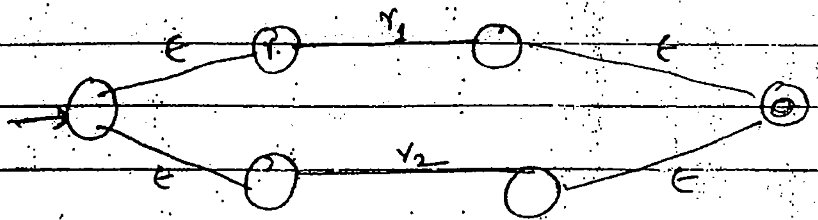


a*.

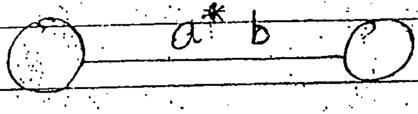
b) Concatenation $(r_1 r_2)$

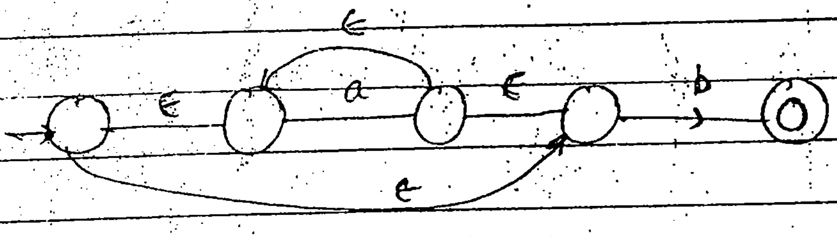


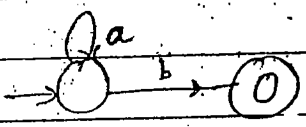
c) Union $(r_1 + r_2)$



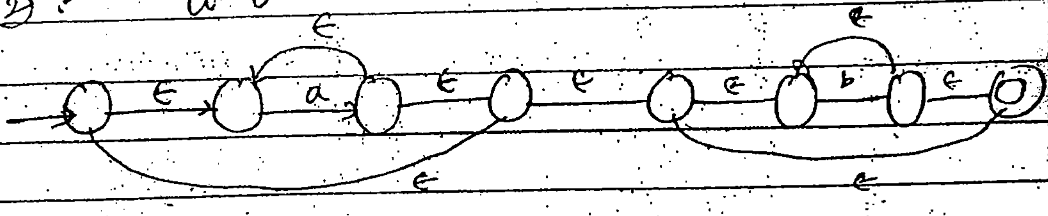
Example :-
$r = a^*b$



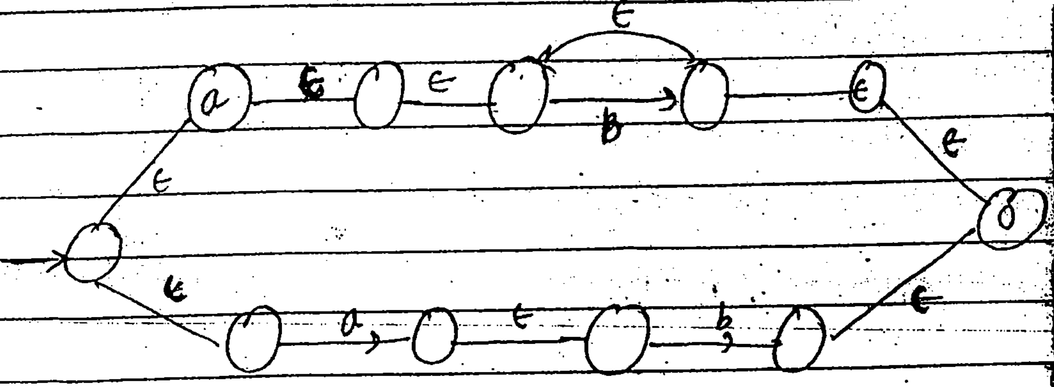




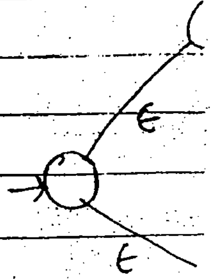
② :- $a^* b^*$



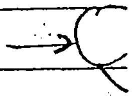
③ $ab^* + ab.$

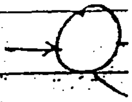


④ (

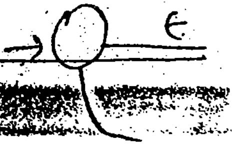


⑤ (ab

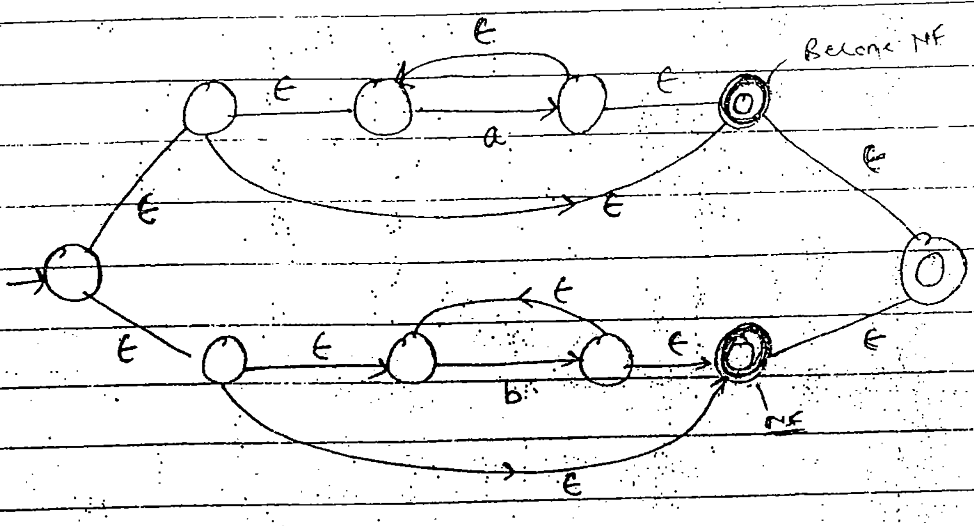




⑥ (a-

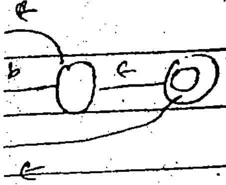


④ $a^* + b^*$

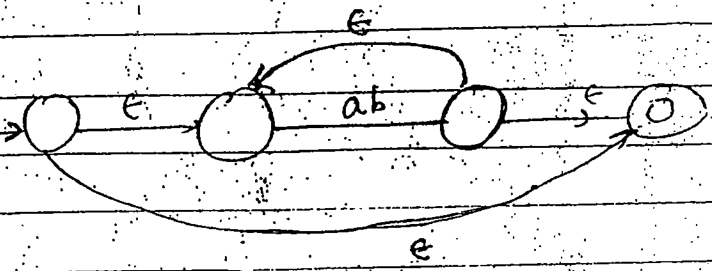


ε
ε
a
ε
Become NF
ε
ε
ε
ε
ε
ε
b
ε
NF
ε

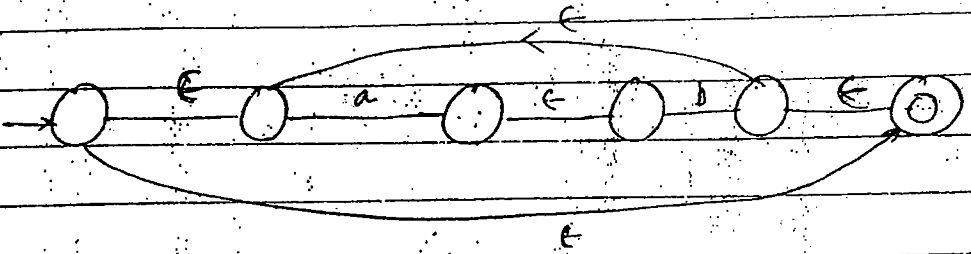
⑤ $(ab)^*$



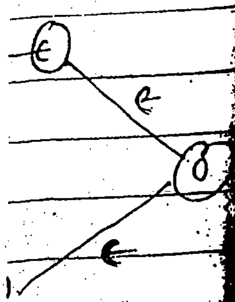
ε
b
ε
ε



ε
ε
ab
ε
ε

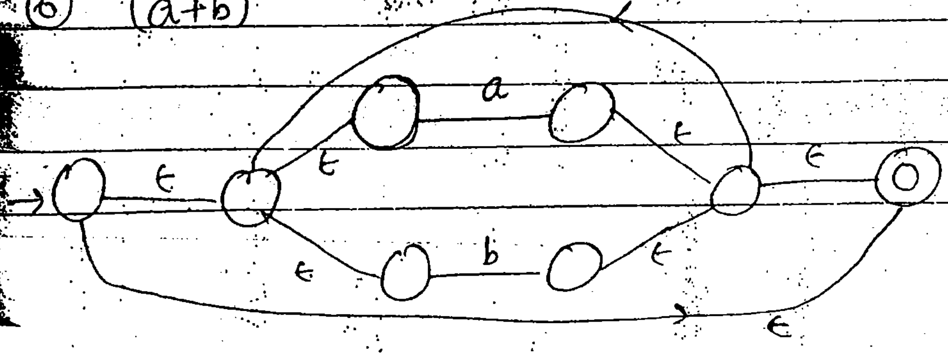


ε
ε
a
ε
b
ε
ε



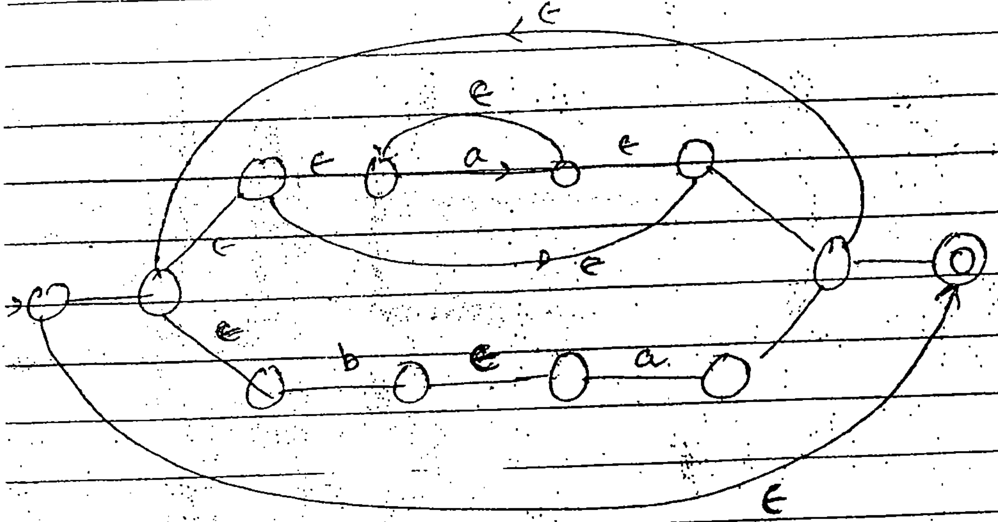
ε
ε
ε

⑥ $(a+b)^*$



a
ε
ε
ε
ε
b
ε
ε
ε

7) $(a^* + ba)^*$

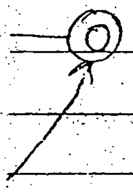


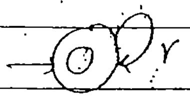
⑧ $((a^*b)^*b + ab^*)^*((ba)^* + a^*ba)^*$

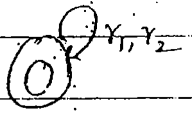
Method

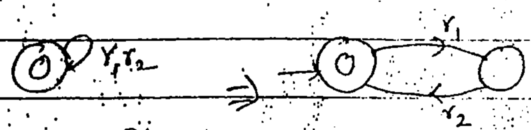
1. r^*
2. $(r_1$ -
3. $(r_1 r$
4. $r_1^* r_2$
5. $r_1 r_2^*$
6. $r_1^* r_2$ -
7. $r_1^* r_2$
8. r
9.
10.
11.
12.
13.
14.

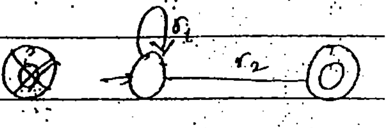
Method of Decomposition :-

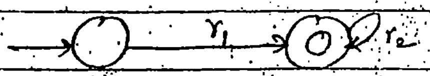


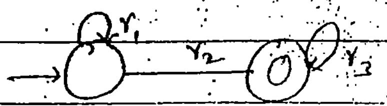
1. $r^* =$ 

2. $(r_1+r_2)^* =$ 

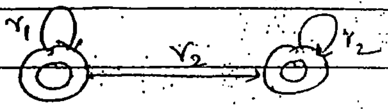
3. $(r_1 r_2)^* =$ 

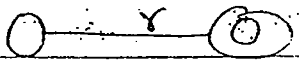
4. $r_1^* r_2 =$ 

5. $r_1 r_2^* =$ 

6. $r_1^* r_2 r_3^* =$ 

a)*

7. Imp $r_1^* r_2^* =$ 

8. $r =$ 

9.

10.

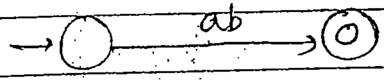
11.

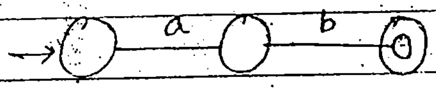
12.

13.

14.

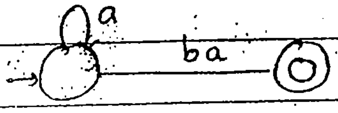
Q:- $r = \underbrace{ab}$ → Requires 3 states

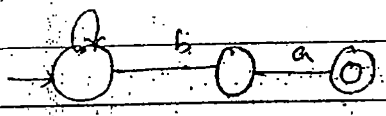




NFA = 3 state
DFA = 4 states.

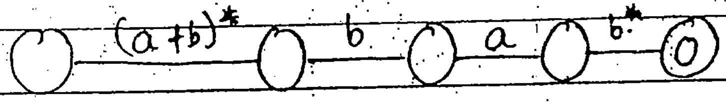
Q:- $\underset{\text{No state}}{a^*}\ \overset{\text{3 states}}{(ba)}$

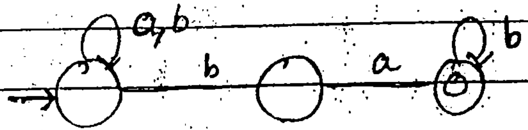




NFA = 3
DFA = 4.

Q:- $(\underset{\text{No state}}{a+b})^*\ \underbrace{ba}_{\text{3 states}}\ b^*$ No state

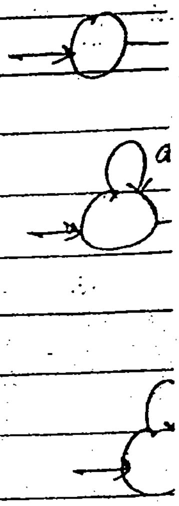




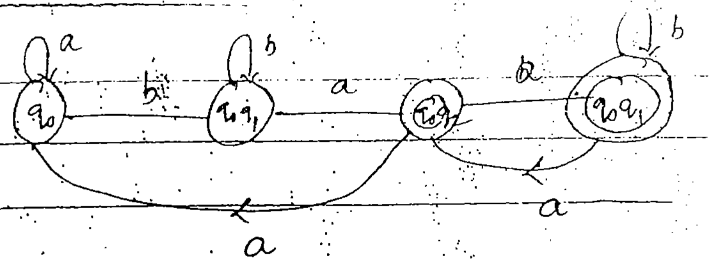
NFA
3 states.

DFA.

Q:- a

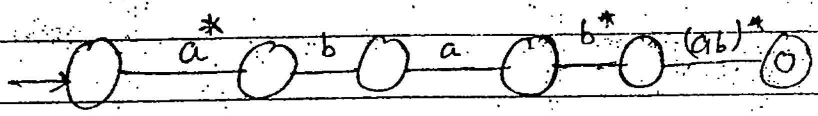


DFA.

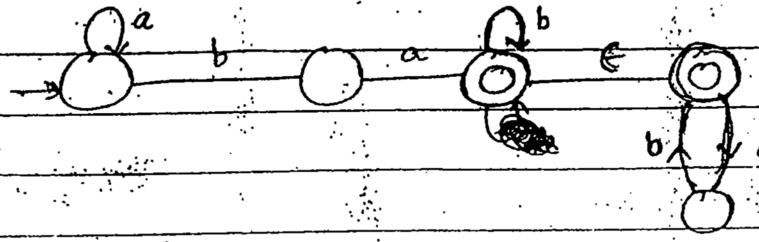


DFA = 4 states

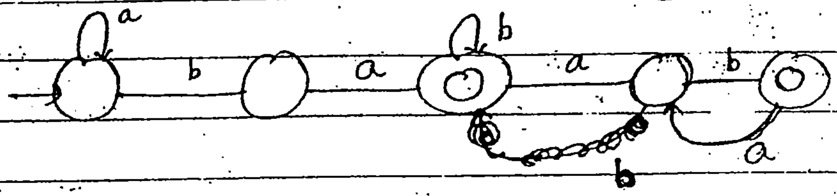
Q:- $a^* b a b^* (ab)^*$.



bab (E)
babab
b ab ab ab



7 = 3
1 = 4



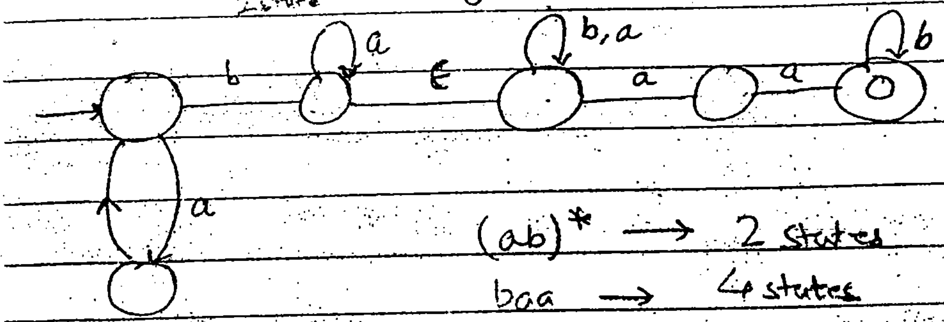
NFA = 5 states   DFA = 6 states.

A
3 states.

baa → 4 states

0 state   0 state ↑

Q:- $\underbrace{(ab)^*}_{2\ state} \; b \; \underset{0}{a^*} \; (b+a)^* \; aab^*$

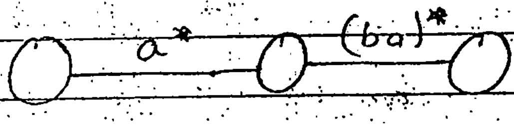


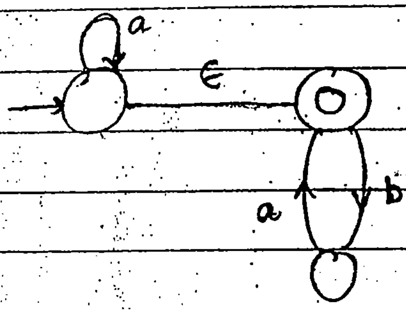
$(ab)^* \longrightarrow$ 2 states

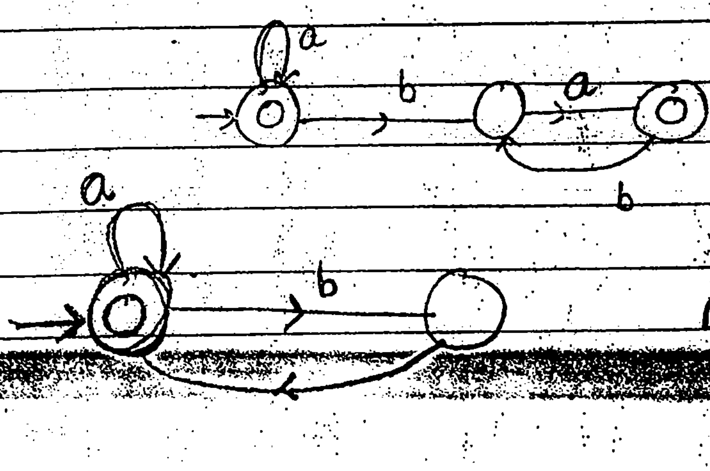
baa $\longrightarrow$ 4 states

Q:- Find No. of states in minimal FA accepting language : $a^*(ba)^*$

Sol:-







NFA = 3 state

DFA = 4 states

Q*:- No

Gate

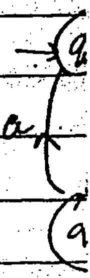
Sol:-

Q:- (a

Sol:-

NFA ⇒

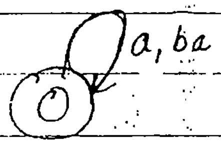
DFA

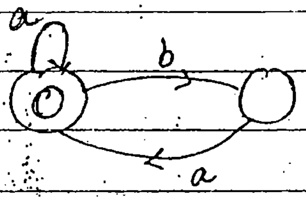


Q*:- No. of states in minimal FA accepting Language $(a+ba)^*$

Gate

Sol:-



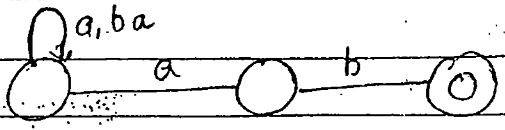


NFA = 2 states

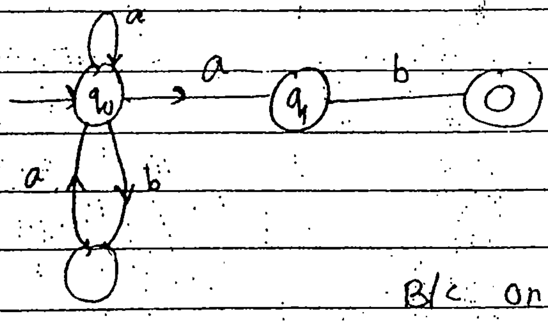
DFA = 3 states.

Q:- $(a+ba)^* ab$.

Sol:-



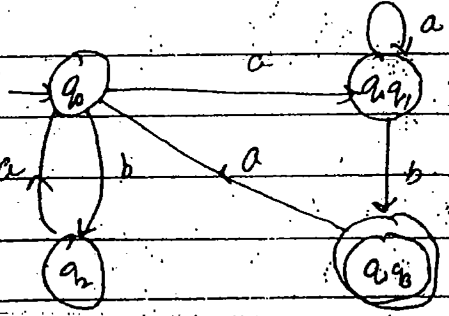
NFA ⇒



4 states

B/c on giving a on q_0 it goes to 2 states we will Cons DFA for it.

DFA



4+ 1
Dead state.

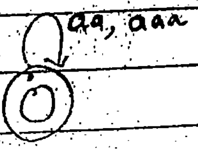
al FA

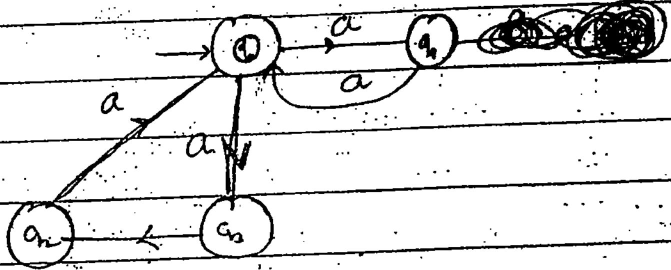
FA = 3 state

= 4 states

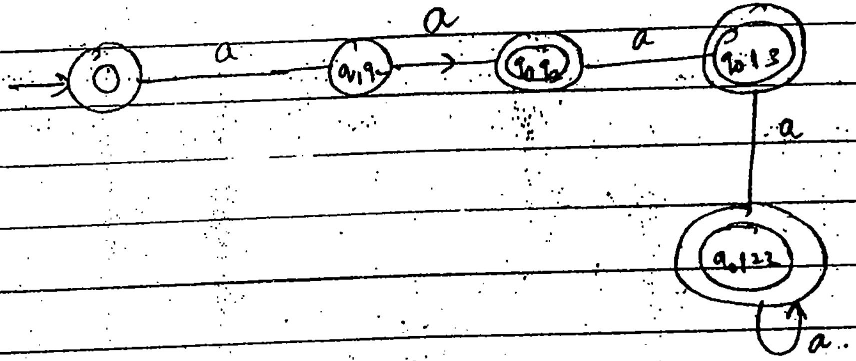
Q:- No. of states in Min FA accept the lang. generated by $(aa + aaa)^*$

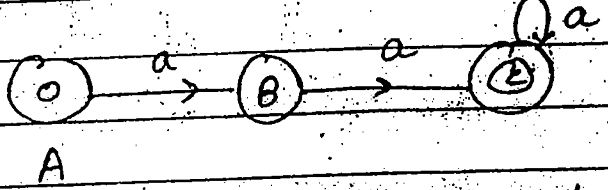
Sol:-





DFA:-

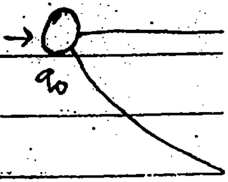




No. of states = 3

Reductio

Q:- (



Froms
q₀/q₀
q₃
q₅/q₀
q₆
q₁₀/q₀

NF

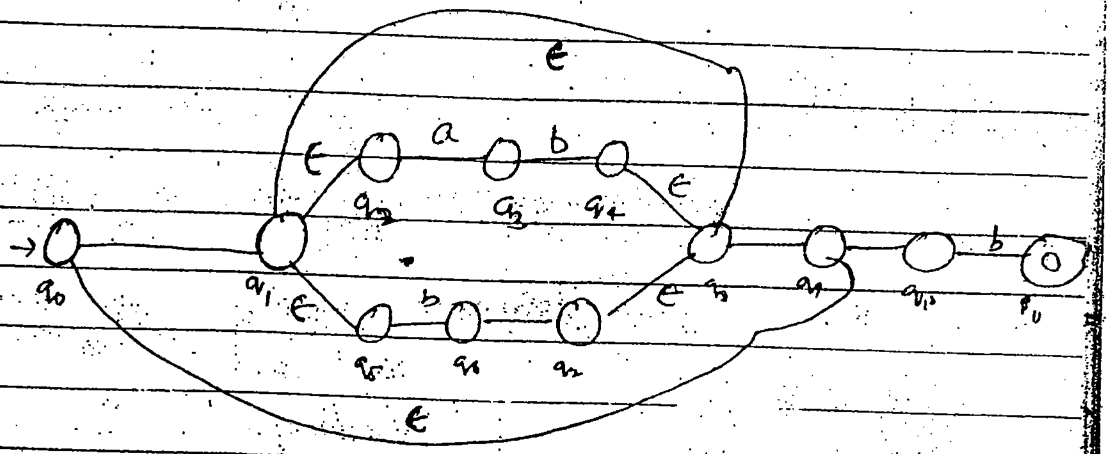
DFA →

cept the

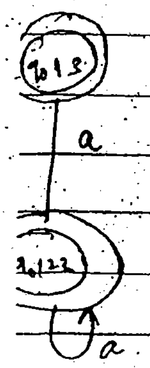
-aaa)*

Reduction of ε to moves.

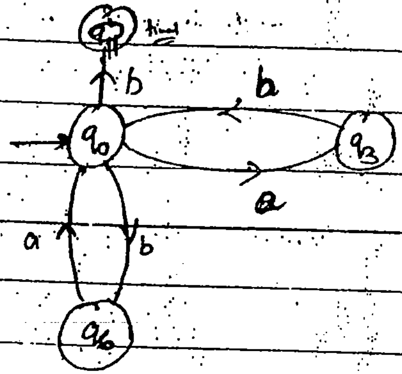
Q:- (ab+ba)* b.



Froms	To state	Symbol.
q2/q0	q3	a
q3	q4/q0	b
q5/q0	q6	b
q6	q7/q0	a
q10/q0	q11	b.

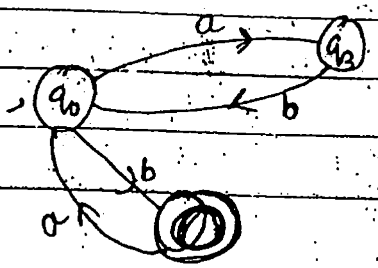


NFA =



3

DFA →



Algebric Properties of R.E.

1. Closure:- R.E. satisfy the closure property w.r.t $*, \cdot, +$. If r is RE then r^* RE. r_1, r_2 are RE then $r_1 + r_2$ & $r_1 \cdot r_2$ are also RE.

2. Associative :-

~~sta~~ Satisfy property w.r.t. to $+, \cdot$

$$(r_1 + r_2) + r_3 = r_1 + (r_2 + r_3)$$

$$(r_1 \cdot r_2)\, r_3 = r_1\, (r_2 \cdot r_2).$$

3. Identity Property :-

$$\left.\begin{array}{l} r + x = r \\ r \cdot x = r \end{array}\right\} \Rightarrow x \text{ is c/d Identity element.}$$

$$\left.\begin{array}{l} r + x = r \Leftrightarrow x = \emptyset \\ r \cdot x = r \Leftrightarrow x = \epsilon \end{array}\right\} \text{ is Identity element.}$$

4. Annihilator :-

$$\left.\begin{array}{l} r + X = X \\ r \cdot X = X \end{array}\right\} \Rightarrow X \text{ is called annihilator.}$$

$r + x = x \iff$ Doesn't exist.

$r \cdot x = x \iff x = \phi$ Annihilator wrt to Concatenation.

5. Commutative :-

R.E is Commutative wrt to Union But not w.r.t. to Concatenation.

$\Rightarrow \; r_1 + r_2 = r_1 + r_2$

$r_1 \cdot r_2 \neq r_2 \cdot r_1$

6. Distributive Property :-

R.E satisfy both distributive prop wrt Concatenation & then Union.

$\Rightarrow \; r_1 \cdot (r_2 + r_3) = r_1 \cdot r_2 + r_1 \cdot r_3$

$\Rightarrow \; \cancel{r_1} (r_1 + r_2) \cdot r_3 = r_1 \cdot r_3 + r_2 \cdot r_3$

RE are not distributive w.r.t. Union are then Concatenation.

$\Rightarrow \; r_1 + r_2 \cdot r_3 \neq (r_1 + r_3)(r_2 + r_3)$.

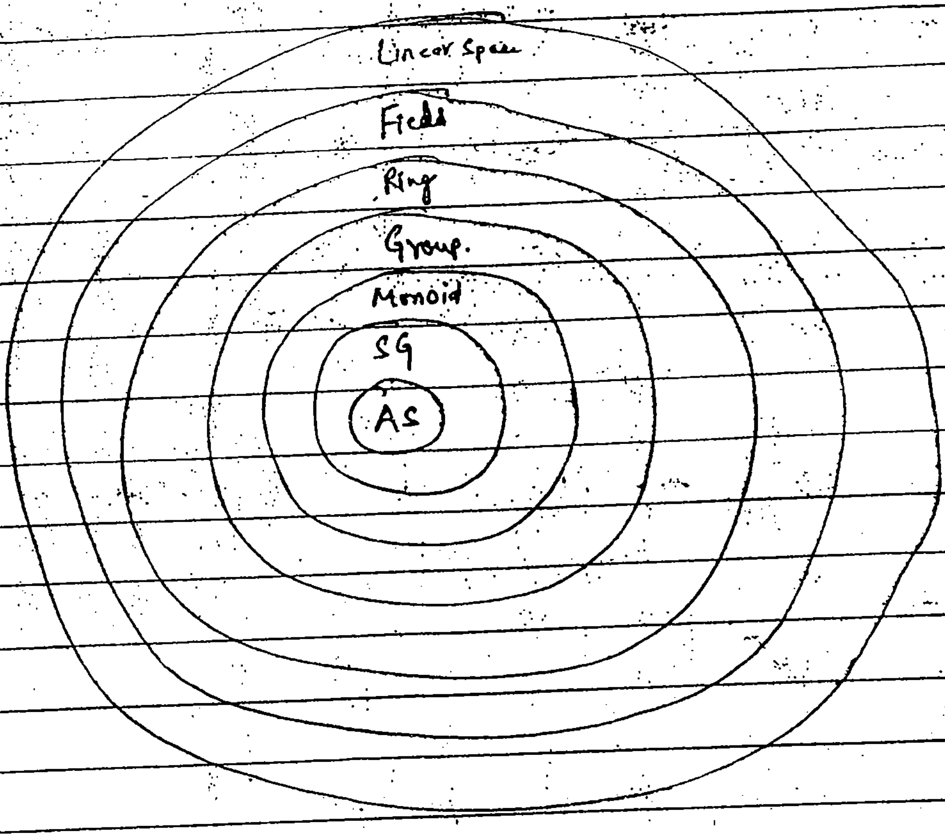
$r_1 r_2 + r_3 \neq (r_1 + r_3)(r_2 + r_3)$.

7) Idempotent Property:-

Regular expression satisfies idempotent prop. w.r.t to Union but not wrt to Concatenation.

$\Rightarrow \quad r + r = r$

$\Rightarrow \quad r . r \neq r$



1.
2.
3.
4.
5.
6) Grou
(Q)

1. Closure properties ⇒ Algebric

Drop to

2. Closure
Associative } ⇒ Semi Group

3. Closure
Associative } ⇒ Monoid
Identity

4. Closure
Ass
Identity } ⇒ Group
Inverse

5. Clo
Ass
Iden } = Abelian Group.
Inv.
Comm.

6) Group → Ring → Field → linear space
(O) (O, O₂) (O₁, O₂) (O₁, O₂)
Boolean

IPs ← Banach space ← Hilbert Space

7)

POSET → lattice → Boolean Algebra

$(\Sigma^*, +) \rightarrow$ closure, Ass., ϕ $\}$ Monoid

$(\Sigma^*, \cdot) \rightarrow$ C, A, ϵ $\}$ Monoid

$(\Sigma^*, +, \cdot) \rightarrow$ Closure, Ass., ϵ $\}$ Monoid

Clo

Regu
fol

→ C
R
P

Sym

Sub
Ma
Min
No

Reve
i

gebra

Closure Property of RL :-

Regular Lang. stisfy closure prop. w.r.t to following operations

- → Compliment L^c
- Reverse L^R
- Prefix $p(L)$
- Klein Closure L^*
- +ve Closure L^+
- Union $L_1 + L_2$
- Concatenation $L_1 \cdot L_2$
- Intersection $L_1 \cap L_2$
- Diff. operator $L_1 - L_2 \,/\, L_2 - L_1$ $\quad L_1 - L_2 = L_1 \cap L_2'$
  $L_2 - L_1 = L_2 \cap L_1'$

Symm. Diff. → $L_1 \Delta L_2$ $(L_1 - L_2) \cup (L_2 - L_1)$

$\frac{L_1}{L_2}$ Quotient operator.

Substitution, Homomorphism, Inverse Homo

Max (L) → i.e. max length gen by language

Min (L), Cycle (L), INIT(L), COR
$\downarrow$ compliment

NOR, $\frac{1}{2}L$, $\frac{1}{4}L$, $\frac{1}{16}L$

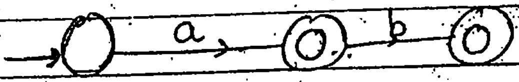
Reverse operator :- If L is Regular then L^R is also Regular language.

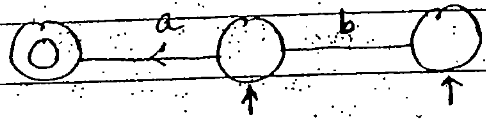
Procedure :- 1) Interchange initial & final state.

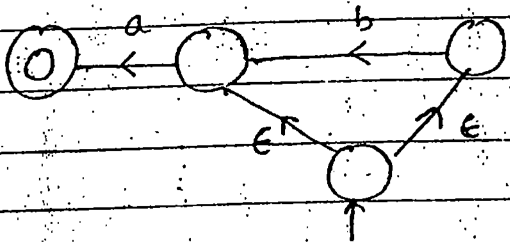
2) Change direction of edge.

3) If more than one initial state occur than simplify the sys. to only one single state.

L = {a, ab}.







Prefix Operator :- If L is Regular lang. then prefix of L is also regular Lang.

Procedure

1) To

2) To
de

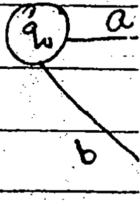
L = }

Prefix.

L=

Prefix.

L = {



Dead,→

al

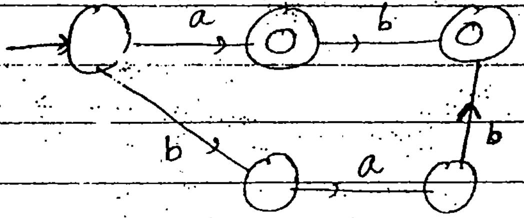
Procedure :-

1) In NFA make all ~~fp~~ state as final state.

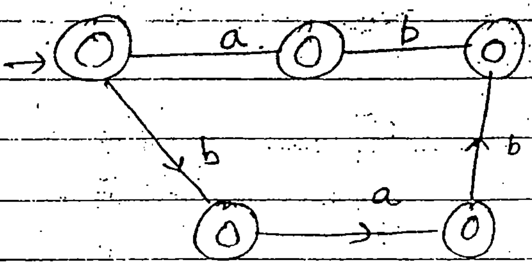
2) In DFA make all state as final except dead state.

L = {a, ab, bab}.
Prefix = {ε, a, ab, b, ba, bab}

L =

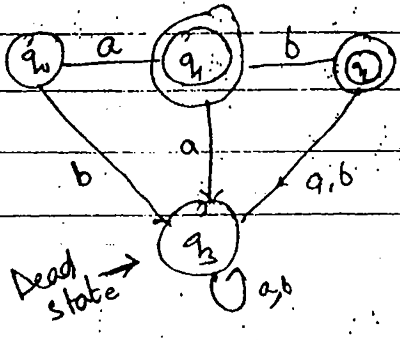


Prefix.

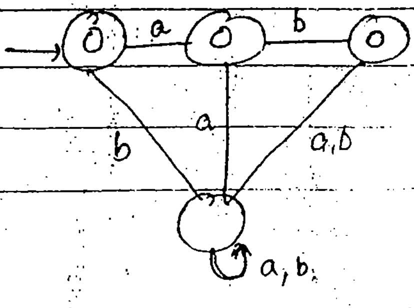


ular lang
is also

L = {a, ab}.   prefix = {εa, ab}.



DFA



Quotient Operator :-

If L_1, L_2 are 2 R.L than L_1/L_2 is also R.L.

where $\frac{L_1}{L_2} = \{ x / xy \in L_1, y \in L_2 \}$.

Quotient

$\frac{xy}{x} = y$ $\qquad$ $\frac{xy}{y} = x$

Left Quotient $\qquad$ Right Quotient

Ex:- $\frac{01}{1} = 0$ $\qquad$ $\frac{010}{010} = \epsilon$

$\frac{011}{01} = 1$ $\qquad$ $\frac{010}{100} = \emptyset$

$\frac{010}{10} = 0$ $\qquad$ $\frac{010}{1} = \emptyset$

$\frac{0101}{01} = 01$

Note :- $\frac{x}{\epsilon} = x$

$\frac{x}{x} = \epsilon$

$\frac{x}{y} = \emptyset \quad (x \neq y)$.

..L than

}
}.

ent

Ex :- $L_1 \{ 001, 101, 110 \}$ $L_2 = \{01, 10\}$.

$$\frac{L_1}{L_2} = \{ 0, 1 \}.$$

Ex :- $L_1 = \{ a^* \}$ $L_2 = \{a\}$.

$$\frac{\epsilon}{a} = \emptyset$$

$$\frac{L_1}{L_2} = \{ a^* \}.$$

$$\frac{a}{a} = \epsilon$$

③ $L_1 = a^* b^*$

$L_2 = b^* a^*$

$$\frac{a^* b^*}{a^*} = b^*$$

$$\frac{a^* b^*}{b^*} = a^*$$

$$\frac{L_1}{L_2} = \{ a^* b^* \}$$

$$\frac{L}{\epsilon} = L$$

a^*, b^* are part of $a^* b^*$ So Ans is.

4) $L_1 = a^* b a^*$ $L_2 = b^* a$

$$\frac{L_1}{L_2} = \{ a^* b a^* + a^* \}$$

When we $a^* = 0$ & then we get ba^* from which we can get ϵ, a, aa, aaa... $\Rightarrow a^*$

NOTE :-

If L is any language on Σ then

① $\frac{\Sigma^*}{L} = \Sigma^*$

② $\frac{L}{\Sigma^*} = \text{prefix}(L)$

③ $\left.\begin{array}{l}\frac{RL}{RL}\\ \\ \frac{RL}{NRL}\end{array}\right] \Rightarrow R.L$ $\left.\begin{array}{l}\frac{NRL}{RL}\\ \\ \frac{NRL}{NRL}\end{array}\right\} \Rightarrow RL/NRL$

Ex :- $L_1 = \{ a^n b^n / n \geq 1 \}$.
$L_2 = \{ b^n a^n / n \geq 1 \}$.

$\frac{L_1}{L_2} = \phi \rightarrow RL$ $\frac{NRL}{NRL} = RL$

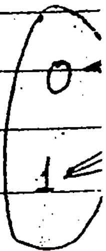
② $L_1 = \{ a^n b^n c^n / n \geq 1 \}$ $\frac{NRL}{RL} = NRL$
$L_2 = \{ c^n / n \geq 1 \}$

③ $L_1 = a^n b^n c^n \mid n \geq 1$ $\frac{L_1}{L_2} = \epsilon$ RL
$L_2 = a^n b^n c^n$

NOTE :

2^{Σ^*}

Σ = {

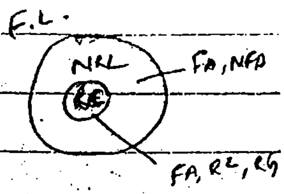


NOTE :- $\Sigma = \{a, b\}$.

$\Sigma^* = \{\epsilon, a, b, \quad\}$ = set of all strings.

$2^{\Sigma^*} = \{$ Set of All F.L. over $\Sigma \}$.

Regular Lang. Non Regular.

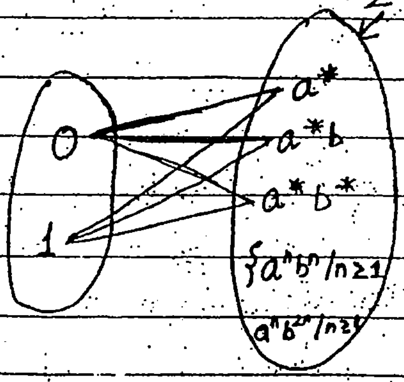


$\Sigma = \{0, 1\}$ $\Delta = \{a, b\}$

$2^{\Delta^*} = \{$ FL over $\Sigma \}$

RL NRL

RL/NRL



RL

= NRL

RL

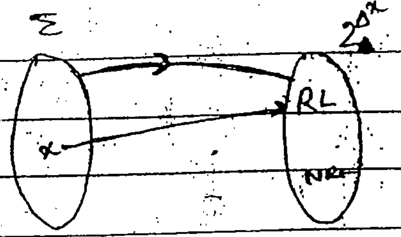
3/11/10.

One to ma ()

Substitution :-

Let Σ & Δ are 2 alphabets then substitution is method from Σ to Δ where symbols of Σ is replaced by regular language of another alphabet Δ.

$\therefore \; s : \Sigma \rightarrow 2^{\Delta^*} \ni s(x) = L \; \{ L \rightarrow \text{R.E. of } \Delta \}$



NOTE :-

1. If L is a regular Language & s is Substitution defined on L then s(L) is also Regular Lang.

2. $s(\emptyset) = \emptyset$
   $s(\epsilon) = \epsilon$
   $s(x^*) = [s(x)]^*$
   $s(x_1 x_2 x_3 \ldots x_n) = s(x_1)\, s(x_2)\, s(x_3) \ldots s(x_n)$

   $s(x_1 + x_2 + x_3 + x_4 \ldots x_n) = s(x_1) + s(x_2) + s(x_3) + \ldots s(x_n)$

example

a) s(

b) s(

c) s(

d) L=

example

a) L=
s(L)

b) L
s(

example

$\Sigma(0,1)$ $\quad \Delta = \{a,b\}$.

$s(0) = a^* b$

$s(1) = b a b^*$

a) $s(00) = a^* b\, a^* b$.

b) $s(10) = s(1) . s(0) = b a b^* a^* b$.

c) $s(\emptyset^*) = [\,s(0)]^* = [(a^* b)]^*$

d) $L = \{01, 11, 10\}. = s(L) = \{s(0)\, s(1), s(1)\, s(1), s(1)\, s(0)\}$

example $\Sigma = \{a,b\}$ $\Delta = \{0,1\}$. $s(a) = (01)^*$

$s(b) = 0^* 1^*$

a) $L = a^* b^*$

$s(L) = [s(a)]^* [s(b)]^* = (01)^* (0^* 1^*)^*$

b) $L = a^* + b a^*$

$s(L) = [s(a)]^* + s(b)\,[s(a)]^*$

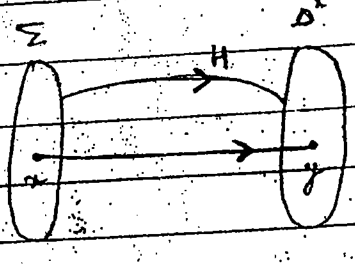
$= [(01)^*]^* + 0^* 1^* [(01)^*]^*$

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

One to one -

Homörphism :- It is a kind of Substitution fm alphabet Σ to Δ which symbols of Σ is replaced by single string of another alphabet Δ.

$$\therefore\ H\ \cancel{\rightarrow} : \Sigma \rightarrow \Delta^{*} \ni H(x) = y \quad x \in \Sigma, \; y \, \Sigma \, \Delta^{*}$$



example :-

Ex : Σ = (0,1)   Δ = {a,b}.

$$\left.\begin{aligned} H(0) &= ab \\ H(1) &= baa \end{aligned}\right\} \text{Homomorphism}$$

$$\left.\begin{aligned} H(0) &= a+b \\ H(1) &= baa \end{aligned}\right\} \text{Substitution.}$$

NOTE

1. If

wh

Example

abb

bab

bb

Q:-

to.

&

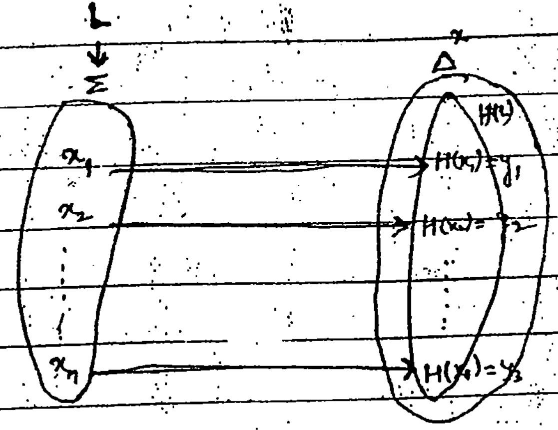
other

$\in \Sigma$

$\Sigma \; \Delta^x$

NOTE

1. If L is Regular Language & H is Homomorphis then H(L) is also regular language. Where H(L) is cld Homomorphic Image of L.

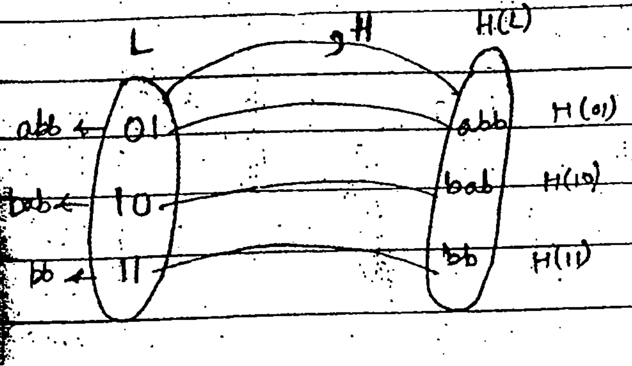


Example :- $\Sigma = (0,1)$; $\Delta = \{a,b\}$.

$L = \{01, 10, 11\}$.

$H(0) = (ab)$

$H(1) = b$



$H(L) = \{abb, bab, bb\}$.

H(L) is Homomorphic Image of L.

Q:- $\Sigma = \{a,b\}$, $\Delta = \{0,1\}$.

$H(a) = 001$

$H(b) = 101$

i) $L = \{ a^* b^* \}$

$H(L) = (001)^* 101$

ii) $L = a b^* a^*$

$= (001)(101)^* . (001)^*$

iii) $L = (a^* + b)^* = ((001)^* + 101)^*$

Inverse Homomorphism :- Let H is homomorphism from Σ to Δ. Then inverse homomorphism of x is denoted by $H^{-1}(x)$ & defined as

$$H^{-1} = \{ x \mid H(x) \in H(L) \}.$$

$$\therefore H^{-1} = \{ x \mid y \; \exists \; y \text{ in } H(L) \text{ s.T. } H(x) = y \}.$$

$$\therefore x \in H^{-1} \Leftrightarrow H(x) \in H(L)$$

$$\Leftrightarrow \exists \, y \in H(L) \ni H(x) = y.$$

* If

If L

$H^{-1}($

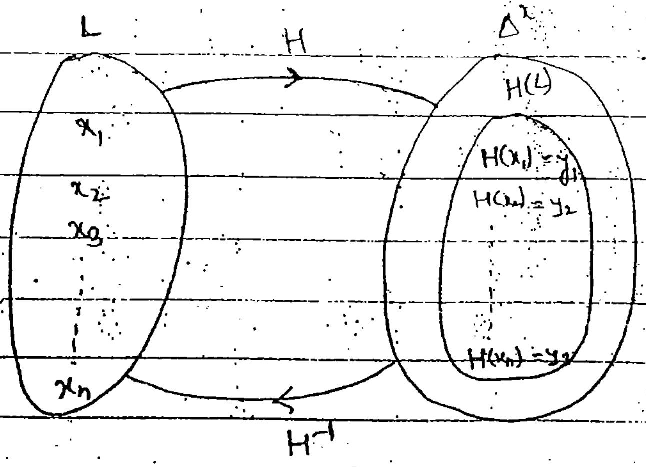
Que :- Σ

$L = \{$

$H(L)$

$H(a$

$H^{-1}($



* If L is a RL ⇒ then H(L) is also RL.

If L is a RL & H(L) is homomorphism then

$H^{-1}(H(L)) = L$ is also a RL.

Que :- $\Sigma = \{a, b\}$ $\Delta = \{0, 1\}$ $H(a) = 0$

$H(b) = 10$

$L = \{0\,0\,\underline{1}0\,\underline{1}0\}$

$H(L) = \{aabb\}$.

$H(aabb) = 001010$

$H^{-1}(001010) = aabb$

② $\Sigma = \{0,1,2\}$ $\Delta = \{a,b\}$ $H(0) = a$

$H(1) = ab$

$L = \{\underline{abaab}\}$ $H(2) = ba$

$H^{-1}(L) = \{101, 021\} = H^{-1}(a) + H^{-1}(ba) + H^{-1}(ab)$

b) $L = \{\underline{bababaa, ababaa}\}$

$H^{-1}(L) = \{\underline{2220, 0220, 1100, 1020}\}$.

c) $L = (ab)^*$

$H^{-1}(L) = 1^*$.

d) $L = \underline{a^*ba}$.

$H^{-1}(L) = 0^*2 + 0^+10 \rightarrow a^+aba$.

Regu

P

Jc

If L

Lan.

Su

Where

NOTE:

① I

d

② F

③ E

④ T

Regular lang par Apply nhi Karte hai.

a
ab
ba

PUMPING LEMMA FOR Regular Language :-

Jab Hume Anspta Ho taki risk lo.

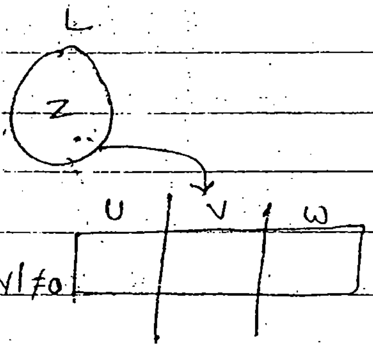
Input is NRL & O/P is also NRL.

a)+H'(as)

If L is any regular Language & $Z \in L$

Such that $|z| \geq n$ then

$uv^iw \in L \ \forall \ i \geq 1$



Where $Z = uvw$, $|uv| \leq z$ & $|v| \neq 0$

$uv^iw \in L \ v^i \geq 1$

↓

string of length 1, 2, 3, 4 ... n

NOTE :-

① It is used to prove that some of the languages are not regular.

ba.

② For pumping lemma I/P is Non Regular & O/P is also non Regular.

③ Every regular lang. satisfy pumping lemma property.

④ The lang. which doesn't satisfy P. Lemma property is Non Regular Language.

Process :-

1) Assume that L is regular language.
⇒ L satisfy Pumping lemma property.

② let $z \in L$ ⇒ $|z| \geq n$ → No. of states in
Minimal FA.

③ split z into 3 parts

z

u v w

$|uv| \leq |z|$

$|v| \neq 0$

④ There exist atleast one value of i such that $uv^iw \notin L$

⇒ L is not satisfying pumming lemma property This is Contradiction

⑤ So L is not Regular Language.

example :- $L = \{a^n b^n / n \geq 1\}$

let $z \in L$

$z = a^n b^n$

a^{n-1} a b^n

$uv^iw = a^{n-1} a^i b^n$

for $i=2$ $\quad a^{n-1} a^2 b^n = a^{n+1} b^n \notin L$

nguage.

property.

$\Rightarrow$ L is Non Regular.

tee in

Example :- $L = \{ a^m b^n / m<n \}$

Let $z \in L$

$\Rightarrow z = a^k b^{k+1}$

$a^{k-1} \quad a^i \quad b^{k+1}$

f i such

$\Rightarrow uv^iw = a^{k-1} a^i b^{k+1}$

lemma

for $i=2$ $\quad a^{k+1} b^{k+1} \notin$ B/c $n>m$ for all values.

So L is not Regular.

③ $L = \{ a^{n^2} / n>1 \}$.

Sol:- $z = a^{n^2}$

$a^{n-1} \quad a \quad \epsilon$

$\Rightarrow uv^iw = a^{n^2-1} a^i$ for $n=2$

$= a^{n^2+1} \notin L$

$\therefore$ L is Not regular Language.

Weak form of Pumping lemma :-

$L = \{ a^{n^2} / n \geq 1 \}$ 1,4,9,16,25 ....

They are not in AP so NRL

$L = \{ a^{2^n} / n \geq 1 \}$ 1,2,4,8,16,32 — NRL.

If L is any language defined over $\Sigma = \{a\}$. i.e Only symbol is there. Such that length of the string of language L are in AP.

Then language L is regular

otherwise

L is Non Regular.

example :-

① $L = \{ a^{2n} / n \geq 0 \}$ → Regular

0,2,4,6,8 → AP.

② $L = \{ a^{3n+2} / n \geq 0 \}$ — Regular.

2,5,8,11

③ $L = \{ a^{n^3} / n > 0 \}$.

$0^3, 1^3, 2^3, 3^3$ Not in AP

Not Regular

⇒

④ L =

1,

⑤ L =

⑥ L =

⑦ L =

⑧ L =

⑨ L = {

(10) L =

#

ϵ a

It

' so NRL

L.

$\Sigma = \{a\}$.

with

Language

egular

$\Rightarrow$

④ $L = \{ a^{n^2+1} / n \geq 0 \}$. → Non Regular

1, 2, 5, 10

⑤ $L = \{ a^{2^n} / n \geq 0 \}$ → NRL

⑥ $L = \{ a^{m} / n \geq 0 \}$ NRL.

⑧ $L = \{ a^{(n!)^2} / n \geq 0 \}$ NRL

⑧ $L = \{ a^n /$ n is prime$\}$ NRL

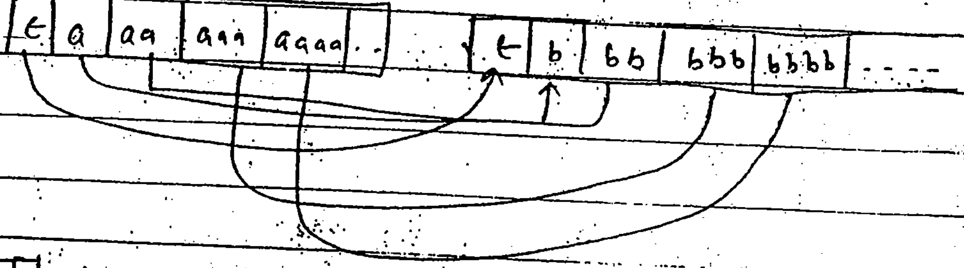
⑨ $L = \{ a^n /$ n is non prime$\}$ NRL

(10) $L = \{ a^{n^2} / n \geq 0 \}$ NRL

\# $a^n / n \geq 0$ $\quad$ $b^n / n \geq 0$

↓ RL $\quad$ ↓ RL

$\{ a^n b^n / n \geq 0 \}$.



It is only Special Arrangement of variable.

Myhill Nerode Theorem:- L is regular if and only if ~~if~~ No. of equivalance classes wrt L is finite.

① In Minimal F.A. we can define some language at each & every state. These language are called equivalance classes & they are Mutually exclusive.

$$Eq_1 \cup EQ_2 \cup EQ_2 \dots \cup E_n = \Sigma^*$$

② Union of all these equivalance classes is equal to Σ^*.

③ Union of some of Equi. Classes = L } Accept by FA

$$EQ_1 \cup EQ_2 \cup EQ_3 = L$$

④* No. of equivalance classes = No. of states in Minimal FA. accepting L. is finite.

∴ the Language L is Regular ⟺ No. of eq. clas w.r.t. L is finite.

⑤ L is Non regular ⟺ No. of eq. classes w.r.t. L is Non finite. i.e. Infinite.

⑥ If

Ex:-

→

(

Q:-

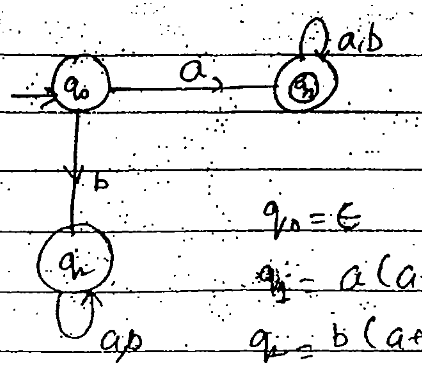
Sol:-

ar
uivalance
some
state.
uivalance
y
Σ^*
classes
$= L \{$ Accepted by FA
of
7.
L.
of eq. clas
q. Classes
ite.

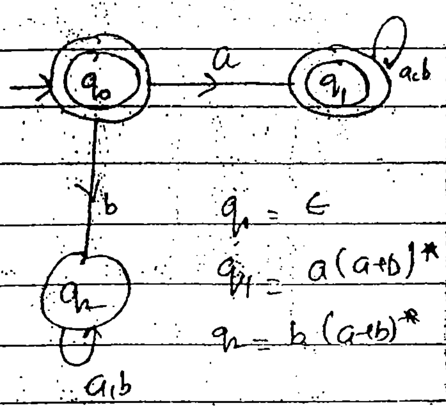
⑥ If L is regular Lang. that is left as well as Right invariant, i.e.
for every $x, y \in L$ $\exists z \in L$

S.T. $xz = yz$
$zx = zy$ }.

Ex:- $\Sigma = (a, b)$
$\Sigma^* = (\epsilon, a, b, \ldots)$
$L = a\cancel{x}$ $\qquad$ $L = \{\epsilon\} \cup \{ax\}$



$q_0 = \epsilon$
$q_1 = a(a+b)^*$
$q_2 = b(a+b)^*$

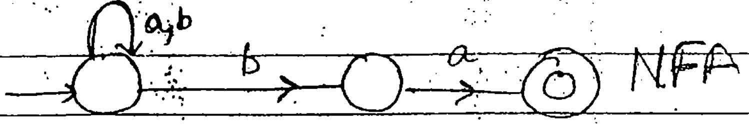


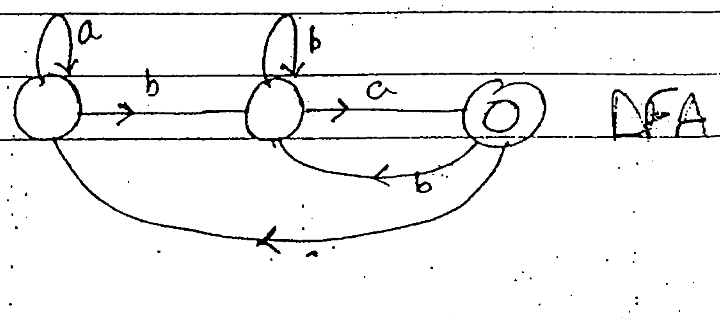
$q_0 = \epsilon$
$q_1 = a(a+b)^*$
$q_2 = b(a+b)^*$

$q_0 \cup q_1 = L$

Q:- find No. of eq. Classes w.r.t to ~~all~~ lang. of all String of a × b where every String end with ba.

Sol:-





Q:- Find No. of equivalance Classes.

Language	No. of Eq. Classes.
1. Starts & end with same Symbol.	5
2. Start & end with diff. Symbol.	5
3. n^{th} Sym. frm left is fixed.	$n+2$
4. n^{th} Symbol from Right is fixed.	2^n
5. $\|W\| = n$	$n+2$
$\|W\| \leq n$	$n+2$
6. $\|W\| \geq n$	$n+1$
7. $\|W\| = r \bmod n$	n
8. $\|W\|_a = r \bmod n$	n
$\|W\|_b = r \bmod n$	n
Binary No = r mod n	

Lang

1. {c

2. {a

3. {

4. $\{a^m$

5.

6.

7.

8. a^{2n}

9. a^{n^2}

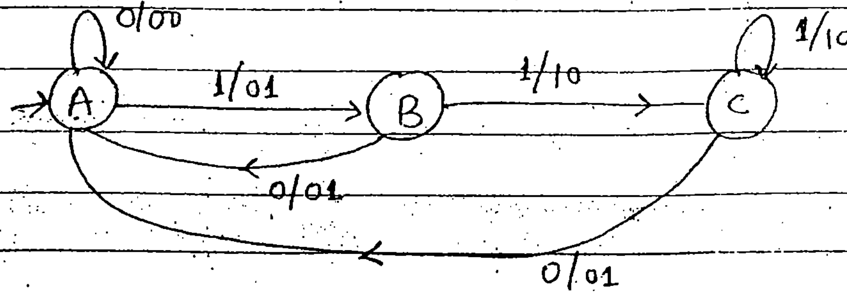
10. $a^{n!}$

a^p,

a^p/

Hog
X

Q:- The finite state m/c described by following state diagram with A as initial state an arc lable x/y x stand for 1 bit I/P & y stands for 2 bit O/P.



hortest
e.

~~a)~~ Outputs the sum of the present previous bit of I/P.

b) O/P 01 whenever I/P sequence contains 11.

c) O/P 00 whenever I/P sequence contains 10.

d) None of the above.

1 → [illegible]
1 → 10
10 ⇒ 10

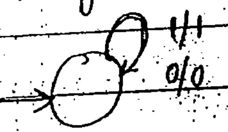
11 → 10   10 → 01
1+1 → 10   1+0 = 01

10
1

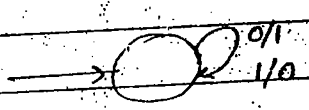
Q:- $(Me)^2$ means that given mealy m/c & ip string is processed it & then O/P string is immediately fed into the M/c as ~~I~~/P & reprocessed only the 2nd resultant O/P is considered as final O/P of $(Me)^2$. If final O/P string is same as original I/P string then we say that $(Me)^2$ as an identity property.

Consider following M/c's.

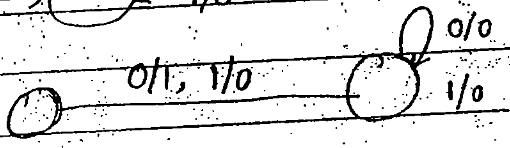
a)



b)



c)



which of m/c have identity property.

a) I Only c) a & c but not b.
b) a & b but not c. d) All.

$\Rightarrow$ 101 → 001
↓
101

00 → 11 → 00.

Q:- $(Me_1)(Me_2)$ mean that I/P string is processed on Me_1 & then O/P string is immediately fed to Me_2 as an I/P & reprocessed. Only the 2nd resultant O/P is consider the final O/P of Me_1 & Me_2. If O/P string is same as original I/P string then we

Language	No of Classese
1. $\{a^n b^n / m,n \geq 0\}$	3
2. $\{a^m b^n / m \geq 2, n \geq 0\}$	5
3. $\{a^m b^n / m \geq 0, n \geq 2\}$.	5
4. $\{a^m b^n / m = n\}$.	Infinite
5. $m > n$	〃
6. $m < n$	〃
7. $m \neq n$	〃
8. a^{2n} $n \geq 0$	2
9. a^{n^2} $n \geq 0$	Infinite
0. $a^{n!}$	〃
a^p / p is prime No.	〃
a^p / p is Not prime No.	〃

F.A with O/P (

Moore ← special case of DFA → Mealay
M/c M/c

They are O/P Generators.

① Both M/c are special case of DFA.

② Both the M/c are O/P generator rather than language acceptors so no need to define any final states. i.e $F = \emptyset$.

③ No concept of final & Dead state in Both missions

O/P depend state only

Moore M/c :- FA where O/P is associated with state itself is c/d Moore M/c.

OR

It is a 6 tuple, $M = \{Q, \Sigma, \Delta, \delta, \lambda, q_0\}$

where Q = set of all states
Σ = I/P alphabet
Δ = O/P alphabet

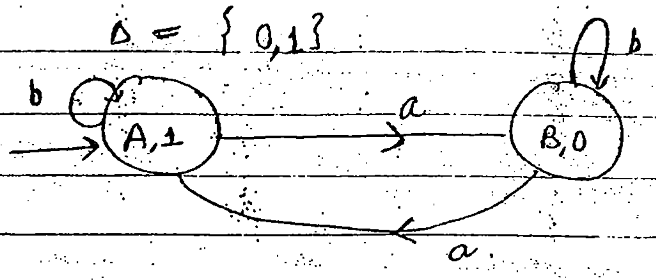
δ = transition f^n

$\lambda = Q \rightarrow \Delta$ is a O/P f^n.

q_0 = initial state.

Ex:- $\Sigma = \{a, b\}$

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

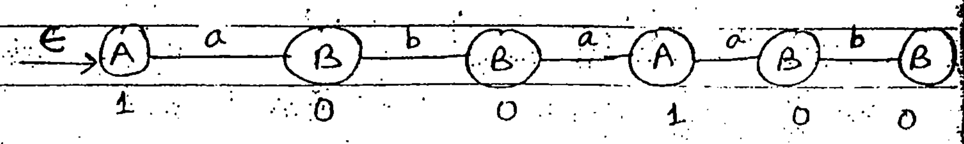


δ	a	b	λ
→ A	B	A	1
B	A	B	0

$\lambda(A) = 1$

$\lambda(B) = 0$

I/P abaab



$\underset{5}{\lambda(abaab)} = \underset{6}{100100}$ → No. of states

$\lambda(\epsilon) = \lambda$ (initial state)

$= \lambda(A)$

$= 1$

NOTE :-

1) O/P depends on state itself.

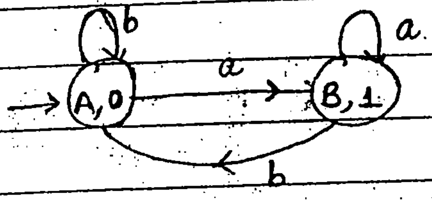
2) Length of I/P is m then length of O/P is m+1.

3) Moore m/c will respond even for ε.

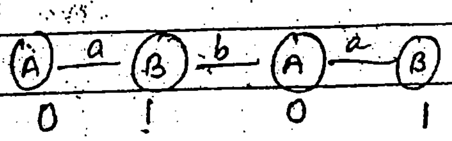
4)* $\boxed{\lambda(\epsilon) = \lambda(\text{Initial state}).}$

Q:- Cons Moore m/c which takes all string of a'b as input & ~~find~~ count No. a's in I/P string.

Sol:- Σ(a,b) , Δ = {0,1}. a = 1 accept
b = 0 Reject.



A-gar a dia to O/P 1 dega whi to 0.

aba = 

λ(aba) = 0101 = 2

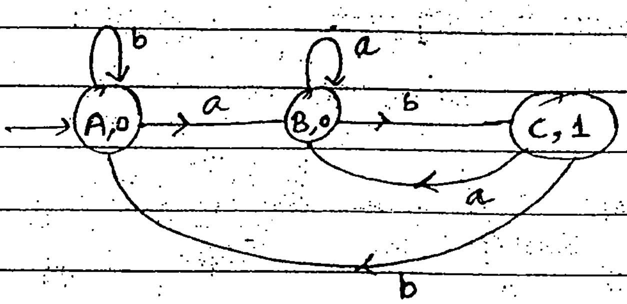
Q:- Cons. Moore M/c Contains all string of a,b as I/P & Count No. of occurance of substring a,b.

of O/P

Sol:- $\Sigma = (a,b)$ ab = 1 $\lambda(\underline{ab}a\underline{ab}b\underline{ab}) = 111$

$\Delta = (0,1)$ xx = 0

ε.



$\lambda(b\underline{ab}a\underline{ab}) = 0001001$

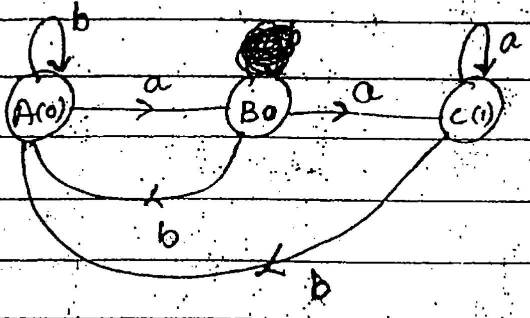
bing of I/P

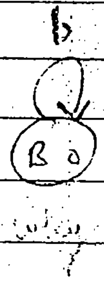
Q:- Cons Moore M/c with string of a's & b's as I/P & Count No. of occurance of 2 Consecutive a's.

accept

Reject.

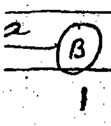
Sol:-





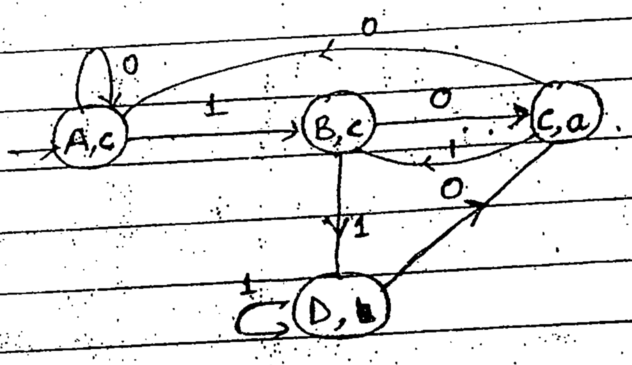
x dia

1 degr.



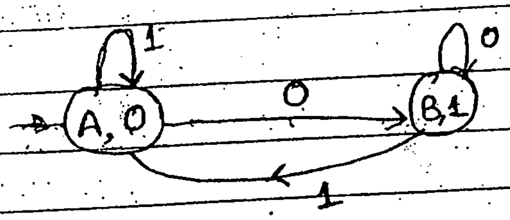
Q:- All string of 0 & 1 & O/P is a when I/P end with 10, 11 → b other wise O/P is C.

Solⁿ :-



Q :- To find 1's Comp of Binary No.

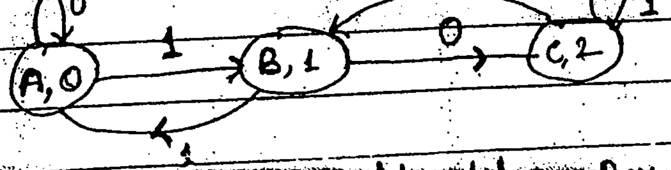
I/P
Sol :- 0 → 1
1 → 0



Ignore first O/P.

Q :- G.M.M/c takes all binary No as I/P & prints Reside Module of 3 as O/P {0,1,2}.

Sol :-



Module wale Case Me Modline Par jo

Meal
wit

6 Tup

Q = S
Σ =
Δ =
δ =
λ =
q0 =

Example

→ f
E

I/P

~~O/P associated with transition~~

when

no;

Mealy M/C :- F.A where O/P is ~~[illegible]~~ associated with Transition is called Mealy M/C.

OR

6 Tuple M/C $M = \{Q, \Sigma, \Delta, \delta, \lambda, q_0\}$.

).

Q = set of all state

Σ = I/P alphabet

Δ = O/P "

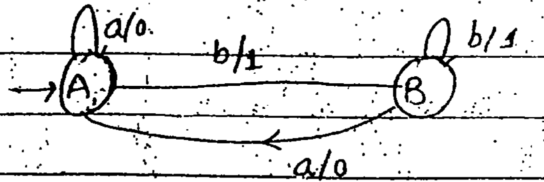
δ = $2 \times \Sigma \rightarrow Q$ Transition f^n

λ = $Q \times \Sigma \rightarrow \Delta$ is o/p f^n.

q_0 = initial state.

0.

Example :- $\Sigma = \{a, b\}$ $\Delta = \{0, 1\}$.

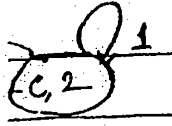


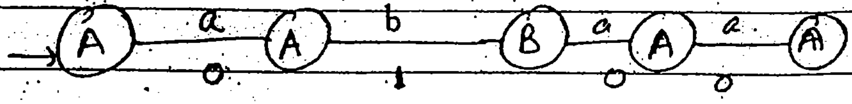
	I/P	O/P	I/P	O/P
δ	a	λ	b	λ
→ A	A	0	B	1
B	A	0	B	1

$\delta(A, a) = A$

No al I/P of 3 as

I/P → abaa.





$\lambda(abaa) = 0100$

ne pay jo

Rai Lo

NOTE :-

1) O/P depends on state & I/P symbol.

2) Length of I/P = Length of O/P.

3) Mealy M/c will not respond for empty string ε.

$\lambda(\epsilon) = \epsilon$

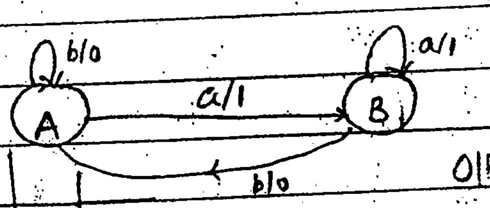
Example :- C. Me. M/c that takes all I/P of a'b' as I/P & count no. of occurance of a.

Sol :- $\Sigma = \{a, b\}$ Let A = 0 = b

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

b = 0

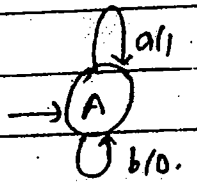
A = B



O/P 0 hoga to A par jayega

O/P 1 hoga to B par.

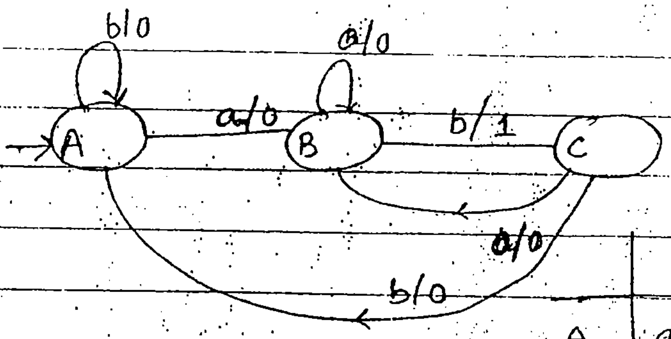
		λ		
A	a	1	b	0
B	a	1	b	0

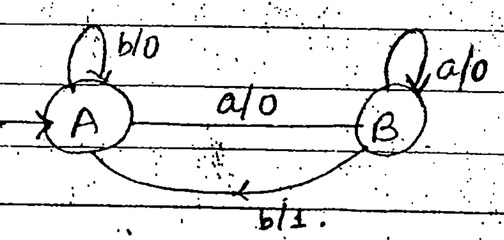


Isme HUM state + O/P par depend Karte hai ki next state kya

Q:- All string of a,b & Count No. of occurance of string ab.

Sol:-





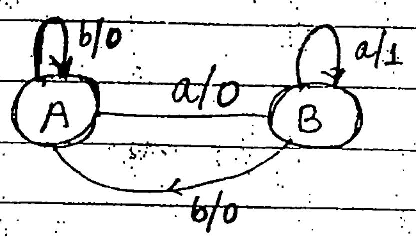
A	a	0	0
B	a	0	1
C	a	0	0

$\boxed{A = C}$

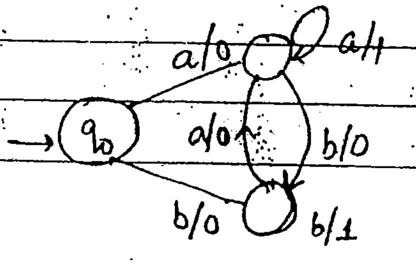
Q:- Cons. Me-M/c takes all string of a,b's as I/P. & Count No. of occurance of 2 Consecutive aa's.

* Construct min DFA then form the table & check $\underbrace{a\,a\,a\,a}$ that 1 2 3. 2 state are equal or Not.

Sol:-



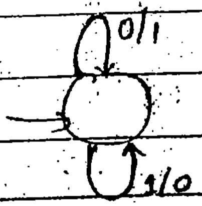
Q:- Cons. Me. M/c for $L = (a+b)^* (aa+bb)$
0,1,2,3.



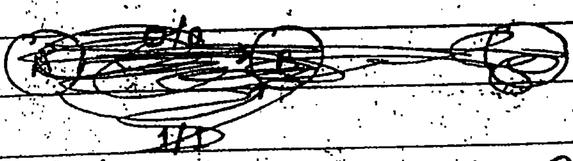
Q:- To find Complement of Binary No.

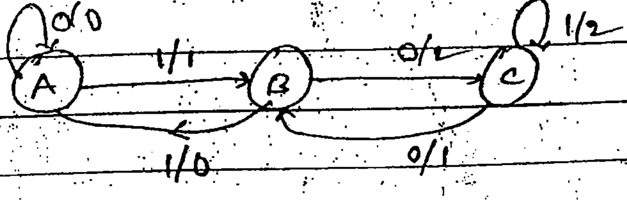
Sol:- 0 → 1

1 → 0



Q:- Me/M/c take all Binary No. as I/P & prints residue Module of 3 as O/P.





Q:-

Equi

No.

Ex①

→

②

→ q0

q1

q2

q3

q4

Equivalance B/w Mealy & Moore M/c.

No. change in No. of states.

Ex①

S	a	b	λ
→A	B	C	0
B	C	A	1
C	A	B	2

Moore

⟺

Mealy

S	a	λ	b	λ
→A	B	1	C	2
B	C	2	A	0
C	A	0	B	1

Mealy

k

O/P.

$\lambda(A) = 0$

$\lambda(B) = 1$

$\lambda(C) = 2$

Sabse pehele Moore M/c se $\lambda(A)$, $\lambda(B)$ $\lambda(C)$ ki value likh to aur Next step me ye values Input value ke samne likh do.

②

s	a	b	λ
→ q_0	q_4 0	q_3 0	0
q_1	q_0 0	q_2 1	1
q_2	q_3 0	q_0 0	1
q_3	q_2 1	q_4 0	0
q_4	q_1 01	q_1 1	0

No. of states Remains Same

Moore ⟹ Mealy.

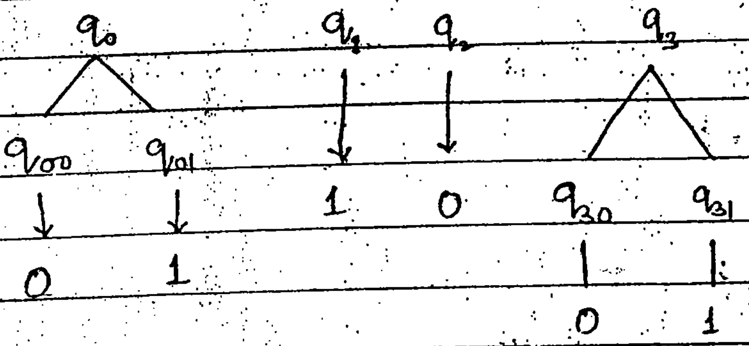
Ex 2 :-

Meelay → Moore :-

- * May be change in no. of states.
- * No. of states depends on Q, Δ.
- * No. of states < $Q \times \Delta$.

Ex :-

δ	a	λ	b	λ
q_0	q_1	1	q_3	0
q_1	q_2	0	q_2	0
q_2	q_0	0	q_0	1
q_3	q_3	1	q_1	1



	a	b	λ
q_{00}	q_1	q_{30}	0
q_{01}	q_1	q_{21}	1
q_1	q_2	q_2	1
q_2	q_{00}	q_{01}	0
q_{30}	q_{31}	q_1	0
q_{31}	q_{31}	q_1	1

Ex 2 :-

δ	a	λ	b	λ
q_0	q_3	0	q_1	0
q_1	q_2	1	q_3	1
q_2	q_4	0	q_5	1
q_3	q_5	0	q_0	0
q_4	q_0	1	q_2	0
q_5	q_1	0	q_4	0

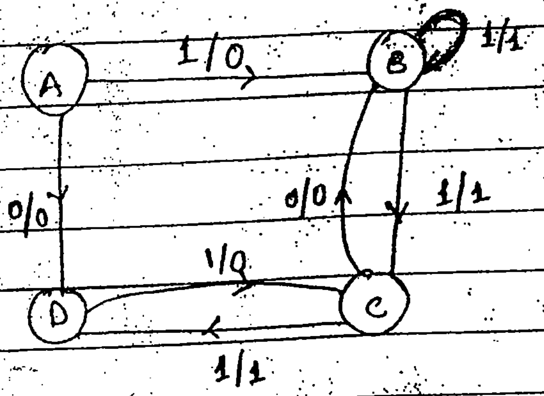
Δ.

δ	a	b	λ
q_{00}	q_{30}	q_1	0
q_{01}	q_{30}	q_1	1
q_1	q_{21}	q_{31}	0
q_{20}	q_4	q_{51}	0
q_{21}	q_{40}	q_{51}	1
q_{30}	q_{51}	q_{00}	0
q_{31}	q_{01}	q_{00}	1
q_4	q_{01}	q_{20}	0
q_{50}	q_1	q_4	0
q_{51}	q_1	q_4	1

Q:- A finite state m/c to the following state table has a single I/P x & single O/P y.

Present state	Next state	
	x = 0	x = 1
A	D, 0	B, 0
B	B, 1	C, 1
C	B, 0	D, 1
D	B, 1	C, 0

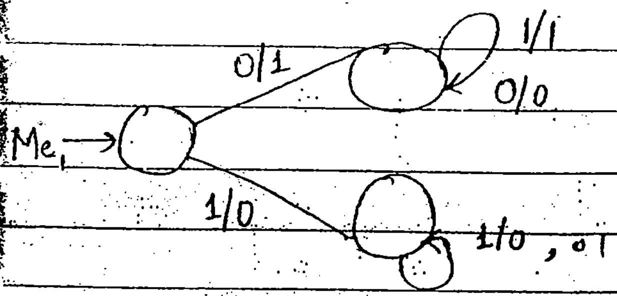
The initial state is unknown then shortest I/P sequence to reach final state.

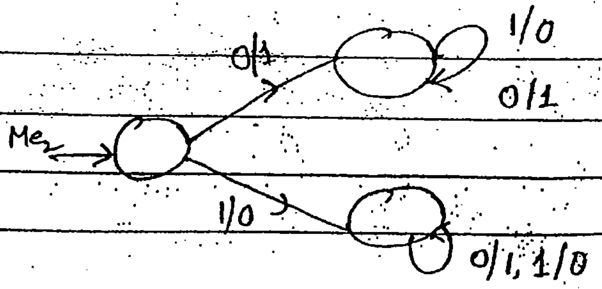


Ans = 01

that say that Me_1, Me_2 as identity property

i.e. $Me_1 Me_2$ = Identity.





ot b.

a) $(Me_1)(Me_2) = (Me_2)(Me_1)$

b) (Me_2) is inverse of Me_1

c) (Me_1) is inverse of Me_2

d) All of these.

is processed

immediately

reprocessed

consider the

o/p string

then we

S.No. Properties.	DFA	NFA	ε N
1. No. of initial state	One	One	One
2. No. of final state	Zero or more	Zero or More	Zero or Mo
3. Edge Labels	frm Σ	from Σ	frm Σ ∪ (
4. No of transition from a state.	$\|\Sigma\|$	Zero or more	Zero or M
5. Tuple Specification	$(Q, \Sigma, \delta, q_0, F)$	$(Q, \Sigma, \delta, q_0, F)$	$(Q, \Sigma, \delta, q_0$
6. Transition f^n	$\delta : Q \times \Sigma \rightarrow Q$	$\delta(Q \times \Sigma) \rightarrow Q$	$\delta : Q \times \Sigma \cup$
7. Output Oriented.	NO	NO	NO
8. Output function	NO	NO	NO
9. Deterministic in Nature.	Yes.	No.	No.

Difference b/w dead state & Unreachable state

① Dead state.

F.A.	FA'
NF	F

② Unreachable state.

FA	FA'
Unreachable	Unreachable.

NFA	ε NFA	Moore	Mealy	Trans. Graph.
ne	One	One	One	One or More
or More	Zero or More	Zero	Zero	Zero or More
Σ	frm $\Sigma \cup (\epsilon)$	frm Σ, Δ	frm $\Sigma\ \Delta$	frm Σ^*
r more	Zero or More	$\lvert\Sigma\rvert$	$\lvert\Sigma\rvert$	Zero or More
$\delta, q_0, F)$	$(Q, \Sigma, \delta, q_0, F)$	$(Q, \Sigma, \Delta, \delta, \lambda, q_0)$	$Q\ \Sigma, \Delta\ \delta, \lambda, q_0$	Q, Σ, δ, I, F set of all initial state.
$\Sigma) \rightarrow Q$	$\delta: Q \times \Sigma \cup (\epsilon) \rightarrow 2^Q$	$\delta: Q \times \Sigma \rightarrow Q$	$\delta: Q \times \Sigma \rightarrow Q$	$\delta: Q \times \Sigma^* \rightarrow 2^Q$
)	No	Yes	Yes	NO
)	NO	$\lambda: Q \rightarrow \Delta$	$\lambda: Q \times \Sigma \rightarrow \Delta$	NO
).	No.	yes.	Yes	No.

reachable state.

ble.

GRAMMER

→ Set of all production rule which are used in generation of a string is called grammer.

Grammer is 4 tuple

$G = (V, T, P, S)$

where,

V = Set of all non terminals or Variables.

T = Set of all terminals.

P = set of all productions

S = start Symbol.

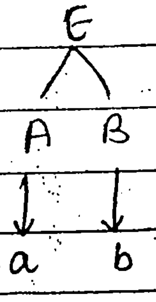
Ex 1 :- $E \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b$

$V = \{E, A, B\}$

$T = \{a, b\}$



$P = \{ E \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b$

$\Rightarrow E \rightarrow ab$

$L(G) = \{a, b\}.$

$S = E$

---

2) $E \rightarrow$

$A \rightarrow$

$E \rightarrow$

$\rightarrow$

L

Q 3 :-

L(G

NOTE :- 1

2) Every

3) A lar

One

Uniqu

4) If G

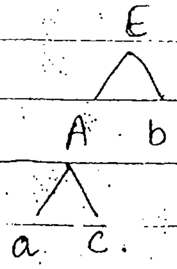
ge

5) Every

6) Let

is a

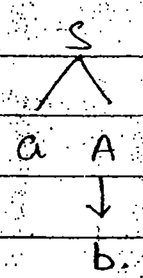
2) E → Ab
A → a/c



E → Ab → ab	E → Ab → cb

L(G) = {ab, cb}.

Q3:- E → aA
A → b



L(G) = {a,b}.

NOTE:- 1) A Grammer is a generating device.

2) Every Grammer represents only one language.

3) A language can be generated by more than one form of grammer i.e. Grammer is not Unique.

4) If G is Grammer than L(G) is language generated by G.

5) Every Grammer has only one start symbol.

6) Let L(G) be language of Grammer G & S is a start symbol of G. Then the string w belongs to L(G). If & only iff $S \rightarrow W$

$W \in L(G) \Leftrightarrow S \rightarrow W$

8/Nov/

Sentential Form :- All the intermediate state involved in derivation is called Sentential form.

e.g. $E \rightarrow AB$
$A \rightarrow a$
$B \rightarrow b$

$$\left.\begin{aligned} E &\rightarrow AB \\ &\rightarrow aB \\ &\rightarrow ab \end{aligned}\right\} \text{Sentential form.}$$

The process of deriving string is called Derivation.

iate state
Sentential
m.
called

8/ Nov/ 10.

Recursive Production :- The Production which has same variable at L.H.S & RHS is c/d Recursive Production.

Recursive Grammer :- The grammer is said to be recursive if it contains atleast one recursive production.

$$\left.\begin{array}{l} A \to \overset{\text{General Recursion}}{aAb} \\ A \to \epsilon \end{array}\right\} \text{Recursive Grammer}$$

Right Recursion
$A \to aA$
$A \to b.$

Left Recursion
$A \to Aa$
$A \to b$

$$\left.\begin{array}{l} E \to E+E \\ E \to id \end{array}\right\}$$

Non Recursive Grammer :- The Grammer which is free from recursive production is c/d Non Recursive Grammer.

① $E \to AB$
$A \to a/b$
$B \to c$

② $S \to aA$
$A \to bB/a$
$B \to a/b.$

NOTE :-

1. The Grammer G generates a language if κ iff G is Recursive Grammer.

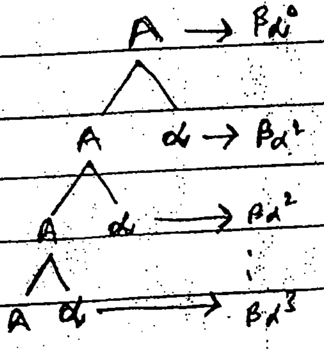
2. Grammer G generates finite Language iff G is Non Recursive Language.

* BNF (Backus Normal form).

$$\left.\begin{array}{l} A \rightarrow \alpha_1 \\ A \rightarrow \alpha_2 \\ A \rightarrow \alpha_3 \\ \vdots \\ A \rightarrow \alpha_n \end{array}\right\} = A \longrightarrow \alpha_1 / \alpha_2 / \alpha_3 / \cdots \alpha_n$$

$A \rightarrow A\alpha / \beta$

Recursive production ke sath jo production hoga us par * lag Jayega.



$A \rightarrow A\alpha / \beta = \beta\alpha^*$

$A \rightarrow \alpha A | \beta = \alpha^* \beta$

$A \rightarrow aA | \epsilon \Rightarrow A \rightarrow a^*$

$A \rightarrow aA / a \Rightarrow A \rightarrow a^+$

$A \rightarrow Aa / a \Rightarrow A \rightarrow a^+$

zge ij
ncy.

* $A \rightarrow aA / bA / \epsilon$

G is

$\Rightarrow (a+b)^* = \Sigma^* \qquad \Sigma = \{a,b\} = T$

* $A \rightarrow aA / bA / a / b$

$\Rightarrow A\ (a+b)^+ = \Sigma^+ \qquad \Sigma = \{a,b\} = T.$

-- α_n

Cons. of Grammar For finite Languages :-

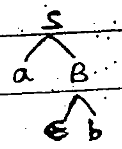
1) $L = \{a\}$

$S \rightarrow a$

③ $L = \{\underline{a}, \underline{ab}\}.$

$S \rightarrow aB$

$B \rightarrow \epsilon / b$



Kc satn jo
Jayega.

(2) $L = \{\epsilon, a\}$

$\beta\alpha^*$

$S \rightarrow \epsilon / a$

④ $L = \{a, \underbrace{ab}_{X}, b\underbrace{ab}_{X}\}.$

$^*\beta$

$S \rightarrow a / X / bX$

$X \rightarrow AB$

$\rightarrow a^*$

$A \rightarrow a$

$B \rightarrow b$

$\rightarrow a^*$

5) $L = \{\underline{ab}, b\underline{ab}, a\underline{ab}, \underline{ab}a\}$

$S \rightarrow X / bX / aX / Xa$

$X \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b$

Q Lang. generated by Grammer. $S \rightarrow aA$

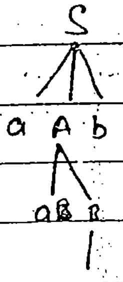
$A \rightarrow b/c$

$L(G)' = \{ab, ac\}$

Q :- Language Generated by Grammer. $S \rightarrow aAb/Ba$

$A \rightarrow aB/b$

$B \rightarrow aC/ a$



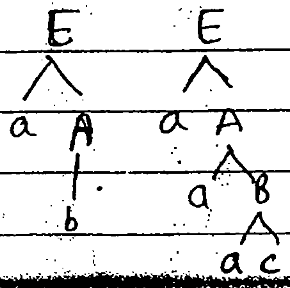
$L = \{aa, aab, aaab\}$

aaab, aa

Q :- Language Generated by Grammer $E \rightarrow aA/bB$

$A \rightarrow aB/b$

$B \rightarrow a/c$



$E(G) = \{ab, aaa, aac, ba, bc\}$

Q:- Comp

b

(a) Incl

Sol:-

(a)

(b)

Q:- Comp.

When

Sol:-

S

X

Q:- W

Sol:-

Construction of Grammer for infinite Language

Q:- Cons G that generates all string of a's & b's

(a) Including ε (b) excluding ε.

sol:- $\Sigma = \{a,b\}$

A

/c

(a) $\Sigma^* = (a+b)^*$

$S \rightarrow aS / bS / \epsilon$

(b) $\Sigma^+ = (a+b)^+$

$S \rightarrow aS / bS / a / b.$

aAb/Ba

aB/b

a/ a

Q:- Cons. G that generate all string of a's & b's where every string starts with a.

Sol:-

$\underset{(a+b)^*}{aX}$

$S \rightarrow aX$

$X \rightarrow aX / bX / \epsilon.$

aA/bB

aB/b

a/c

Q:- Where every string ends with ba.

1, aac, ba, bc

Sol:- $S \rightarrow Aba$

$A \rightarrow aA / bA / \epsilon$

$\underset{(a+b)^*}{\underset{\downarrow}{X}}ba$

Q:- Where every string starts and ends with A.

Sol:- $S \rightarrow aAa \,/\, a$ $\quad$ aAa, $\downarrow$ $(a+b)^*$

$A \rightarrow aA \,/\, bA \,/\, \epsilon$

Q:- Where every string starts & end with same symbol:

bXb

aXa

$S \rightarrow bXb \,/\, aXa \,/\, b \,/\, a$

$X \rightarrow aX \,/\, bX \,/\, \epsilon$

Q:- Where every string string start & end with diff. symbol.

Sol:- $S \rightarrow aXb \,/\, bXa$

$X \rightarrow aX \,/\, bX \,/\, \epsilon$

Q:- Where each string contains exactly 2 a's.

Sol:- $S \rightarrow XaXaX$ $\quad$ $W = XaXaX$, $b^*ab^*ab^*$

$X \rightarrow bX \,/\, \epsilon$

l with A.

Q:- Where each string Contains 2 a's.

$S \rightarrow XaXaX$

$X \rightarrow aX/bX/\epsilon$

$w = XaXaX$
$\downarrow$
$(a+b)^*$

)*

same

Q:- Where each string Contains atmost 2 a's.

b

B

Sol $S \rightarrow XYXYX$

$X \rightarrow bX/\epsilon$

$Y \rightarrow a/\epsilon$

$X\ (a+\epsilon)\ X\ (a+\epsilon)\ X$

$X \quad Y \quad X \quad Y \quad X$

$b^* \quad b^* \quad b^*$

Q:- Where No. of a's in string is even.

end with

$|w|_a = 0 \bmod 2$
$\downarrow$
$[b^*ab^*ab^*]^* + b^*$

$S \rightarrow XS/\epsilon/B$

$X \rightarrow BaBaB$

$B \rightarrow bB/\epsilon$

fly 2 as.

$= XaXaX$

$b^*ab^*ab^*$

Q:- Where 3rd symbol from left end is always B.

Sol:- $W = XXb....$

$(a+b)^2 b\ (a+b)^*$

$\quad A \qquad\quad B$

$S \rightarrow AbB$

$A \rightarrow XX$

$X \rightarrow a/b$

$B \rightarrow aB/bB/\epsilon$

Q:- 3rd symbol from right end is always a.

sol:- W = ---a X X $\qquad \underset{A}{(a+b)^*} \, a \, \underset{B}{(a+b)^2}$

S → AaB

A → aA/bA/ε

B → XA

X → a/b.

Q:- Length of string is exactly 3.

sol:- W = XXX = $(a+b)^3$

S → XXX

S → a/b.

Q:- Length of string is atleast 3.

sol:- W ≥ 3

$(a+b)^3 (a+b)^*$

S → AB

A → XXX

X → a/b

B → aB/bB/ε

Q:- Le

Sol^n:- l

Q:- Wher

Sol^n:-

S –

X –

A

Q:- L

Sol^n:-

ray a.

$b)^2$

3

Q:- Length of string is atmost 3.

Solⁿ:- $|W| = 0, 1, 2, 3.$

$$re = (a+b+c)^3$$

$$S \rightarrow XXX$$

$$X \rightarrow a/b/\epsilon$$

Q:- Where length of string is divisible by 3.

Solⁿ:- $W = 0 \bmod 3$

$$\downarrow$$

$$[(a+b)^3]^*$$

$$S \rightarrow XS/\epsilon$$

$$X \rightarrow AAA$$

$$A \rightarrow a/b$$

Q:- $L = \{a^m b^n \mid m, n \geq 0\}.$

Solⁿ:- $L = \{a^*, b^*\}$

$$\begin{matrix} A & B \end{matrix}$$

$$S \rightarrow AB$$

$$A \rightarrow aA/\epsilon$$

$$b \rightarrow bB/\epsilon$$

$n, m \geq 1$

a^+, b^+

$$S \rightarrow AB$$

$$A \rightarrow aA/a$$

$$B \rightarrow bB/b$$

Q :- $L = \{ a^m, b^n / m = n \}$

$m, n \geq 0$

Sol^n :- ϵ, ab, aabb, aaabbb, ...

$S \longrightarrow aSb / \epsilon$

Q :- $L = \{ a^m b^n / m < n \}$ $m, n \geq 1$

abb, aabbb, aaabbbb

Sol :- $S \longrightarrow aSb / aAb$

$A \longrightarrow bA / b$

Q :- $L = \{ a^m b^n / m > n \}$

aab, aaabb,

$S \longrightarrow aSb / aAb$

$A \longrightarrow aA / a.$

Q :- $L = \{ a^m b^n / m \neq n \}.$

$\{a^m b^n / m < n\} \cup \{ a^m b^n / m > n \}.$

Q :- L

S

Q :- L

S

A

B

Q :- L

S

A

B

Q :- L

$S \rightarrow S_1 / S_2$

$S_1 \rightarrow aS_1b / aAb$

$A \rightarrow aA / a$

$S_2 \rightarrow aS_2b / aBa$

$B \rightarrow bB / b$

Q:- $L = \{ a^n b^{2n} / n \geq 0 \}$ abb, aabbbb, ...

$S \rightarrow aSbb / \epsilon$

Q:- $L = \{ a^m b^n c^p / n = m+p \}$ $m,n,p \geq 0$

$a^m b^{m+p} c^p$

$\underbrace{a^m b^m} \underbrace{b^p c^p}$

$S \rightarrow AB$

$A \rightarrow aAb / \epsilon$

$B \rightarrow bBc / \epsilon$

Q:- $L = \{ \underset{A}{a^m} \underbrace{b^n c^n}_{B} / m,n \geq 0 \}$

$S \rightarrow AB$

$A \rightarrow aA / \epsilon$

$B \rightarrow bBc / \epsilon$

Q:- $L = \{ a^n b^m c^n / m,n \geq 0 \}$

$S \rightarrow aSc / aAc$

$A \rightarrow bA / \epsilon$

Q:- $L = \{ \underbrace{a^m \underbrace{b^n c^n}} d^m \mid m, n \geq 0 \}$.

sol:-

$S \rightarrow aSd \mid aAd \mid \epsilon$

$A \rightarrow bAc \mid \epsilon$

Q:- $L = \{ (ab)^n \mid n \geq 1 \}$. ab, abab, ababab

$S \rightarrow abS \mid ab$.

Q:- $L = \{ a^n b^n c^n \mid n \geq 1 \}$.

$S \rightarrow aSAc \mid abc$

$cA \rightarrow Ac$

$bA \rightarrow bb$

Q:- $L = \{ ww^R \mid w \in (a,b)^* \}$.

$S \rightarrow aSa \mid bSb \mid \epsilon$

$L = \{ wcw^R \mid w \in (a,b)^* \}$.

$S \rightarrow aSa \mid bSb \mid c$

Q:- L

S

#

Based
classifie

1. Typ
2. Ty
3. Ty
4) Ty

1) To
typ

NOTE :-
* Eve
at

Ex.

Q:- $L = \{ w \in (a,b) / |w|_a = |w|_b \}$

$S \rightarrow SaSbS / SbSaS / \epsilon$

CHOMSKY Hirarchy.

ababab

Based on form of Production Grammer Can be classified into 4 types.

1. Type 0 or REG.
2. Type ~~0~~ 1 / CSG
3. Type 2 / CFG
4) Type 3 / Regular G.

1) T_0 / REG :- The grammer G is said to be type 0 if every production is in the form

$$\alpha \longrightarrow \beta \quad , \quad \alpha, \beta \in (V+T)^*$$

NOTE :-

* Every Grammer should hv atleast one variable at LHS of any production.

?

Ex. $S \rightarrow aSAc / \epsilon$ $\qquad$ $S \rightarrow aASb$

$cA \rightarrow Ac$ $\qquad$ $aA \rightarrow ab$

$bA \rightarrow bb$ $\qquad$ $bA \rightarrow aAb$

2) T_1 or CSG :- Grammer G said to be of T_1/CSG if every production is in the form,

$$\alpha \rightarrow \beta \quad , \quad |\alpha| \leq |\beta| \quad \alpha, \beta \in (V+T)^*$$

$$\Rightarrow \beta \neq \epsilon$$

$S \rightarrow \epsilon$

$|S| = |\epsilon|$ Ex :- $S \rightarrow aASc$

$1 \overset{}{\underset{\times}{=}} 0$ $cA \rightarrow Ac$

$bA \rightarrow bb$

Note :-

1) If $B = \epsilon$ $\alpha \rightarrow \beta \Rightarrow \alpha \rightarrow \epsilon \Rightarrow 1 < 0$ Not possible.

2) CSG doesn't contain any production that contains ϵ.
i.e. Lang. generated by ϵ is free from ϵ.

3) T_2 or CFG :- Grammer G is said to be T_2 or CFG if every production is in the form

$$\text{only 1 V} \leftarrow A \rightarrow \underset{\text{No restriction}}{\underset{\downarrow}{\alpha}} \qquad A \in V$$

$$\alpha \in (V+T)^*$$

Ex.

1) $S \rightarrow aSb/\epsilon$ $E \rightarrow E*E$

2) $S \rightarrow aSbS/a/b$ $E+E$

3) $S \rightarrow Sab/\epsilon$ int

4) T_3 or

is

At least

Ex :- 2

① S

② S

A

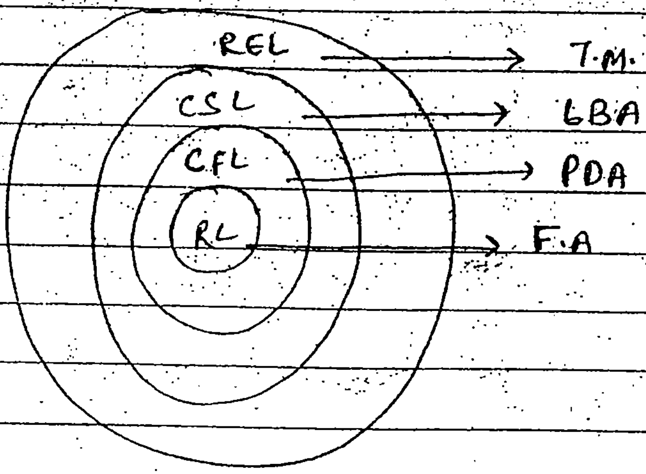
NOTE :-

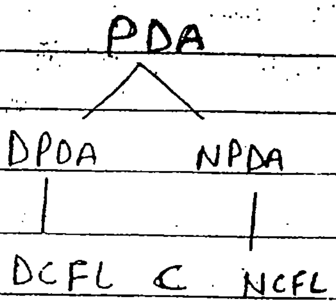
1.

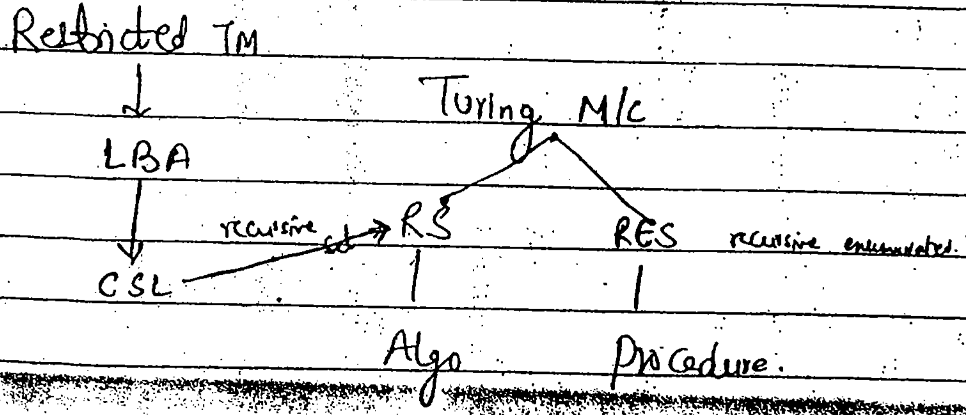
$\Rightarrow$

Type of Formal G Language:- It can be classified into 4 types.

1) T_0 or REL
) T_1 or CSL
) T_2 or CFL
) T_3 or Regular generated by ~~geo~~regular Grammar







4) T_3 or Regular Grammer :- Grammer G is said to be T_3 or Regular if every production is in the form.

$$A \to xB / Bx / x \qquad A, B \in V$$

At least One variable on RHS. $x \in T^*$

Ex :- 2 Consecutive Variable are Not possible.

① $S \to 001\cancel{1}S/1$

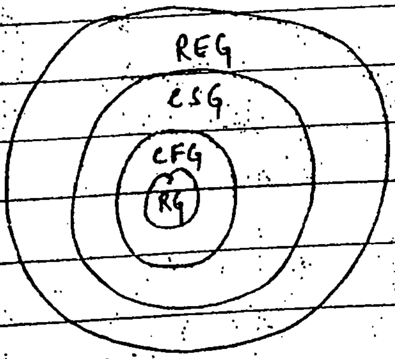
③ $S \to S101/01001$
$A \to A110/0.$

② $S \to 101S/0A$
$A \to 100A/0$

NOTE :- Hum kabhi bhi lagatar 2 variable nahi Hoskte Hai.

1. $T_3 \subset T_2 \subset T_1 \subset T_0$

$\Rightarrow$ RG $\subset$ CFG $\subset$ CSG $\subset$ REG



CSG = CSG $\cup$ {ε}

Can

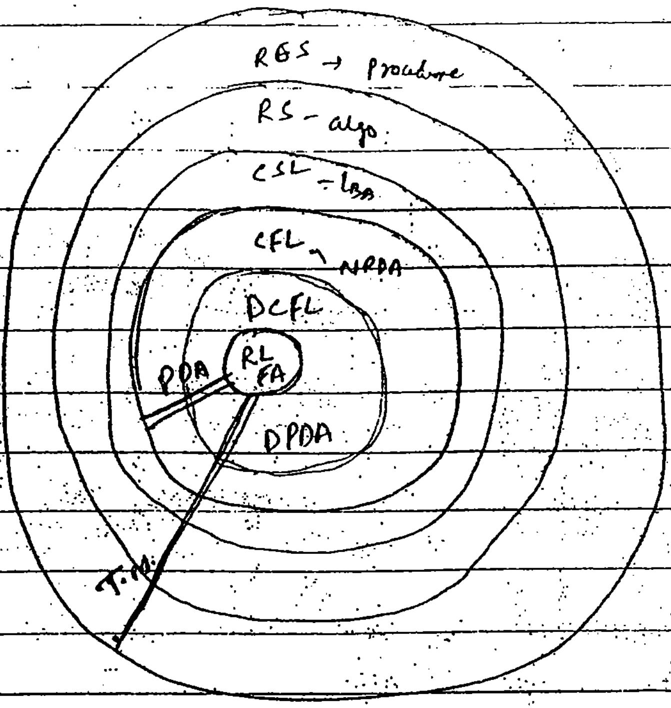
ılar Grammar

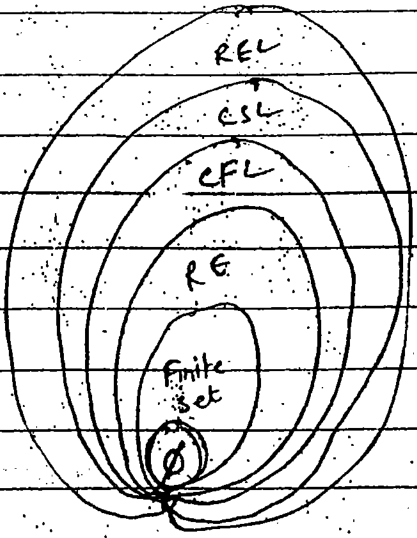
T.M.

LBA

PDA

FA





ive enumerated

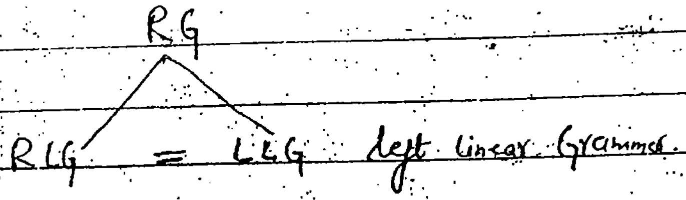
9/11/10.

Regular Grammer :- that generates Regular Lang.

Grammer G is said to be Regular if production is in form

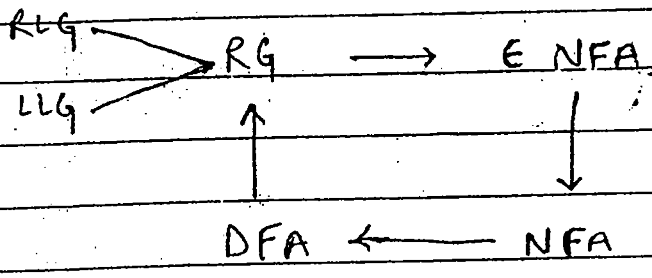
$$A \rightarrow Bx / xB / x \qquad A, B \in V$$
$$x \in T^*$$

Types of Regular Grammer :- Regular Grammer is of 2 types.



Note :-
1. Regular Grammer is rep by either RLG or LLG.

Equivalence b/w R. Grammer & F.A.



Method

1) I

② δ(

③ F

wher

Q :-

S

ular Lang.

production

∈ V

T*

c is

Grammar

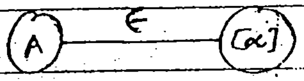
RLG or LLG

Non Linear

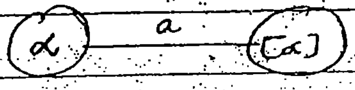
Method → RLGram to ∈ NFA

1) If $A \rightarrow \alpha$ is a production then

$$\delta(A, \epsilon) = [\alpha]$$

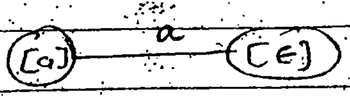


② $\delta(a\alpha, a) = [\alpha]$



③ For every terminal $a \in T$

$$\delta(a, a) = [\epsilon]$$

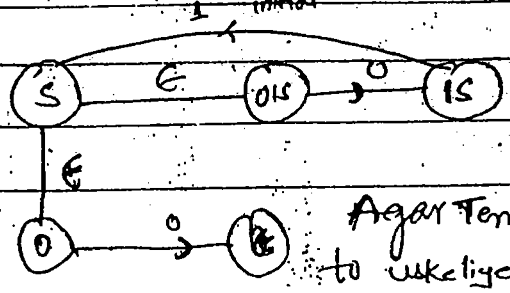


where [α] is R.H.S of any production or ~~Va~~ Suffix for RHS of production.

Q:- $S \rightarrow \underline{01S} / 0$

$Q = [S], [01S] [0] [\epsilon] [1S]$

↑ initial ↑ final



Agar Terminal hai to uskeliye ∈ bhi ←

$\delta(S, \epsilon) = (01S)$

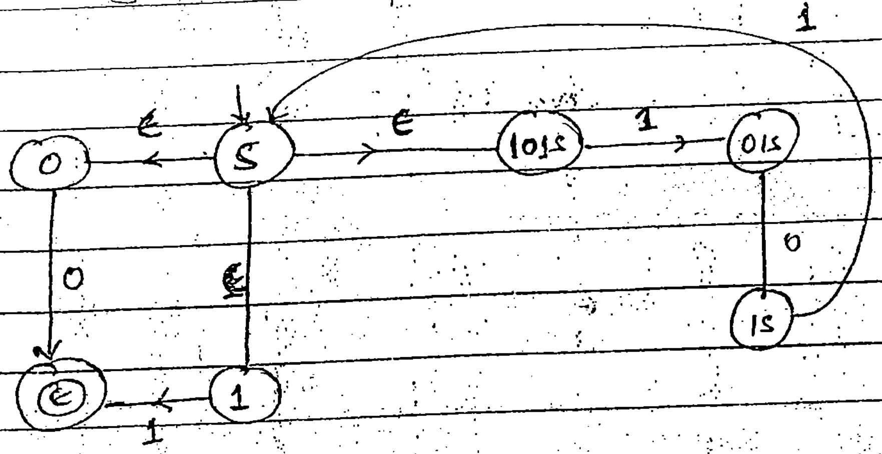
$= 0$

$\delta(01S, 0) = (1S)$

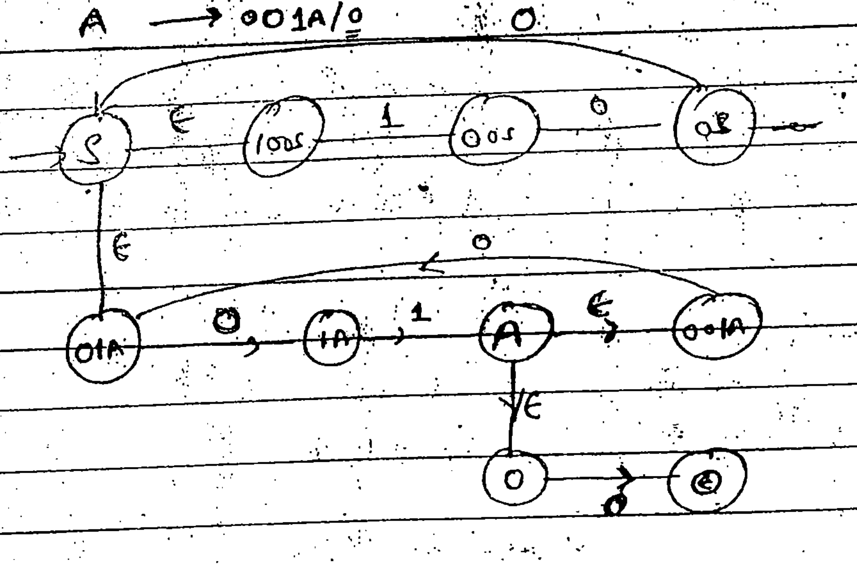
$\delta(1S, 1) = S$

$\delta(0, 0) = \epsilon$

② $S \longrightarrow 101S/0/1.$



③ $S \longrightarrow 100S / 01A$
$A \longrightarrow 001A/0$



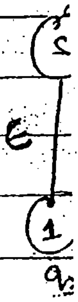
RLG → $A \rightarrow xB / x$ LLG $A \rightarrow Bx/x$.

1.1 Kam karte jao aur uski Next state Banate jao.

Agar terminal dia hai to usko final state me dal do.

Agar last me Terminal hai upar wale I/p par pahuchate hai, Aur ∈

Q:- f
a

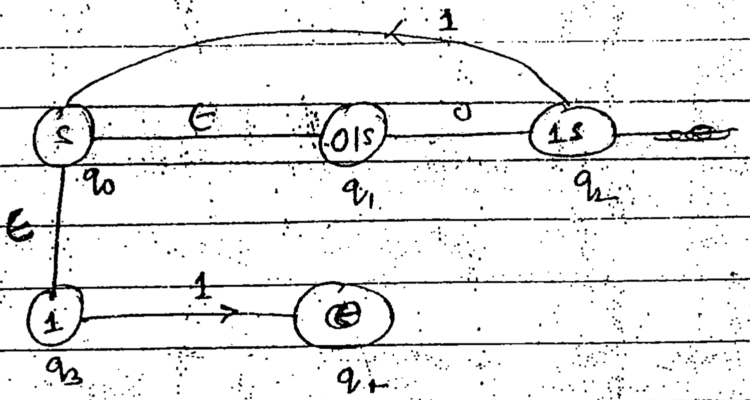


NFA

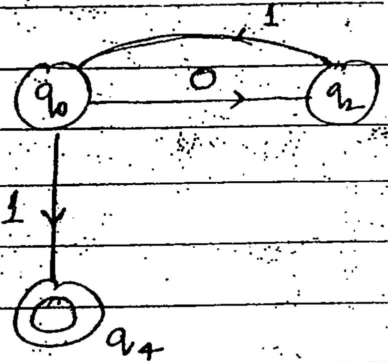
DFA :

Q:- find No. of state in Minimal FA that accept all strings generated by Grammar

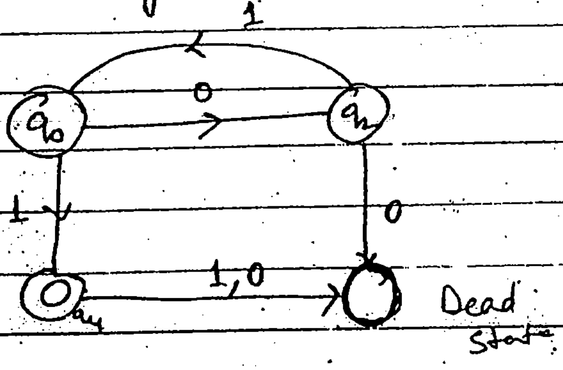
$S \rightarrow 01S / 1$



NFA



DFA :- No. of states in DFA is 4



→ Bx/x

Next

who

or whic I/

Ans. ε

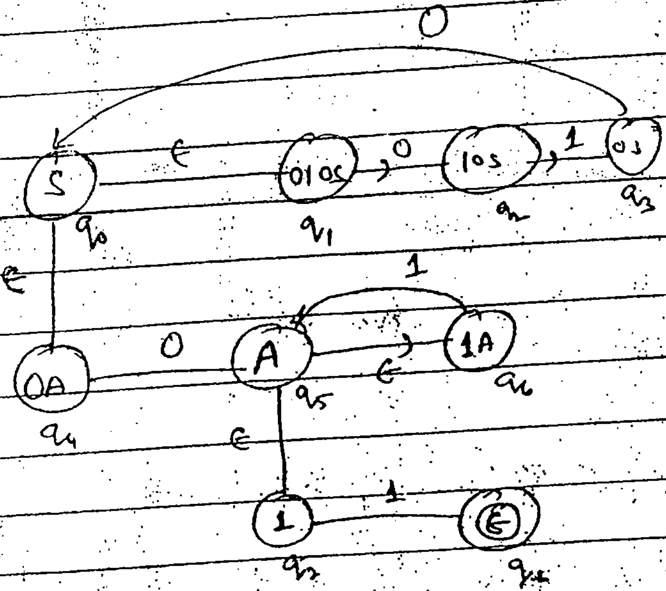
C

D

Q:- No. of states min FA. of Grammer:-

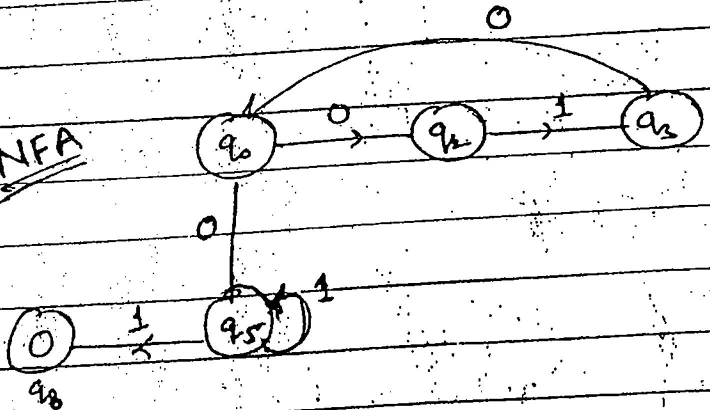
$S \rightarrow 010S / 0A$

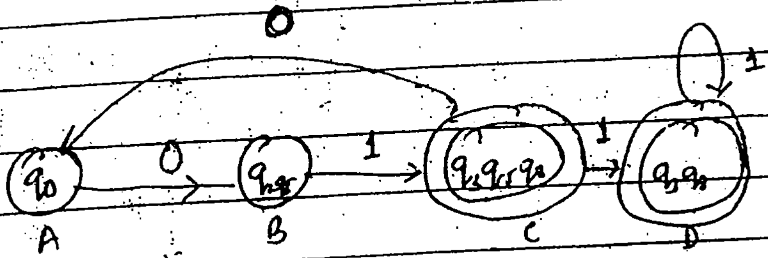
$A \rightarrow 1A / 1$



Q:-

NFA





(C,D)
G1

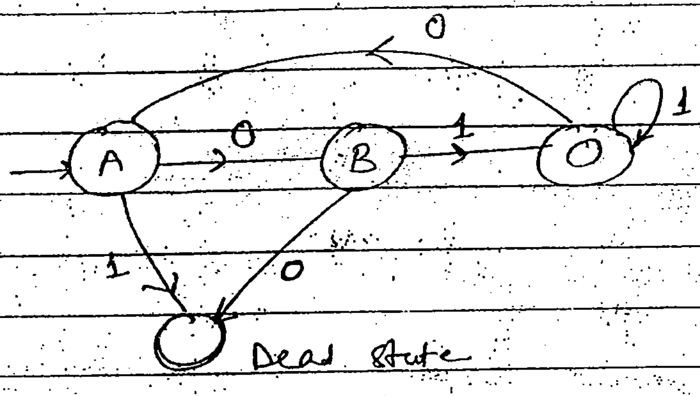
(A,B)
G2

md :-

C	G_2	G_1
D	G_2	G_2

⇒ C = D

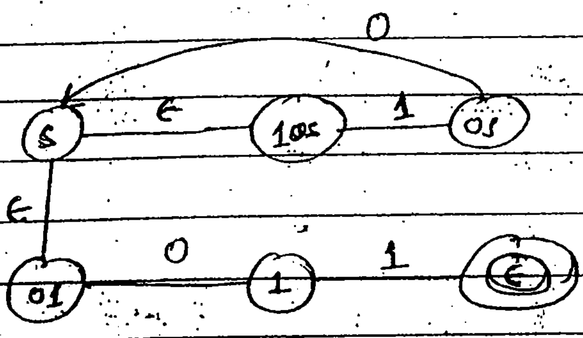
(A), (B) (C, D).

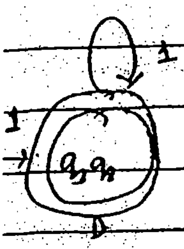


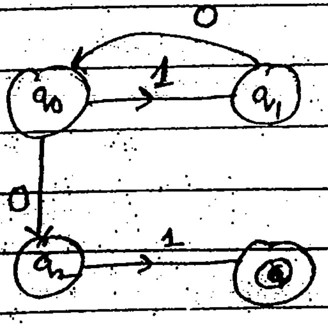
No. of states = 4.

Q:- No. of Min states in Grammer.

S ⟶ 10S / 01







Total 5 States

ε NFA

Method :- from LLG to Finite Automata.

① Reverse RHS of every production.

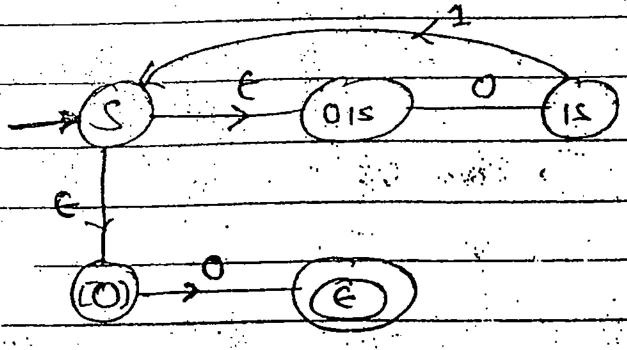
② Obtain ε NFA.

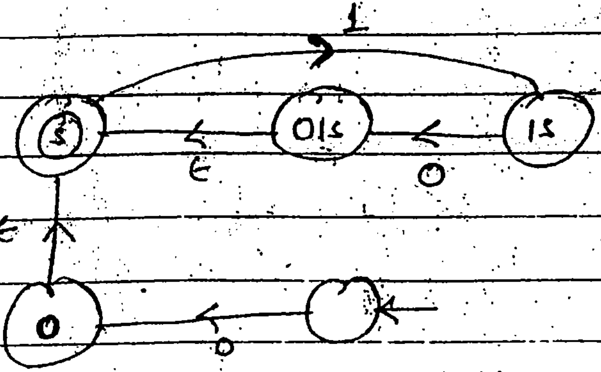
③ Interchange initial & final states.

④ change the direction of edges.

Q :- S → S10/0

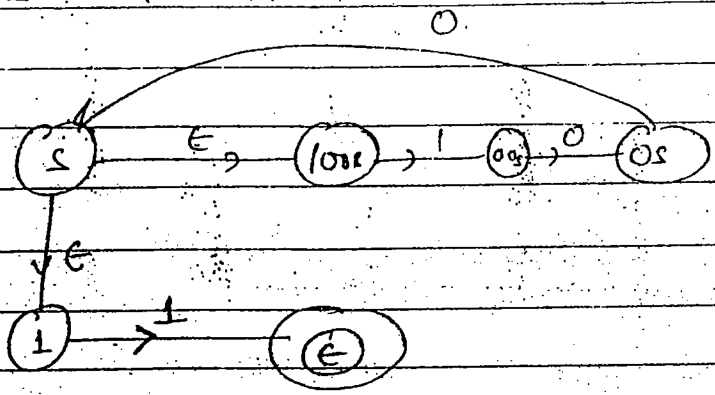
① S → 01S/0

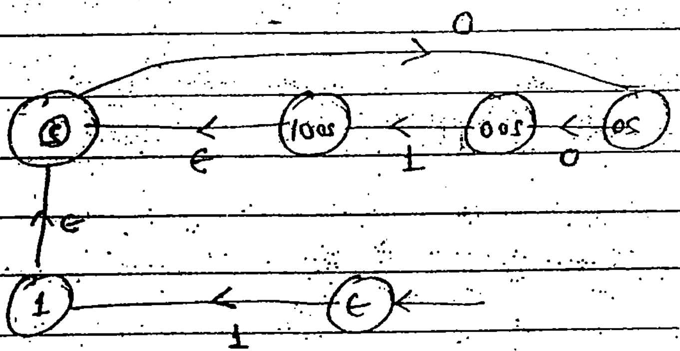


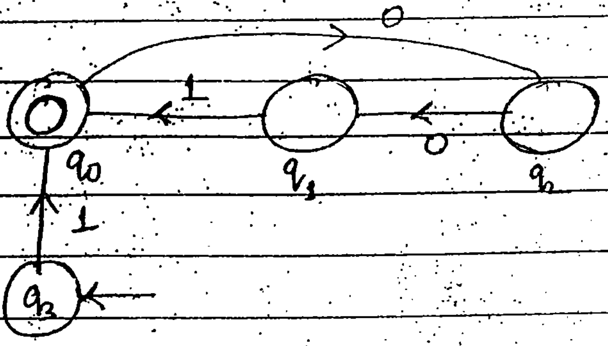


② S → S001/1

S → 100S/1





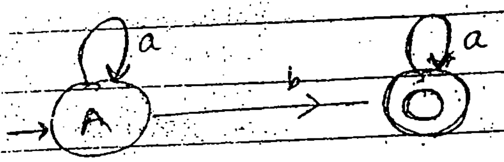


Total No states = 4 + 1
↑
Dead state

Equivalance b/w F.A. & Regular Grammer.

Q :-

Case (1) :- F.A. $\longrightarrow$ RLG



Q :-

No. of Variable = No. of states

Initial state = Start Symbol.

If $\delta(A, a) = B$ is transitive then

$A \rightarrow aB$ is a production

If B is final state. then add

Q :-

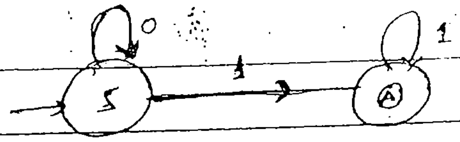
$B \rightarrow \epsilon$

or

$A \rightarrow a$

rammar.

Q :-



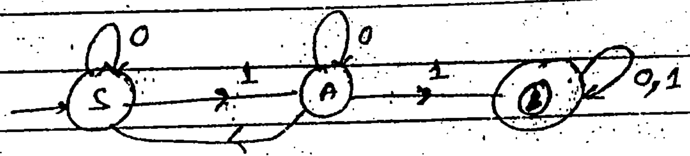
$S \rightarrow 0S / 1A / 1$ → B/c A is final state.

$A \rightarrow 1A / 1$

~~$A \rightarrow \epsilon$~~ OR we can add.

$A \rightarrow \epsilon$

Q :-



tes

d.

$S \rightarrow 0S / 1A$

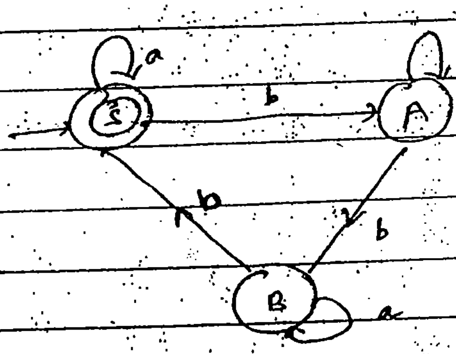
$A \rightarrow 0A / 1S / 1B / 1$

$B \rightarrow 0B / 1B / 1 / 0$

final state ke sath jitne Bhi Terminal Honge wo sab last me Likhte hai

Q :-

add.



$S \rightarrow aS / bA / a$

$A \rightarrow aA / bB$

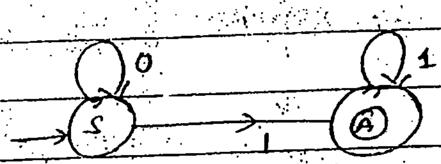
$B \rightarrow aB / bS / b$

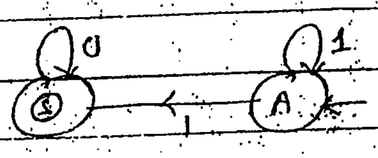
Case II :- FA to LLG.

① Interchange initial & final states.

② Change the direction of labels.

Ex :- ①

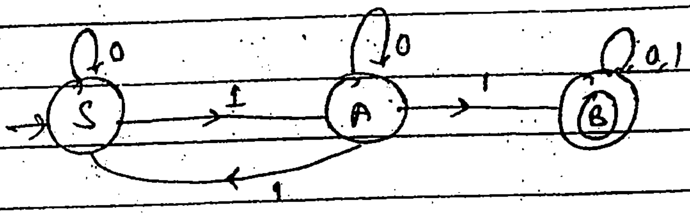


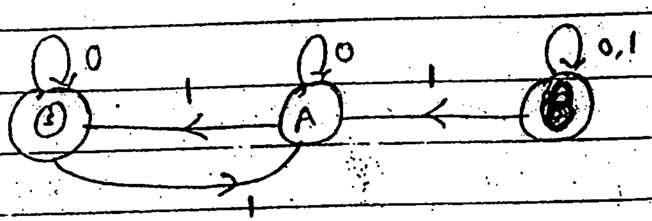


$A \rightarrow A1 / S1 / 1$

$S \rightarrow S0 / 0$

②





$S \rightarrow S0 / 0 / A1 \quad B \rightarrow B0/$

Case 3 :- Regular Grammer to R.E.

$A \rightarrow \alpha / \beta / \gamma$

$= \alpha + \beta + \gamma$

$= \alpha + \beta / \gamma$

$= \alpha + \gamma / \beta$

$= \gamma + \beta / \alpha$

$A \rightarrow \alpha A / \beta$

$= \alpha^* \beta$

$A \rightarrow A\alpha / \beta$

$\beta \alpha^*$

states.

xls.

Variable ke sath jo bhi terminal hoga uska * lenge.

Q:- $S \rightarrow 10A / 00B$

$A \rightarrow 10A / 0 \quad = (10)^* 0$

$B \rightarrow 11B / 01 \quad = (11)^* 01$

$S \rightarrow ($ [illegible] $10(10)^* 0 + 00(11)^* 01$

$RE = (10)^+ 0 + 00(11)^* 01$

0,1

Q:- $S \rightarrow 11A / 01B$

$A \rightarrow 1A / 01 \rightarrow 1^*(01)$

$B \rightarrow 101B / S \rightarrow (101)^* S$

0,1

$S \rightarrow 11(1)^* 01 + 01(101)^* S$

$S \rightarrow \beta / \alpha S$

$\Downarrow$

$\alpha^* \beta$

$$\boxed{[01(101)^*]^* \; 11^+ 01}$$

$B \rightarrow B0/B1$

Q :- $S \rightarrow 1A / 0B / \epsilon$

$A \rightarrow 0A / S \longrightarrow 0^* S$

$B \rightarrow 1B / S \longrightarrow 1^* S$

$S \rightarrow 10^* S / 01^* S / \epsilon$

$S \rightarrow (10^* + 01^*) S / \epsilon$

$RE = (10^* + 01^*)^*$

Equivalance b/w RLG & LLG.

① Reverse the RHS of productions.

② Obtain Regular expression.

③ Reverse the Regular expression.

④ = ② { Obtain the RLG

⑤ = ① Reverse RHS of every production.

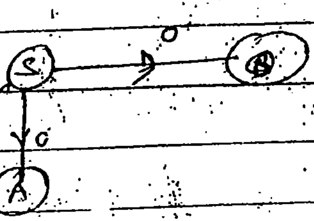
$S \rightarrow 01S/0$

$S \rightarrow S10/0$

$RG = 0(10)^*$

$Rev\ RG = (01)^*0$

④



LLG = $S \rightarrow B0/0$

$S \rightarrow A0$

$A \rightarrow S1$

2

Case 2 :- LLG to RLG

ssion.

① Obtain the RE.

② Reverse the RE.

③

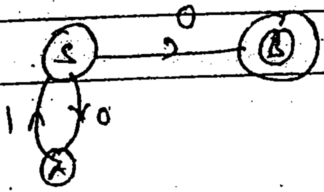
Induction.

Ex :- $S \rightarrow S10/0$

① RE ⇒ $0(10)^*$

$(01)^*0$

RLG :-



$S \rightarrow 0B/0$

$S \rightarrow 0A$

$A \rightarrow 1S$

10/11/2010.

CFG, CFL, PDA

CFL
^
CFG = PDA

CFG :- Grammer which generates CFL is cld CFG.

Grammer G said to be CFG if production is in the form.

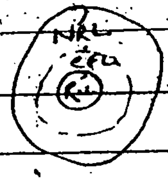
$$A \longrightarrow \alpha \qquad A \in V$$
$$\alpha \in (V+T)^*$$

CFL :- Language which is ~~app~~ accepted by PDA or generated by CFG is called CFL.

NOTE :-

1. Every Regular language is CFL

2. Some N.R. Lang is CFL.



example :- ① $L = \{ a^m b^n / m = n \}$ → CFL

② $L = \{ a^m b^n / m < n \}$

③ $L = \{ a^m b^n / m > n \}$

④ $L = \{ a^m b^n / m \neq n \}$

⑤ L=

⑥ L = {

⑦ L = {a

⑧ L =

⑨ L = {

⑩ L = {a

⑪ L = {

⑫ L = {

⑬ L =

⑭ L = {

⑮ L = {

⑯ L = {

⑰ L = {

⑱ L = {

⑲ L =

⑳ L =

CFL

CFG = PDA

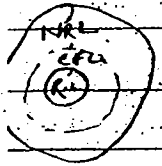
is cld

production

1+T)*

by PDA

called CFL.



⑤ $L = \{ a^m b^n / m = 2n \}$

⑥ $L = \{ a^m b^n / \gcd(m,n) = 1 \}$

⑦ $L = \{ a^m b^n c^p / m = n \}$

⑧ $L = \{ a^m b^n c^p / m = n \ \& \ n = p \}$.

⑨ $L = \{ a^m b^n c^p / m+n = p \}$

⑩ $L = \{ a^m b^n c^p / m = np \}$

⑪ $L = \{ a^m b^n c^n d^m / m,n \geq 0 \}$

⑫ $L = \{ a^m b^n a^n / m,n \geq 0 \}$

⑬ $L = \{ w \in \{a,b\}^* / |w|_a = |w|_b \}$.

⑭ $L = \{ w \in (a,b,c)^* / |w|_a = |w|_b + |w|_c \}$

⑮ $L = \{ w \in (a,b)^* / |w|_a < |w|_b \}$.

⑯ $L = \{ w c w^R / w \in \{a,b\}^* \}$.

$L = \{ w w^R / w \in (a,b)^* \}$.

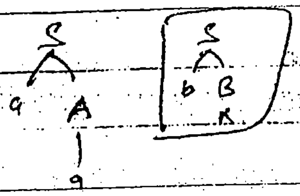
$L = \{ c w w^R / w \in (a,b)^* \}$.

$L = \{ ww / w \in (a,b)^* \}$. → Not CFL.

$L = \{ w \in (a,b,c)^* / |w|_a = |w|_b = |w|_c \}$. Not CFL

$S \rightarrow aA \mid bB$

$A \rightarrow a$

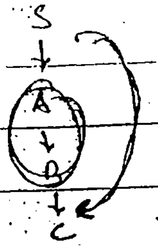


Useless Production

$S \rightarrow A$

$A \rightarrow B$

$B \rightarrow C$



1,

Unit production

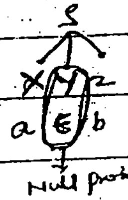
Just forwding

$S \rightarrow XYZ$

$X \rightarrow a$

$Y \rightarrow \epsilon$

$Z \rightarrow b$



Null prod
Nhi Hone se grammar parkoi fark ni padega..

Simplification of CFG :-

1. The process of detection & elimination of Variables & productions whose presence or absence will not affect the language of the grammer is c/d simplification of CFG.

→ The presence ~~of~~ or absence of useless Symbol, Unit Productions & ~~useless~~ Null productions will not affect the lang. of Grammer.

Useless Symbols :- The variable which is not involved in derivation of any string is called Useless Symbol.

Note :-

① If any variable which is not reached by

---

② Any Va
Nhi

ex.

$S \rightarrow aA$

$A \rightarrow b$

③ Gramme
Syn

① Ex :- S –
A –
B –

③ Ex

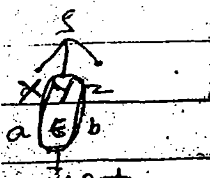
④ S
A
B

$S \to xYz$

$x \to a$

$Y \to \epsilon$

$z \to b$



Null prodn Hon se grammar mein koi farke ni padga..

② Any variable derived by S pr uska koi terminal Nhi hai to wo bhi Useless hai.

ex. $S \to aA / \overset{\alpha}{\boxed{bB}} \to$ B doesn't derive any production so delete whole production

$S \to aA$ $\Leftarrow$ $A \to b$

$A \to b$

③ Grammer obtain by after elimination of Useless symbols of related production is c'ld Reduced CFG.

① Ex :- $S \to aA$ $\quad\Rightarrow\quad$ $S \to aA$

$A \to b$ $\quad\quad\quad\quad$ $A \to b$

$\boxed{B \to a}$ Not reached by start symbol.

③ Ex $S \to aAb / B\overset{\alpha}{ac}$

$A \to aB / b$ $\quad\Rightarrow\quad$ $S \to aAb$, $A \to aB / b$, $B \to b$.

$B \to \underset{\alpha}{ac} / b$

④ $S \to aA / Ba / A\overset{\alpha}{bc}$ $\quad\quad$ $S \to aA / Ba$.

$A \to aB / bCa^{\alpha}$ $\quad\quad$ $A \to aB$

$B \to aA / b$ $\quad\Rightarrow\quad$ $B \to aA / b$.

$\boxed{C \to D}$

nation of

presence

the

cld

seless

Null

the lang.

hich is

nation of

ess Symbol

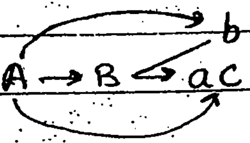
reached by

Delete it.

Unit Productions :- production of form $A \rightarrow B$, $A, B \in V$ is called Unit productions.

① Select all unit productions & delete them by replacing the equivalent derivations.

Example :-



① $S \rightarrow aA / bB$

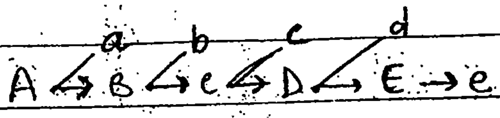
$A \rightarrow B / a$

$B \rightarrow ac / b \Rightarrow S \rightarrow aA / bB$

$C \rightarrow c.$ $\quad A \rightarrow \cancel{b} / aC / a$

$B \rightarrow aC / b$

$C \rightarrow c$



② $S \rightarrow aAb / Bb$ $\quad S \rightarrow aAb / Bb$

$A \rightarrow B / a$ $\quad A \rightarrow a/b/c/d/e$

$B \rightarrow C / b$ $\quad B \rightarrow b/c/d/e$

$C \rightarrow D / c$ $\quad C \rightarrow c/d/e$

$D \rightarrow E / d$ $\quad D \rightarrow d/e$

$E \rightarrow e.$ $\quad E \rightarrow e$

$\left.\begin{matrix} C \rightarrow c/d/e \\ D \rightarrow d/e \\ E \rightarrow e \end{matrix}\right\}$ Useless Symbols

Useless Symbols are C, D, E b/c they are not reached by starting variable. So,

$$\left.\begin{matrix} S \rightarrow aAb / Bb \\ A \rightarrow a/b/c/d/e \\ B \rightarrow b/c/d/e \end{matrix}\right\}.$$

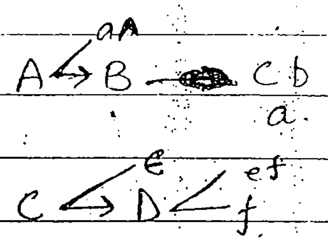
③ $S \rightarrow AB / BaA$

$A \rightarrow aA / B$

$B \rightarrow Cb / a$

$C \rightarrow D / \epsilon$

$D \rightarrow eF / f$



⇓

$S \rightarrow AB / BaA$		$S \rightarrow AB / BaA$
$A \rightarrow aA / cb / a$		$A \rightarrow aAb / a / cb$
$B \rightarrow cb / a$	⇒	$B \rightarrow cb / a$
$C \rightarrow f / eF / \epsilon$		$C \rightarrow f / \epsilon$
$D \rightarrow eF / f$ ✗		

ϵ Productions

NULL Productions :- productions of form $A \rightarrow \epsilon$ is called Null productions.

Method:-

Select null productions & delete them with selecting the equivalent derivation.

Ex:- $S \rightarrow AaB$		$S \rightarrow Aa / aB / a / AaB$.
$A \rightarrow b / \epsilon$	⇒	$A \rightarrow b$
$B \rightarrow a / \epsilon$		$B \rightarrow a$

② $S \rightarrow XaY / XbY$ $\quad$ $S \rightarrow XaY / aY / Xa / a$

$XbY / bY / Xb / b.$

$X \rightarrow aY / \epsilon$ $\Rightarrow$

$Y \rightarrow bX / \epsilon$ $\quad$ $X \rightarrow aY / a$

$Y \rightarrow bX / b$

start symbol

③ $S \rightarrow XY$

$X \rightarrow a / \epsilon$

$Y \rightarrow b / \epsilon$

$\left\{ \begin{array}{l} S \rightarrow XY / Y / X / \epsilon \\ X \rightarrow a \\ Y \rightarrow b \end{array} \right.$ → Not valid b/c we r removing ε production so, we take new term.

$S' \rightarrow S / \epsilon$ ←

Order of process of Simplification :-

1. elimination of ε productions.

2. Elimination of Unit productions.

③ Elimination of Useless productions.

C

Choms

1) CN

be in

form.

i.e

Ex :-

Sol^n :-

③ S —

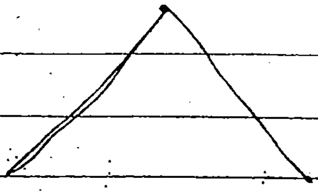
A —

B →

a/a

b/b.

NORMAL FORMS



CNF — Chomsky

GNF — Greighbaik.

Not valid b/c we r ~~removing e~~ production so, we take new term.

1) CNF (Chomsky) :- The grammer G is said to be in CNF if every production is in the form.

$A \rightarrow BC / a$ $A, B, C \in V$

i.e. $a \in T$

Variable ⟶ Variable Variable

Variable ⟶ Terminal

w.:-

Ex :- $S \rightarrow XaY / XbY$

$X \rightarrow b$

$Y \rightarrow a$

Sol^n :- $S \rightarrow X\underbrace{YY}_{A} / X\underbrace{XY}_{B} \Rightarrow S \rightarrow XA / XB$

$A \rightarrow YY$

$B \rightarrow XX$

$X \rightarrow b$

$Y \rightarrow a$

10AS.

② $S \rightarrow Aa / BbA.$

$A \rightarrow aB / b$

$B \rightarrow bB / a.$

A & B derives more than 1 prod. so we take. $X \rightarrow a$

$Y \rightarrow b.$

$S \longrightarrow AX \mid B\overbrace{YA}^{C}$ $\Rightarrow$ $S \longrightarrow AX / BC$

$A \longrightarrow XB \mid b$ $\qquad C \rightarrow YA$

$B \rightarrow YB / a$

$X \longrightarrow a$

$Y \longrightarrow b$

③ $S \longrightarrow aSb / ab$ $\qquad S \rightarrow A\underbrace{SB}_{C} / AB.$

$A \rightarrow a$

$B \rightarrow b.$ $\Rightarrow$

$S \rightarrow AC \mid AB$

$C \rightarrow SB$

$A \rightarrow a$

$B \rightarrow b$

② GNF (GreighBack) :- Grammer is said to be in GNF if production is in the form.

$A \rightarrow a\alpha$ $\quad A \in V$

Var → Terminal $\quad a \in T$

Var → Ter. Var. $\quad \alpha \in V^*$

Var → Ter. Var. Var.

Var → Ter. Var. Var. Var. ......

Ex :- ① $S \rightarrow aAb / abB / ab.$

$A \rightarrow a$

$B \rightarrow b.$

$S \rightarrow aAB / aBB / aB$

$A \rightarrow a$

$B \rightarrow b.$

② $S \rightarrow AbB / bAb$ $\Rightarrow$ $S \rightarrow aBbB / bbB / bAb$

$A \rightarrow aB / b$ $\quad$ $A \rightarrow aB / b$

$B \rightarrow aB / a.$ $\quad$ $B \rightarrow aB / a.$

$\Downarrow$

Ans = $\left\{ \begin{array}{l} S \rightarrow aBYB / bYB / bAY \\ A \rightarrow aB / b \\ B \rightarrow aB / a \\ Y \rightarrow b \end{array} \right\}$

③ $S \rightarrow aSb/ab$.

$S \rightarrow aSB/aB$
$B \rightarrow b$.

④ $L = \{ ww^R / w \in (a,b)^* \}$.

$S \rightarrow aSa/bSb/\epsilon$

$S' \rightarrow S/\epsilon$

$S \rightarrow aSa/bSb/aa/bb$

$$\left.\begin{array}{l} S \rightarrow aSA/bSB/aA/bB \\ A \rightarrow a \\ B \rightarrow b. \end{array}\right\} \text{GNF}$$

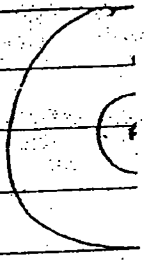
NOTE:-

1. Grammer with out ε-rule Can be Converted into CNF or GNF.

2. To generate a string of length n in CNF 2n-1 productions are required.

3. In GNF n productions are req. to generate string of length N.

1. Emptyn
2. Finiten
3. Membe



We can

Emptynes

ε

Symbo
then
Langu
Oth

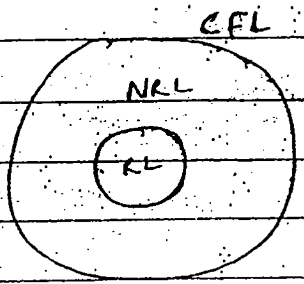
Ex:① S →
A →

A –

L = ∅
empty

Decision Properties

1. Emptyness
2. Finiteness
3. Membership



Decidable	Undecideable
① Emptyness	① Ambiguity
② Non emptyness	② Equality
③ Finite/Infinite	③ Unequality
④ Membership	④ Regularity/Non Regular
	⑤ Univercality

We can't predict from CFL than it is regular/Non Regular.

Emptyness :- After reducing the CFG if it Contains atleast One Symbol derived from start symbol of grammer X derives any terminal then the grammer Generates Non Empty language.
Otherwise it generates empty language.

Ex ①	②	③
$S \to aAB / bB$	$S \to aA / bB$	$S \to aAB / bB / a$
$A \to aB / b$	$A \to aA / b$	$A \to aB$
	$S \to aA$ }	$B \to aA$
$A \to a$	$A \to b$	
		$S \to a$
$L = \emptyset$ empty	L = Non empty	generated Non empty Lang.

2. **Finiteness** :- ① Convert grammer into CNF.

② Draw variable dependency Graph.

③ If VDGraph Contain loops or cycles then it generates ∞ language.

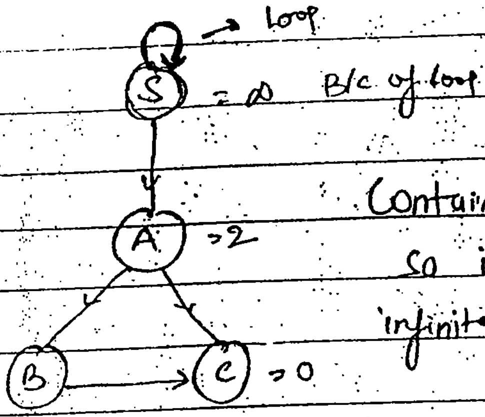
④ If VDGraph No. loops or cycles then it generate finite Language.

Ex :- $S \to SS/A$

$A \to BC/a$

$B \to CC/b$.

VDG :-

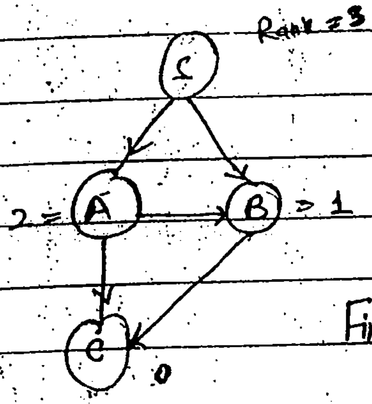


Contain loop so it generate infinite language.

② $S \to AB$

$A \to BC/a$

$B \to CC/b$



Finite Language.

---

③ S-

A-

B-

C-

NOTE :

1 :-

Va

② R

③ Gr

R

3. Me

② I

③ CYK

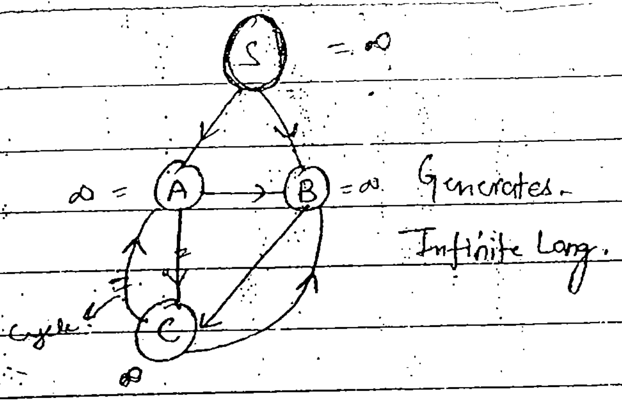
④ B.

③ $S \rightarrow AB$

$A \rightarrow BC / a$

$B \rightarrow CC / b$

$C \rightarrow AB.$



Generates Infinite Lang.

NOTE :-

1 :- In VDG we can define rank for every variable.

② Rank (A) = length of longest path from A to last variable.

③ Grammer G generates finite Language if Rank of each variable is finite otherwise if any atleast one variable has Rank = ∞ the the Grammer is Infinite.

3. **Membership** :- ① used to verify abitrary string generated by Gram or Not.

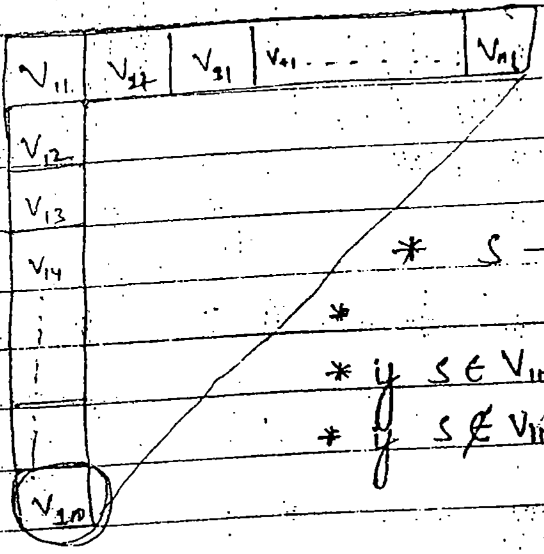
Cokes Young Kasami → SUN microsys.

② Implemented by GYK Algo.

③ CYK algo. is implemented on CFG if it is CNF.

④ Best example for Dynamic Programming.

$|w| = n$



* $S \rightarrow$ start Symbol.

*

* if $S \in V_{1N} \Rightarrow W \in L(G)$

* if $S \notin V_{1N} \Rightarrow W \notin L(G)$.

Ex:- $S \longrightarrow AB$

$A \longrightarrow BA/b$

$B \longrightarrow BB/a$.

$w = ba$

b	a
A	B
$\boxed{S}$	

derives b ← A

B → which derives a

$\boxed{S}$ → it is also start sym. so $W \in L(G)$.

↑ which derives AB

② $w = ab$

a	b
B	A
A	

$\therefore ab \notin L(G)$

③ $W = aba$

a	b	a
B	A	B
A	S	
$\boxed{S}$		

$W \in L(G)$.

④ W =

b
A
Ø
Ø

⑥ W =

a
B
A
Ø
Ø

⑧ W

a
B
A
S
S

art Symbol.

w ∈ L(G)

w ∉ L(G).

④ w = bba

b	b	a
A	A	B
∅	S	
∅		

w ∉ L(G)

⑤ w = aab

a	a	b
B	B	A
B	A	
A		

w ∉ L(G).

⑥ w = abba.

a	b	b	a
B	A	A	B
A	∅	S	
∅	∅		
∅			

⑦ w = babb.

b	a	b	b
A	B	A	A
S	A	∅	
∅	∅		
∅			

w ∉ L(G).

∉ L(G).

⑧ w = abaa.

a	b	a	a
B	A	B	B
A	S	B	
S	S		
S			

w ∈ L(G).

w = aabaa.

a	a	b	a	a
B	B	A	B	B
B	A	S	B	
A	S	S		
S	S			
S				

w ∈ L(G).

,a

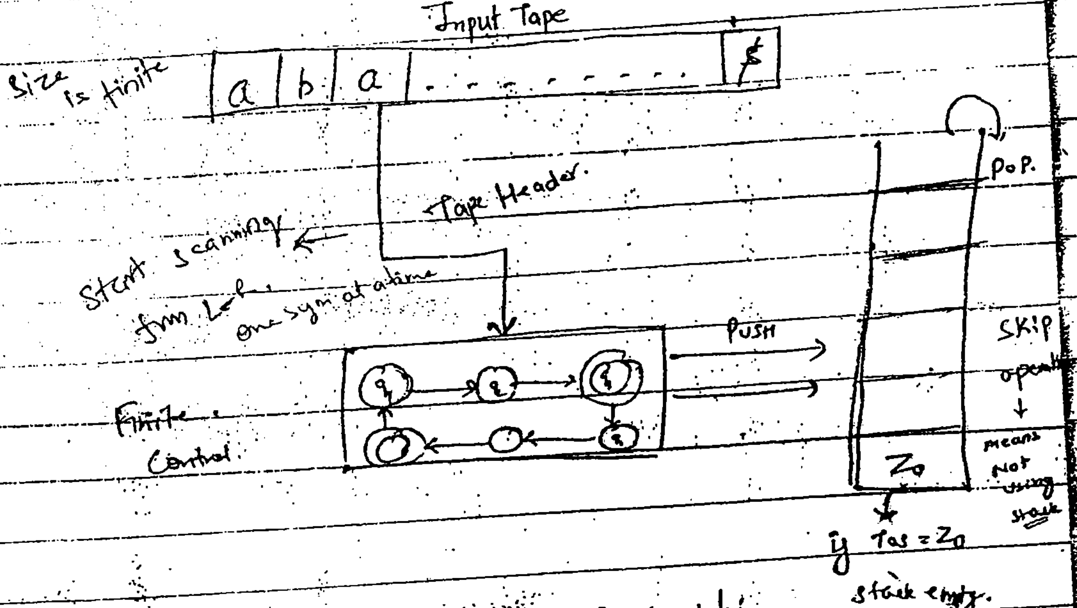
a

B

L(G).

11/11/10

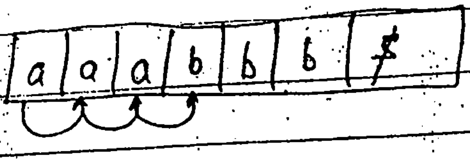
PUSH DOWN AUTOMATA (More powerful than FA)

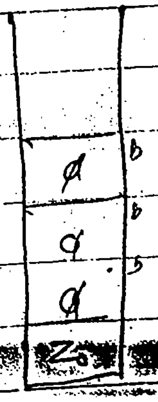


Pehle F.A Banao agar nhi Banti hai to phr PDA use kro kynki PDA is more costlier.

Ex :- $L = \{ a^n b^n \mid n \geq 1 \}$.

ab, aabb, aaabbb ---------





When b encounter we pop a from stack if at end of string we reach to Z0 the string is accepted.
i.e. Z0 must be top element.

top
(a, Z0)
(a, x)
(b, a)

Q
δ (q0

Definatio

Mathem

PDA i

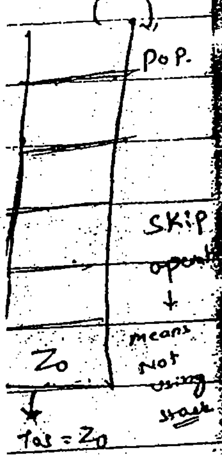
Q →
Σ →
Γ →
Z0 →
q0 →
δ →
F →

More powerful than FA.

$(a, \overset{\overset{\text{top}}{\uparrow}}{z_0}) \rightarrow$ push a $\rightarrow az_0$

$(a, a) \rightarrow$ push a $\rightarrow aa$

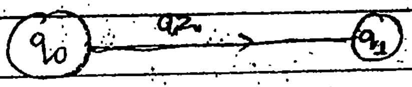
$(b, a) \rightarrow$ pop a $\rightarrow \epsilon$



stack empty.

Kyuki

$$\delta(\overset{Q}{\overset{\uparrow}{q_0}}, \overset{\Sigma}{\overset{\uparrow}{a}}, \overset{\overset{\text{tau } \{\text{stack symbol}\}}{\Gamma}}{\overset{\uparrow}{z_0}}) = (\overset{Q}{\overset{\uparrow}{q_1}}, \overset{\Gamma^*}{\overset{\uparrow}{az_0}})$$



Defination :- FA with memory is Called P.D.A.

OR

Mathematical Representation of CFL is cld PDA.

OR

PDA is a 7 ~~type~~ Tuple $M = \{Q, \Sigma, \Gamma, z_0, \delta, q_0, F\}$

$Q \rightarrow$ set of all states

$\Sigma \rightarrow$ Input alphabets

$\Gamma \rightarrow$ set of stack symbols

$z_0 \rightarrow$ Topmost symbol of stack

$q_0 \rightarrow$ Initial state

$\delta \rightarrow$ Transition $f^n \leftarrow Q \times \Sigma \times F \rightarrow Q \times \Gamma^*$

$F \rightarrow$ set of all final state.

re pop a.

end of

to Z0

cepted.

top

) Purpose of symbol Z_0 is to see that wheather stack is empty or Not.

) PDA Can recognize data through stack symbols.

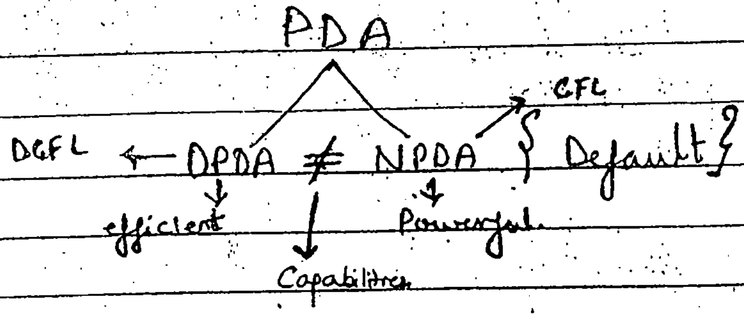
) PDA uses stack as external storage location.

language which is accepted by PDA is CFL.

It can accept all lang accepted by FA.

PDA accept some of lang which are not accepted by FA.

)



NPDA is more powerful the DPDA.

8) Every CFL is accepted by NPDA.

9) For every CFL we can't Construct DPDA.

10) Every lang is accepted by DPDA is also accepted

that
Not.
state
storage
A is CFL.
by FA.
are not
}
DPDA.
NPDA.
nstruct DPDA.
also accepted

by NPDA But Converse need Not be True.

$\Rightarrow L_D \subset L_N$

L_D = Set of all languange accepted by DPDA.

L_N = Set of all lang. accepted by NPDA.

1) Every DCFL is CFL But Converse is Not True.

→ for every processing symbol or I/P symbol at a state based on TopMost symbol of stack. If it perform PUSH, POP, skip operations then stack being used by PDA.

→ for every processing symbol if Topmost symbol of stack (Z_0) remain same this means that stack is not used by PDA. Hence PDA is equivalent to FA.

Instentaneous Description :- (ID)

It describe next move of PDA. i.e. next move of PDA depends on 3 entities.

1. Current state.
2. Current processing Symbol.
3. " Top Most Symbol of stack.

$$\delta(\underset{\substack{\checkmark\\ \text{Current state}}}{q_i}, \underset{\substack{\downarrow\\ \text{Current}\\ \text{processing}\\ \text{Symbol}}}{a}, \underset{\substack{\downarrow\\ \text{Current}\\ \text{Top Most}\\ \text{symbol of}\\ \text{stack}}}{Z_0}) = (\underset{\substack{\downarrow\\ \text{New}\\ \text{state}}}{q_j}, \underset{\substack{\text{New top most}\\ \text{symbol of}\\ \text{stack}}}{aZ_0}).$$

Acceptance by PDA :- PDA accept string in 2 ways.

1. Acceptance by empty stack.
2. Acceptance by final state.

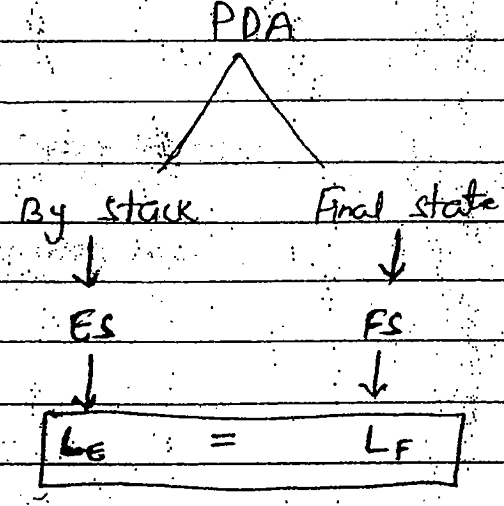
① By Stack :-
After Reading entire I/P stack is empty then the I/P is accepted by PDA & this process
... A.B.E. stack.

② By final state :- After reading all I/P by PDA reaches final state than I/P is accepted by PDA & this mechanism is K/n as acceptance by F.S.

. nat.

ntitres.

NOTE :-



ule.

stop most value of stack

L_E = Set of all languages accepted by PDA with ES Mechanisms.

t string in

L_F =

Operations in PDA :-

1. PUSH ② POP ③ SKIP.

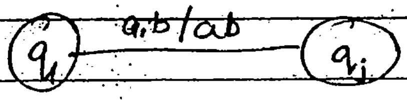
$$\delta(\underset{ps}{q_i}, a, \underset{TS}{b}) = \{q_j, \underset{TS}{ab}\}$$

e I/P

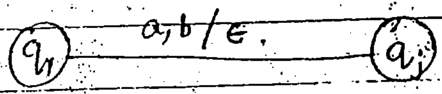
I/P is

process

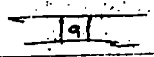
Processing symbol

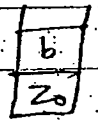


② POP = $\delta(q_i, a,b) = (q_j, \epsilon)$



③ Skip :-





$\delta(q_i, a,b) = (q_j, b)$.

No change in Top most Seymbol stack.

Pumping Lemma for CFL :- If L is any CFL & $\in L$ $|z| \geq n$

f $uv^iwx^iy \in L$ $\forall i \geq 1$

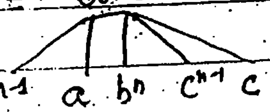
then L is non CFL.

1) Pumming L is used to prove that some of Language is Non CFL. for P.L input is non CFL & O/P is also Non CFL.

ex :- ① $L = \{a^n b^n c^n / n \geq 1\}$

Let $z \in L$

$\Rightarrow z = a^n b^n c^n$



from

$a^{n-1} a^i b^n c^n$ let i=2

$\Rightarrow a^{n+1} b^n c^n$ $= a^n \neq b^n = c^n$

so language is Non CFL.

② $L = \{ww / w \in \{a,b\}^*\}$. NCFL

③ $L = \{www / w \in \{a,b\}^*\}$ *

4) L =

5) L = {

NOTE :-

If L is

Such

L are

Other

Ex :- {

0, 2, 4, 6,

② {a

1, 4, 7, 10

③ L =

NC

2, 4, 8

④ L = {

NC

1, 4,

⑤ L = {

⑥ L =

⑦ L = { a

4) $L = \{ww w^R / w \in \{a,b\}^*\}$ NCFL

5) $L = \{ \underbrace{a^m b^n}_{\omega} \underbrace{a^m b^n}_{\omega} / m,n \geq 1 \}$ NCFL

NOTE :-

i.e. Contains only 1 alphabet. ↑

If L is any lang. defined over alphabet $\Sigma = \{a\}$ such that length of all strings of Language L are in A.P. then the language L is CFL.
Other wise Non CFL.

Ex :- $\{ a^{2n} / n \geq 0$

0, 2, 4, 6, 8 ⇒ CFL

② $\{a^{3n+1} / n \geq 0\}$

1, 4, 7, 10 ⇒ CFL

③ $L = \{ a^{2^n} / n \geq 1$

NCFL

2, 4, 8 . . . .

④ $L = \{ a^{n^2} / n \geq 1 \}$

NCFL

1, 4, 9, 16 . . . . .

⑤ $L = \{ a^{n!} / n \geq 0 \}$ → 1, 2, 6, 24 . . .

NCFL

⑥ $L = \{ a^p / p \text{ is prime} \}$ 2, 3, 5, 7, 11 . . . . .

NCFL

TURING M/c

The mathematical rep. of recursive Enumerated (RE) lang is called T.M/c.

OR

T.M. is a 7 Tuple $M = \{Q, \Sigma, \Gamma, B, \delta, q_0, F\}$.

Q → set of all states

Σ → set of I/P sym.

Γ → set of all tuple f^n.

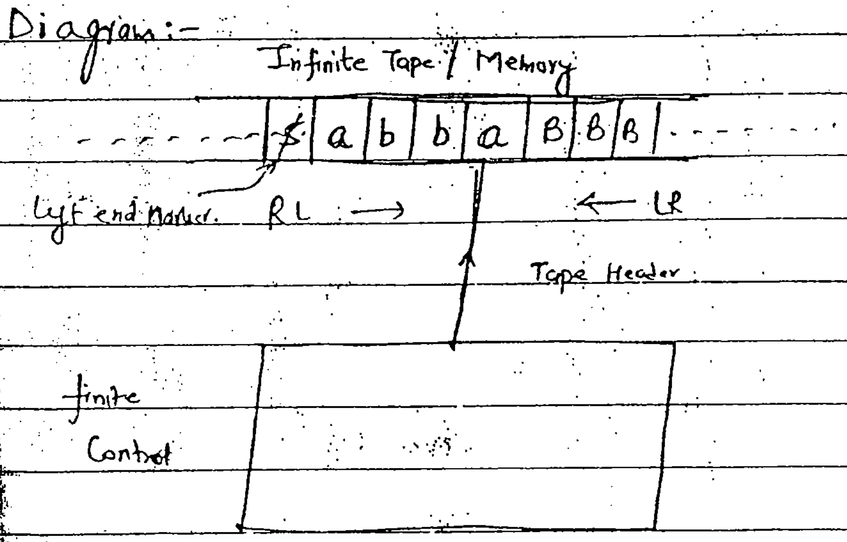
B → Blank symbol.

δ → $Q \times F \rightarrow Q \times \Gamma \times \{L, R\}$ → is transition f^n.

q_0 → Initial state

F → Set of final states.

Diagram:-



T·M/c has 3 Components:

① Infinite Tape :- divided into Cell & each cell is capable of holding one Symbol. It is 2 way ∞ tape & Can be Converted into 1 way ∞ tape by adding some Left end Marker ($)

● The Blank ~~Symbol~~ ([illegible]) Cells of tape are filled by B. Symbol (B)

② Tape Header :- It is read the data from ∞ tape starting from leftmost end of the string. After reading I/P symbol from I/P tape the tape Header can move in both the direction i.e. (L & R).

The movement of Tape header is Bidirectional. B/c of this movement of Tape Header Turing M/c work as Reader & well as writer.

③ Finite Control :- It takes care of movement of T·M/c. from one state to another state.

NOTE :-

1. F.

Ca

2. Turing

Sy

3. Turin

a) lan

b)

c) Tr

4. Turin

5. Any

6. Tur

a) Re

b)

7. T·M

F

8. T·M

ar

NOTE :-

1. F.A. with memory, & with reading & writing Capability is Turing M/c.

2. Turing M/c is abstract model of Real Computer Sys.

3. Turing M/c works as.
   a) Language Acceptor
   b) " Generator.
   c) Transducer.

4. Turing M/c through tape symbol.

5. Any language which is accepted by T.M. is Called as T.M. recognizable language.

6. Turing M/c recognizable language are of 2 types.

No order.

   a) Recursive Enumerated Langage (REL) → Procedure
   b) Recursive Language → Algo.

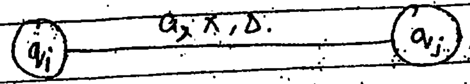
7. T.M/c recognized all Lang accepted by FA & PDA.

8. T.M/c also accepts Some of language which are not accepted by P.D.A.

9. Expressive power of T.M & Computer is same.

Instantaneous Description:- ID describes the next move of T.M. The next move of T.M depends on 2 entities.

$$\delta(\underset{\text{state}}{q_i}, \underset{\text{processing sym}}{a}) = (q_j, x, D).$$



Halt:- A state where transition is not defined is called halt.

Acceptance By T.M:-

After taking the input string the T.M. has 3 possibilities

① May go to final Halt.

② May go to Nonfinal Halt.

③ May go to a Loop.

final Halt

→ After fi by

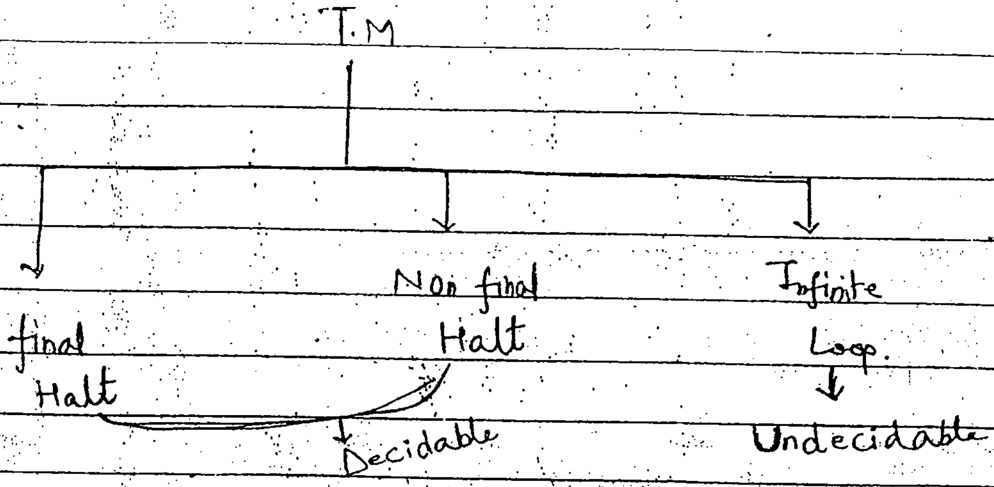
→ Afte

→ Af. R y R

IN is Same.

he

depends



→ After reading the entire I/P if T.M. reaches final Halt then the I/P is accepted by T.M.

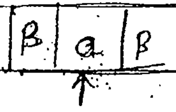
→ After reading entire I/P if T.M. reaches Non final Halt then I/P is rejected by T.M/c.

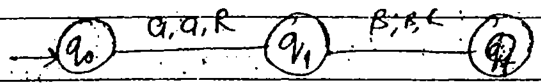
not

→ Af. R. a part ~~of~~ or Complete I/P string if T.M. goes to a loop, then the I/P string is neither accepted nor Rejected.

T.M. has

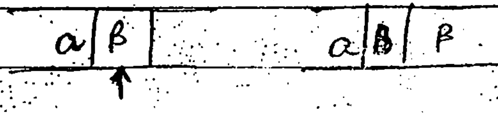
Q:- $L = \{a\}$

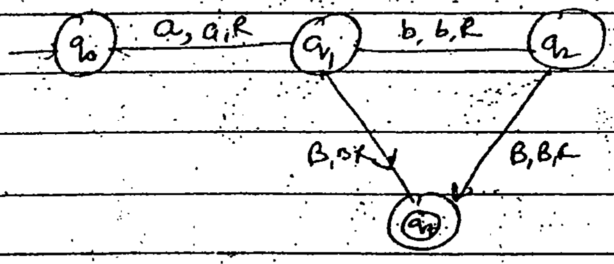




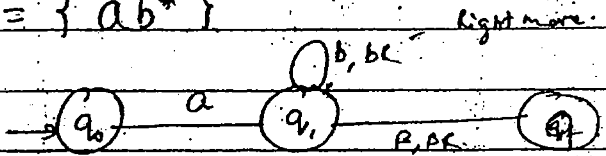
	a	β
q0	q1,aR	Halt
q1	Halt	β,BR
qf	Halt	Halt.

Q:- $L = \{a, ab\}$.





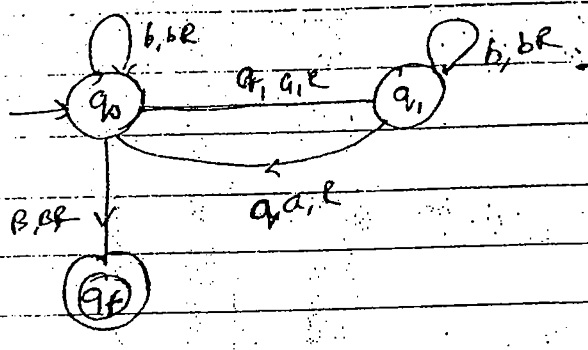
Q:- $L = \{ab^*\}$



	a	b	β
q0	q1,aR	Halt	Halt
q1	Halt	q1,BR	qf,BR
qf	Halt	Halt	Halt.

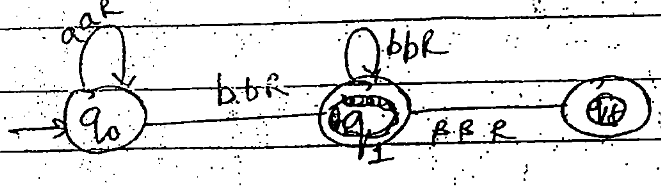
Q :- $L = \{ \omega \in (a,b)^* \;/\; |\omega|_a = 0 \bmod 2 \}$

$|\omega|_a$ = even.

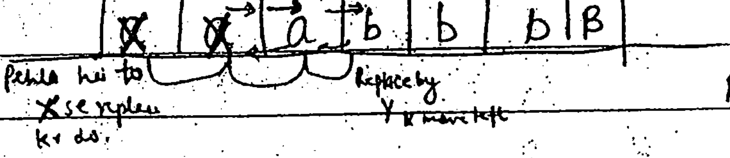


	a	b	B
q_0	q_1,aR	q_0bR	q_fBL
q_1	q_0aR	q_1,bR	H
q_f	Halt	Halt	H.

Q :- Construct T.M. for language $L = \{ a^m b^n / {m \geq 0 \atop n \geq 1} \}$

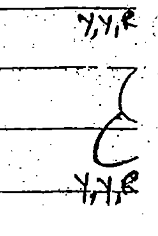


Q :- Cons. T.M for the language $L = \{ a^m b^n / m, n \geq 1 \}$ m = n



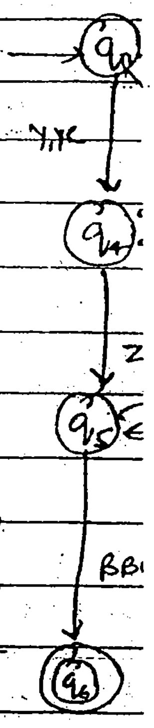
Aur jaise hi X aaye to Us state ko Initial se join kr do.

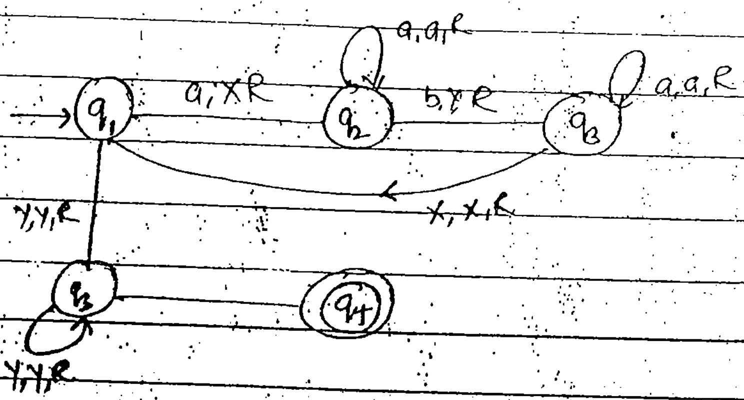
$\Gamma\,(X, Y, a, b, B)$



Q :-

F =



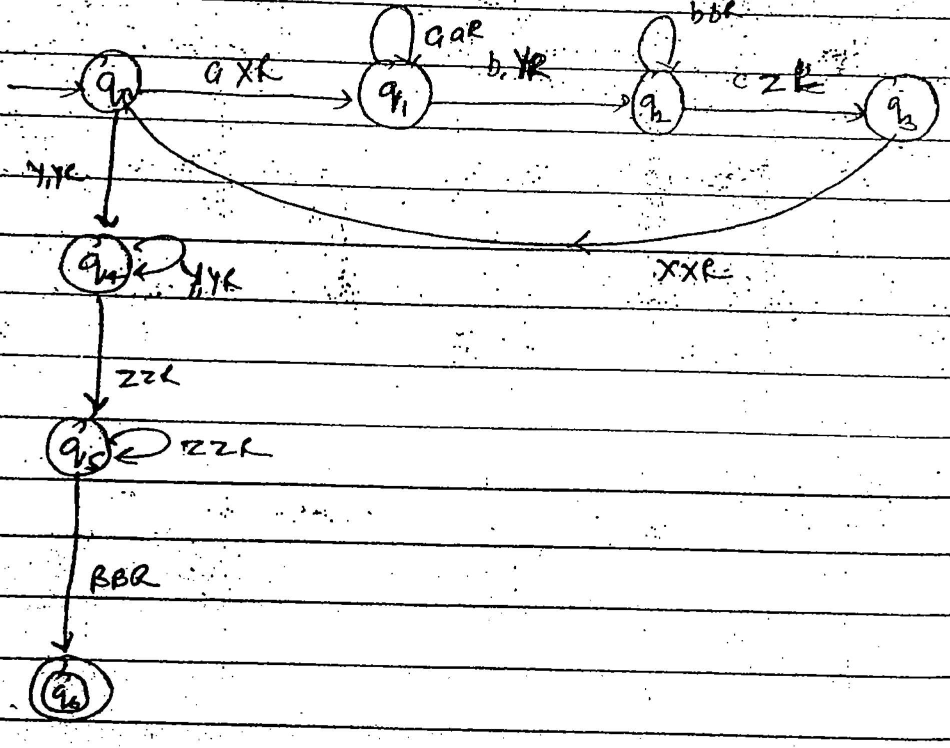


b	β
q0 bR	qf BL
q1 bR	H
Halt	H.

Q:- $L = \{a^n b^n c^n / n \geq 1\}$.

$\Gamma = \{a, b, c, X, Y, Z, \beta\}$.

$a^m b^n / m > 0, n \geq 1\}$



$\{a^m b^n / m, n \geq 1\}$ $m \geq n$

Aur jaise hi aaye to us state ko ~~Initial se join~~ kr do.

Q:- Cons. 2 way infinite TAPE T.M. defined as follows:-

δ	0	1	β
δ			
q_0	q_0 0R	q_2 1L	q_f BR
q_1	q_2 1L	-	q_f BR
q_2	q_2 1,L	q_2,0,L	q_2,BR
q_3	Halt	Halt	Halt

a) Doesnt accept 0000

b) Loop on 01

c) accepts all string ending with 1

d) doesnot accept any string.

Q:- Cons. T.M. as follows:-

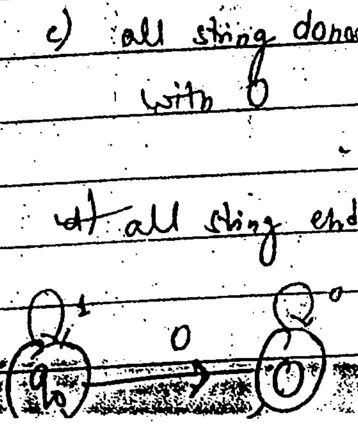
δ	0	1	β
q_0	q_1 0R	q_0,1,R	Halt
q_1	q_1 0R	q_1,1,R	q_f β,R
q_2	Halt	Halt	Halt

a) all string end with 00

b) end with 10

c) all string donot end with 0

d) all string end with 0.



Ueux:-

nt accept
00

p on 01

s all string
ig with 1

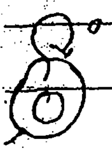
t accept
ing.

ing ends with 00

ts 10

g donot end

ing end with 0



Gate 2008

Q:- Cons T.M. defined as follows.

lang accepted by m/c is

$\delta(q_0, 0) = (q_0, B, R)$

$\delta(q_0, 1) = (q_1, B, R)$ ①

$\delta(q_1, 1) = (q_1, B, R)$ a)

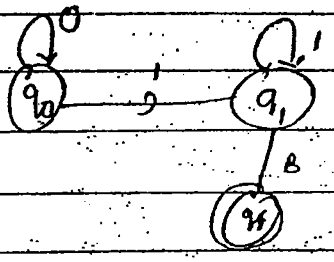
$\delta(q_1, B) = (q_f, B, R)$ b) $0^m 1^n / m \geq 0 \; n \geq 1$

c)

d)

② No. of state in min FA

ⓐ 2
ⓑ 3
c 4
ⓓ 5



Recursive Set & Recursive enum set.

A language accepted by T.M is cld T.Mc recognizable language.

T.M. reconizable lang. can be classified into 2 types ?

① Recursive sets
② Recursive Enum sets.

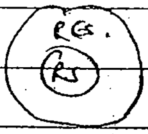
① **Recursive sets** :- language accepted by T.M for which membership prop is decidable is cld Recursive language / sets

② **R. Enum sets** :- Language accepted by T.m for which membership Prop is Undecidable is cld RE sets.

① $L = \{ a^n b^n c^n \mid n \geq 0 \} = RS.$
$L = \{ a^m b^n c^m d^n / m,n \geq 0 \}$ RSets
$L = \{ a^{n^2} / n \geq 0 \} - RS.$
sell of all C^{++} prog} REL

every Recursive set is RE set. But every RES need not to be RS.

RS ⊂ REs



Closure Prop. of RS:-

① Closed	Not Closed
∪	Kleen Closure
∩	Homomorphism
Concatenation	Substitution
Inverse Homomorphism.	Quotient with Regular Lang.
Compliment.	
Reverse operator	
Intersection with Regular L.	

Closure Prop of REs.

Closed

① Union

Concatenation

Kleen Closure.

Intersection

Substitution

Homomorphism

Inverse Homo.

Reverse operator

∩ with Regular set

Quotient with "

---

n set.

Cld 7 Mk

sujied noto

cepted by

ship prep

Recursive

pted by

bership

s cld

Rs.

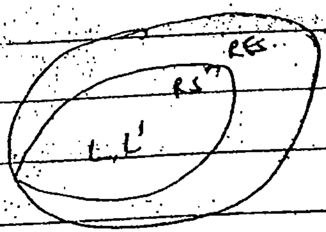
Rsets

eg} REL

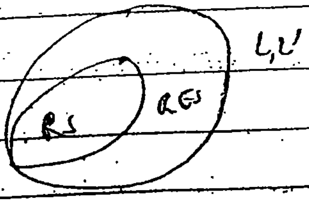
① * Recursive Enum is not closed wrt. Complement.

② L, L' are RE Set than L must be recursive language.

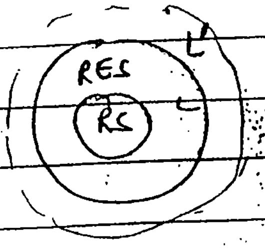
③ for Language L, L' following 3 possibility exist



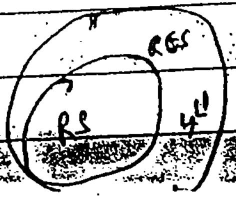
* L & L' are not R. Enum sets



* L is RES But not recursive & L' is not RES



The fallowing Case doesn't arises.



Linear

with ..

2

Tape

Rig

Content

accy

Ex :-

Complement,

Linear Bounded Automata :-

A non Deterministic T.M.

recursive

with. The following properties is Cld LBA. Includes 2 special symbol \$ & ¢.

sibility exists

Tape header Can't move fm Left fm \$ & Right fm ¢. :.

Left Hand Marker

$$M = (Q, \Sigma\, \delta, B\, \$, ¢, \Gamma, q_0, F).$$

Right end Mark.

$Q \rightarrow$ --

Context Sensitive Lang :- Language Gen. by CSG or accepted by LBA is Called CSL.

L' is not re

Ex :- $L = \{a^n b^n c^n \mid n \geq 1\}$ CSL

$\{a^n b^n c^{2n} / n \geq 1\}$ CSL

$\{a^m b^n a^m b^n / m \geq 1, n \geq 1\}$ CSL

$\{a^{n^2}\}_{n \geq 1}$ CSL

$\{a^p / p \text{ is prime}\}$ CSL

$\{ww / w \in$

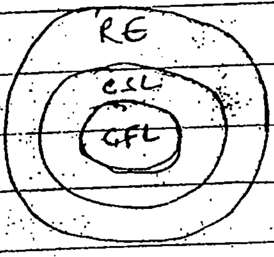
L.

Note:

1) CSL is free from ϵ.

② Every CFL (without ϵ) is CSL But every CSL need Not to be CFL.

③ Every CSL is a Recursive set / R. Enu set. Vice versa is Not TRUE.



④ LBA is mathematical representation of CSL.

Closure Properties of CSL:- CSL is Closed w.r.t. to following operations.

① Union ⑦ ϵ free Homomorphism.
② Concatenation ⑧ Inverse Homo
③ +ve Closure ⑨ Intersection with regular sets.
④ Intersection
⑤ Substitution *CSL Not Closed w.r.t to Kleen Closure
$L \to CSL$
⑥ Reversal

If L

→ Almo

Undecid

exist

D

C

(P) Easy

ive By. ↓

DTM

Ex:→ T

or

① Ho

② E

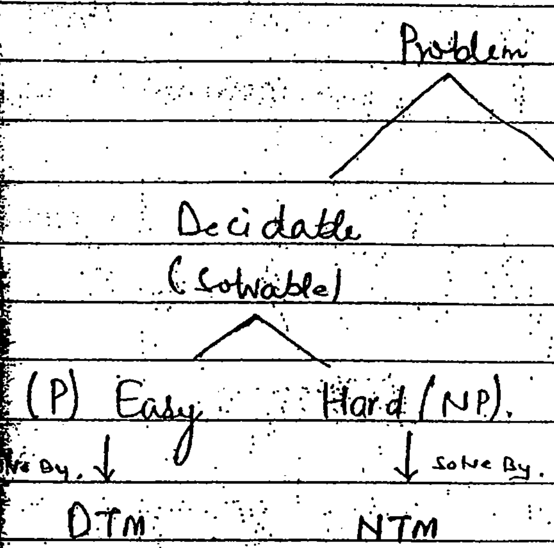
The

① E

→ If L is CSL L' - ? Undecidable.

→ Almost all properties of LBA are undecidable

Undecidability :- The problem for which No algo. exist to get the solⁿ is called Undecidable.



Ex :→ There is no such algo by which we conclude that Grammer is Ambiguous or Not.

① Halting of TM. is Undecidable.
② Equality.

The following prop. of TM (REL) is Undecidable

① Emptyness ② Non emptyness ③ Finiteness.

But every
L.
Ence cut.
of CSL.
losed w.r.t.
gular sets.
Kleen
Closure.

4) Infinite Ness. ⑦ Context Freedom
5) Regularity ⑧ " Dependency.
6) Recursive ness ⑨ Recursive Enum.

Domino Game

Post Corrspondance Problem (PCP) :-

Let $X = \{x_1, x_2, x_3 \dots x_m\}$
$Y = \{y_1, y_2 \dots y_n\}$ are ordered set of strings then the process of finding integers $i_1, i_2, i_3 \dots i_k$

$$X_{i_1}, X_{i_2}, X_{i_3} \dots X_{i_k} = Y_{i_1}, Y_{i_2}, Y_{i_3} \dots Y_{i_k},$$

is Called PCB.

where
$i_1, i_2, i_3 \dots i_k$ is soln of PCB.

NOTE:-
PCB is Undecidable.

Ex:-	X	Y
1	bbab	a
2	ab	abbb
3	baa	aa
4	b	bb
5		

(2,1

a

Ex :-

1

2

3

length=

Lin

RICE

set

$(2,1,4,3) \longrightarrow$ solⁿ.

$x_2 x_1 x_4 x_3 = y_2 y_1 y_4 y_3$

abbbabbbaa = abbbabbbaa

rdered

of

$\ldots Y_{ik}$,

Ex :-

	X	Y
1	aab	aba
2	bb	bba
3	aaaa	b
length=	9	7

Length are Not equal so No solⁿ.

RICE's :- The non Trivial property of R Enum set is Undecidable.

Type 0 / Unrestricted Grammers :- Turing M/c

$\underset{\text{Var/Ter}}{\alpha} \to \underset{\text{Var/Ter}}{\beta}$ No restriction on either side of productions.

Atleast one Non Terminal on L.H.s.

$aA \to abCB$ $bA \to a$

$aA \to bAA$ $S \to aAb \mid \epsilon$

2) Type 1 / Contexte Sensitive Grammer :- LBA

$\underset{\text{Var/Ter}}{\alpha} \longrightarrow \underset{\text{Var/Ter}}{\beta}$ $|\alpha| \le |\beta|$

Rules of form.

α, β may be empty But γ must be Non empty.

$$\alpha \underset{\text{Var}}{A} \beta \longrightarrow \alpha \underbrace{\gamma}_{\text{string of Ter/Non Ter.}} \beta$$

α: String of Ter/NT

$S \longrightarrow \epsilon$ is included in it.

3) Type 2 / Context Free Grammer :- PDA

$\underset{\text{Non Terminal}}{\alpha} \longrightarrow \beta \to$ Terminal/N.T.

$|\alpha| = 1$

4) Type 3 / Regular Grammer :-

Single Non-Terminal on L.H.S.

RHS :- Single Terminal or String of terminals with Single N.T. on left or Right end.