

THEORY OF COMPUTATION

Date / /

Page

10/10
bills on 10
Student Notebooks

7-12 Ques.

F.A. & R.L. - 60%

PDA & CFL - 20%

LBA & CSL - 0%

TM & REL - 2%

Undecidability } 20%
Complexity Theory }

TOC:

1. It is the mathematical study of computing m/c's and its capabilities, means observation before go for practical.

2. It is the study of F.L. & Automata theory.

F.L.:

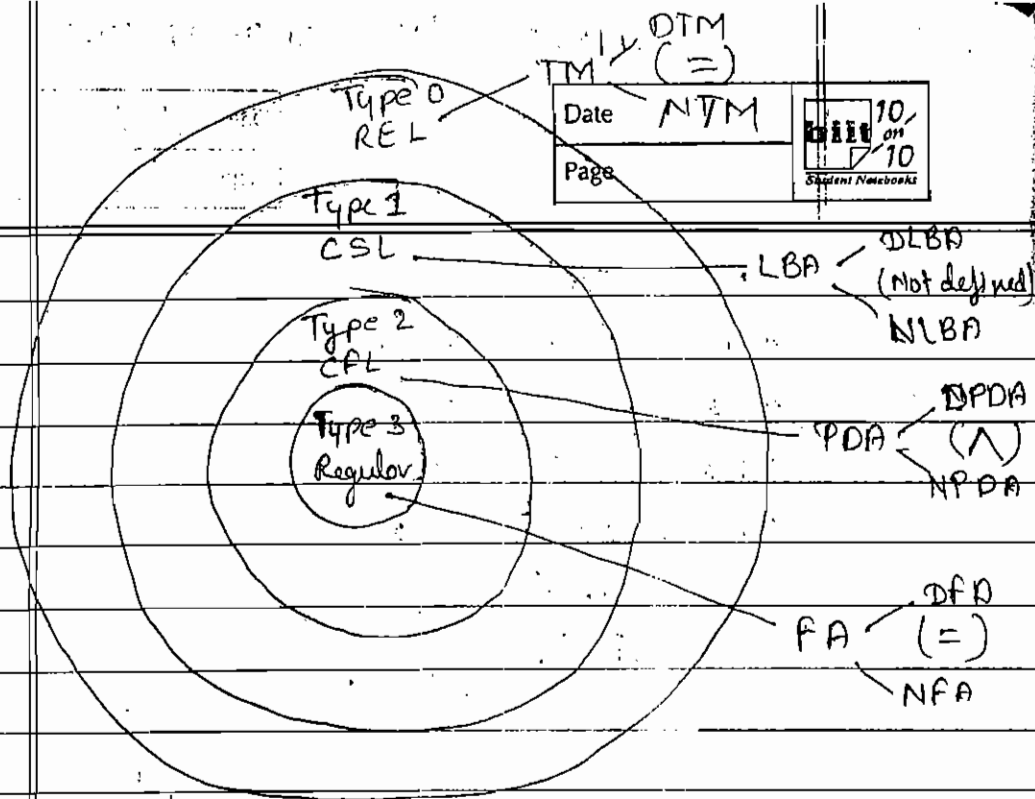
Collection of strings where these strings are formed based on some conditions.

Type 3 FL $\rightarrow$ Regular Lang. : FA.

Type 2 FL $\rightarrow$ Context Free Lang. : PDA.

Type 1 FL $\rightarrow$ Context Sensitive L. : Linear Bounded a.

Type 0 FL $\rightarrow$ Recursive Enumerable : T.M.



Expressive Power:

No. of lang. accepted by a particular M/C.

$$\text{FA}^{(1)} < \text{PDA}^{(2)} < \text{LBA}^{(3)} < \text{TM}^{(4)}$$

M/C accepts the lang. in two ways - Deterministic & Non-D (more than 1 solⁿ).

E.P. of DLBA & NLBA is undecidable.

$$\Rightarrow L_1 = \{a^n \cdot b^m \mid n, m \geq 1\}$$

$$L_2 = \{a^n b^n \mid n \geq 1\}$$

$$L_3 = \{a^n b^n c^n \mid n \geq 1\}$$

$$L_4 = \{a^n b^n c^n d^n \mid n \geq 1\}$$

DLBA
(Not defined)
NLBA

PDA $\left\{ \begin{array}{l} DPDA \\ (\wedge) \\ NPDA \end{array} \right.$

DFD
(=)
NFA

FA fails to recognize lang. L_2 becoz L_2 requires memory element (comparison b/w symbols of lang.).

Hence to recognize lang. L_2 the suitable M/C is PDA.

PDA fails to recognize lang. which requires >1 memory element (>1 comparison).

To recognize such kind of lang. suitable M/C is TM.

TM capable of recognizing all F.L.

Alphabet (Σ): Any finite non-empty set of symbols.

String: Any finite sequence of symbols over the given alphabet.

Prefix of String: Any seq. of leading symbols over the given alphabet is known as POS.

For string of length n , POS is $(n+1)$.

TOC $\Rightarrow E, T, TO, TOC$

Suffix of String: Any seq. of trailing symbols over the given string alphabet.

TOC $\Rightarrow E, C, OC, TOC$

s- Deterministic

1.

Substrings: Any consecutive seq. of symbols over the given string.

TOC $\Rightarrow$ T, O, C, TO, OC, TOC, ϵ

$$\text{No. of substrings} = \frac{n(n+1)}{2} + 1$$

Language: Any set of strings over the given alphabet.

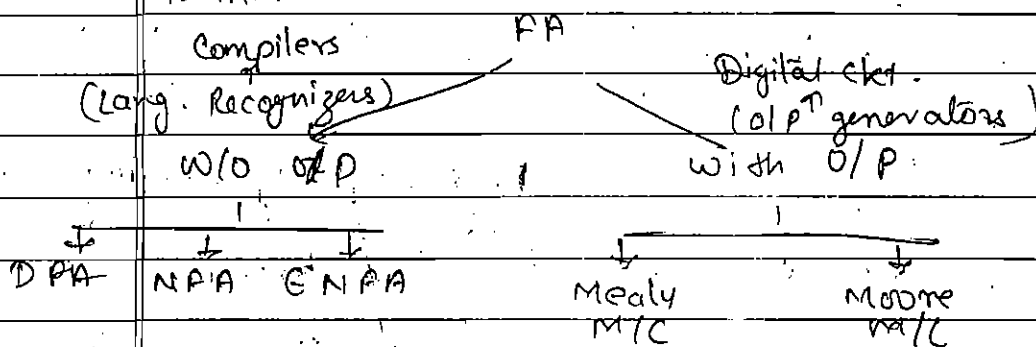
$L = \{ \epsilon \}$ - finite lang.

Grammar: Set of rules used to describe string of a lang.

$$G = (V, T, P, S)$$

Finite Automata:

Mathematical model which contains finite no. of states and transitions defined b/w them.



Representation or Notation of F.A.

X-tional diagrams

Transitional diagrams
Table

DFA!

ymbols

It is FA in which from each & every state on every I/P symbol exactly one transition should exist.

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

Q : Finite no. of states

Σ : I/P alphabet

δ : X-tion funⁿ.

q_0 : initial state

F : set of final states

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

the

scribe

Acceptance Mechanism!

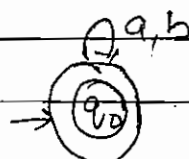
By reading string x , from left to right if DFA enters into final state, given string is accepted, otherwise not accepted. If there is no final state in DFA, empty lang. will be accepted by DFA.

ains

el bld

Ques!

Minimal DFA accepts all strings of a 's & b 's including ϵ .



ns)

7.

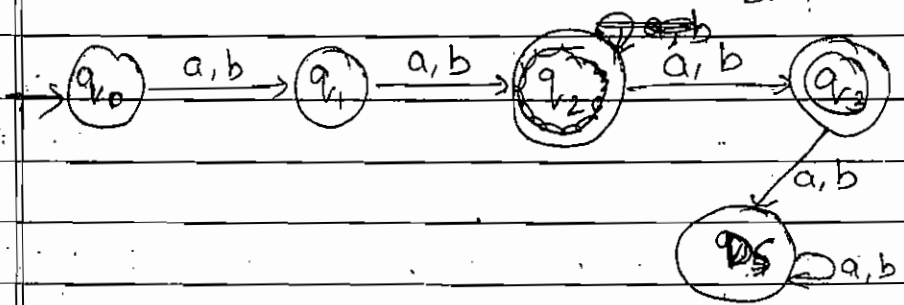
ble

LENGTH OF STRING!

→ Construct minimal DFA that accepts all strings of A's & B's where ^{length of} each string is exactly 3.

$2 \times 2 \times 2 = 8$ possibilities. $\Sigma = (a, b)$

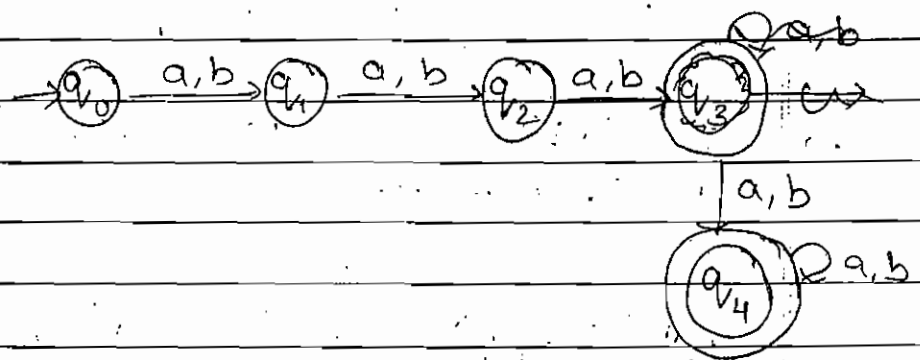
$L = \{aab, aba, abb, bab, aab, bbb\}$



NO. OF STATES!

⇒ The minimal DFA that accepts all strings of A's & B's where length of string is exactly n , requires $(n+2)$ no. of states.

→ $\Sigma = (a, b)$ length of string is at least 4.
 $L = \{aaaa, bbbb, \dots\}$

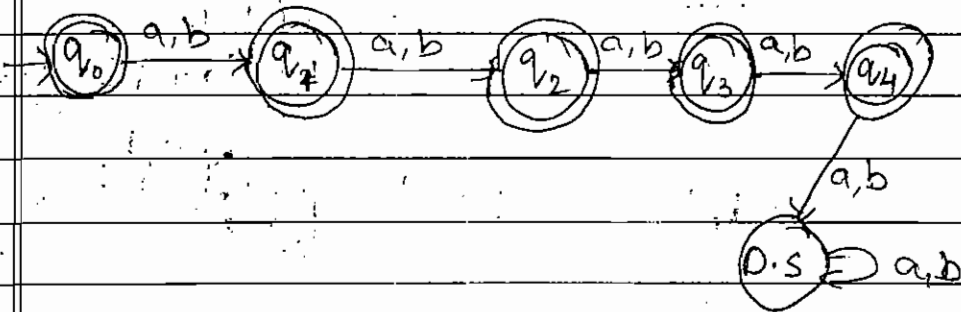


⇒ The minimal DFA that accepts all strings of a's & b's where length of string is at least n , req. $(n+1)$ no. of states.

→ $\Sigma = (a, b)$

$L = \{0, 1, 2, 3, 4\}$

length of string is
at most 4.



⇒

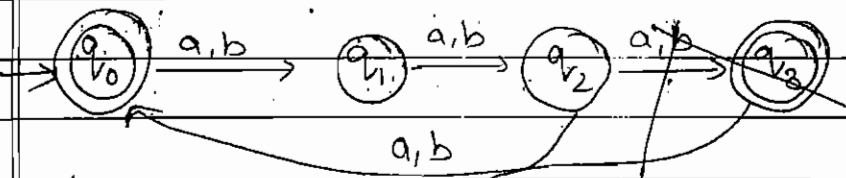
If length of string is at most n ,
no. of states = $n+2$

→

$\Sigma = (a, b)$

where LOS is divisible
by 3.

$L = \{0, 3, 6, 9, \dots\}$

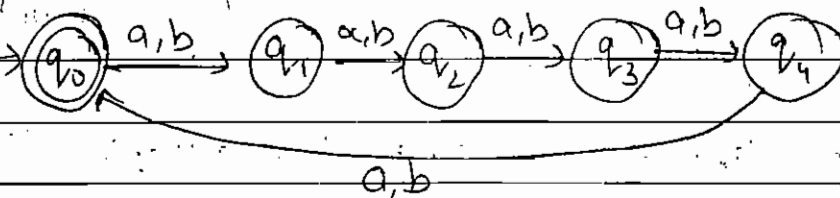


→

$\Sigma = (a, b)$

LOS divisible by 5

$L = \{0, 5, 10, 15, \dots\}$



⇒

If length of string is divisible by n , no.
of states = n .

thing is
4.

$$\Sigma = (a, b)$$

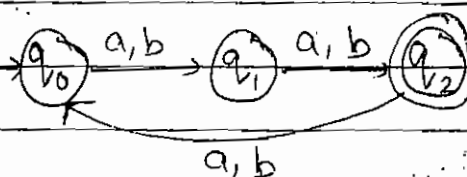
where LOS is ~~quadrant~~
Congruent to $2 \bmod 3$.
(equal)

$$L = \{2, 5, 8, 11, 14, \dots\}$$

3 3 3

$2 \bmod 4$
means remainder
is 2.

$3 \bmod 5$
mean reme
is 3



Σ $\subseteq$ a, b

is n,

$\Rightarrow$ If LOS is congruent to $m \bmod N$,
no. of states = N .

is divisible

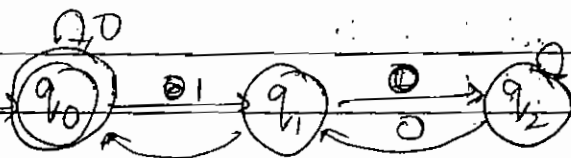
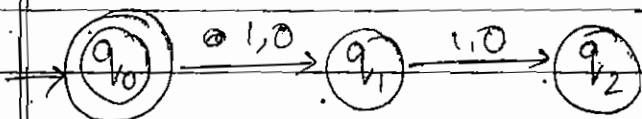
NUMBER SYSTEM:

accepts
all binary nos, which
are divisible by 3.

$$\Sigma = (1, 0)$$

$$L = \{0, 3, 6, 9, \dots\}$$

0 11 110 1001



	0	1
q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

→ For problems related to no. system, method is -

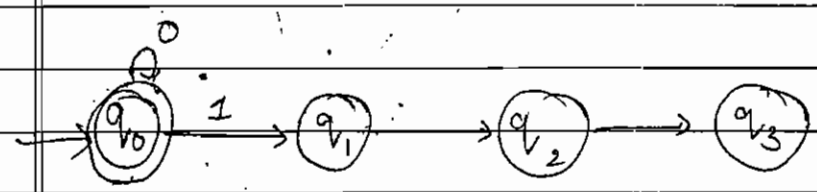
① Find no. of states which is equal to no. of possible remainder. i.e. remainders for any no. divided by 3 are 0, 1, 2.

② Find final state 1st
Initial state because, remainder is 0.

③ Transitions
• Better solve it through transition table.

→ $\Sigma = (0, 1, 2)$ that accepts all base 3 nos. which are divisible by 4.
 $L = \{$

No. of states = 4 0, 1, 2, 3
Final state =



⇒ The DFA that accepts all base n nos. which are divisible by n requires n no. of states.

→ Construct DFA that accepts all binary nos. which are divisible by

(i) either 2 or 3.

(ii) by 2 but not by 3.

(iii) 3 but not by 2.

(iv) not by 2 or 3.

(i) no. of states = $LCM(2, 3) = 6$

$\{0, 1, 2, 3, 4, 5\}$

final states = nos. of nos. divisible by 2 $\cup$ nos.

divisible by 3
 $= \{0, 2, 3, 4\} = q_0, q_2, q_3, q_4$

Transitions same as earlier.

(ii) no. of states = 6

final states = {nos. divisible by 2 but not by 3}
 $= \{2, 4\} = q_2, q_4$

(iii) no. of states = 6
 final state = 3

(iv) no. of states = 6

final states = $\{1, 5\} = q_1, q_5$

2nd m

requires

→ Construct DFA that accepts all base 4 nos. which are divisible by either 2 or 3 but not by 5. 0, 1, 2, 3

Binary

$$\text{no. of states} = \text{LCM}(2, 3, 5) = 30$$

$$\{0, 1, 2, \dots, 29\}$$

$$\text{final states} = \{0, 2, 4, 6, \dots\} \cup \{0, 3, 6, 9, \dots\} - \{0, 10, 15, 20, 25\} \\ = \{2, 3, 4, 6, \dots\}$$

6th

1st nos.

divisible by 3?

q₃, q₄

→ How many nos. of states in DFA that accepts all hexadecimal nos. which are divisible by 3 by not by 5 and 7.

not by

$$\text{no. of states} = \cancel{3 \times 5} \text{ LCM}(3, 5, 7) = 105$$

q₂, q₄

→ How many no. of DFA's are possible with 2 states x & y for input alphabet $(0, 1)$.

~~2~~ $\times 16$

(x) (y)

	0	1	
2 x	2	2	possibilities
2 y	2	2	
<u>4</u>	<u>16</u>	<u>16</u>	$= 64 + 64 = 128$

possibilities on symbols $\Rightarrow$ $x \xrightarrow{0} x$ & $x \xrightarrow{0} y$
 $x \xrightarrow{1} x$ & $x \xrightarrow{1} y$

possibilities on states $\Rightarrow$ (x) or x

→ How many no. of FAs are possible with 3 states x, y, z over the i/p alphabet (a, b, c) , where x is always initial state.

	a	b	c
2 x	3	3	3
2 y	3	3	3
2 z	3	3	3
<u>8</u>	<u>27</u>	<u>27</u>	<u>27</u>

3^9

⇒ NPDA is default PDA.

possible

with alphabet

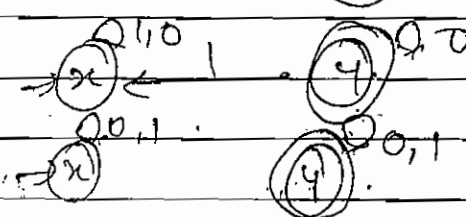
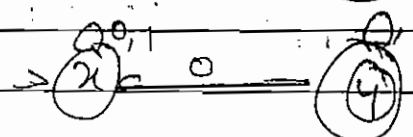
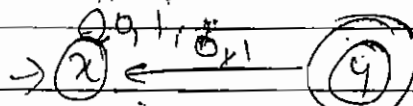
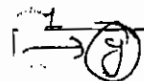
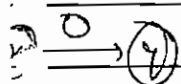
→ How many p no. of i fns are possible with 2 states x & y over the I/P alphabet (a, b) which accepts empty lang. only, where x is always initial state.

but,

	a	b
x	2	2
y	2	2

20

= 128



able

if P
ys

→ How many DFA are possible with 2 states x & y over I/P alphabet $(0, 1)$ which accepts complete lang., where x is always initial state.

	0	1
x	2	2
y	2	2

8 + 4 = 20

→ How many nos. of FA are possible with N states ($Q_1, \dots, Q_n$) for the $\Sigma = \{1, 2, 3, \dots, m\}$, where Q_1 is always initial.

1 2 3 ... m

$2 Q_1$ $n \cdot n$ m $-$ m

$2 Q_2$

$2 Q_n$

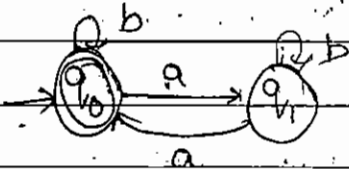
$2^n \times n^{nm}$

no. of FA's = $2^n \times n^{nm}$

→ (i) Construct Min^u. DFA that accepts $\Sigma = (a, b)$ where no. of a 's are even in every string.

$\Sigma = (a, b)$

$L = \{a^2b, a^4ab, \dots\}$



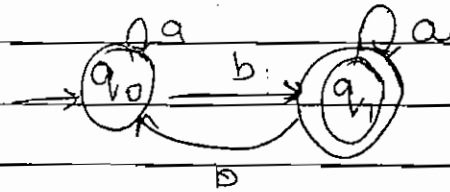
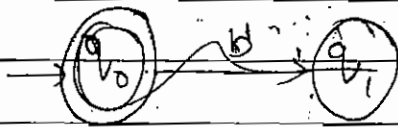
(i) no. of b's odd

is b's

for

ie Q_1

$$L = \{0, 1, 3, \dots\}$$

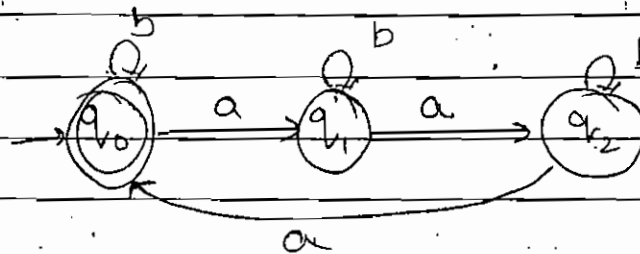


→

$$\Sigma = (a, b)$$

no. of a's are
divisible by 3

$$L = \{0, 3, 6, \dots\} \text{ in each string.}$$

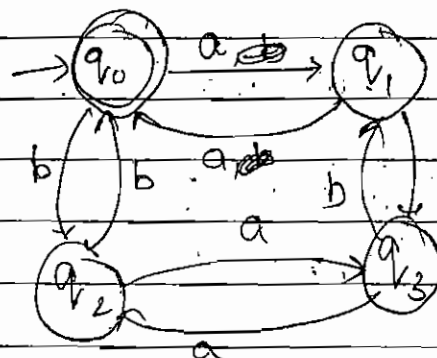


→

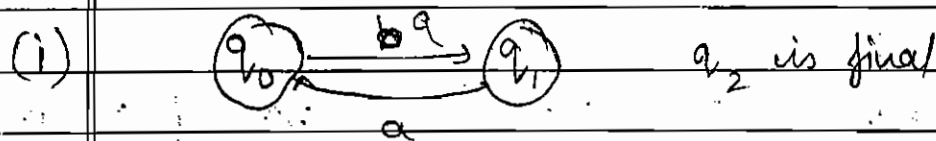
$$\Sigma = (a, b)$$

where no. of a's
& no. of b's are

$$L = \{0, 2, 4, 6, \dots\} \text{ even.}$$



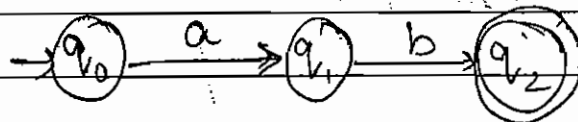
- (i) a is even, b is odd.
 $L = \{ab, 3, 5, \dots\}$
- (ii) a is odd, b is even.
 $\{1, 3, 5, \dots\}$
- (iii) a is odd, b is odd.



(ii) q_1 is final

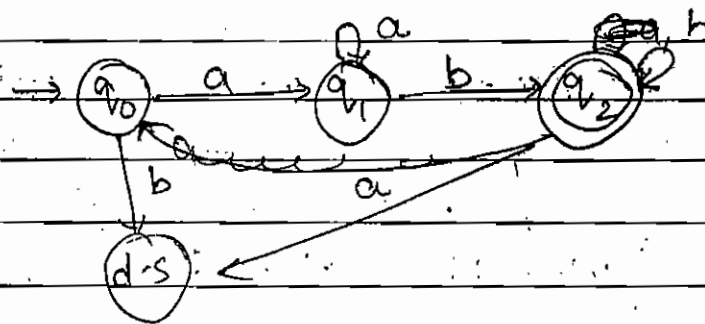
(iii) q_3 is final

→ Cons: DFA for lang, $L = \{a^n b^n \mid n \geq 1\}$.
 $\Sigma = (a, b)$
 $L = \{ab, aabb, aaabbb, \dots\}$



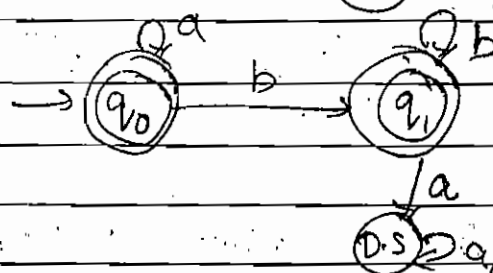
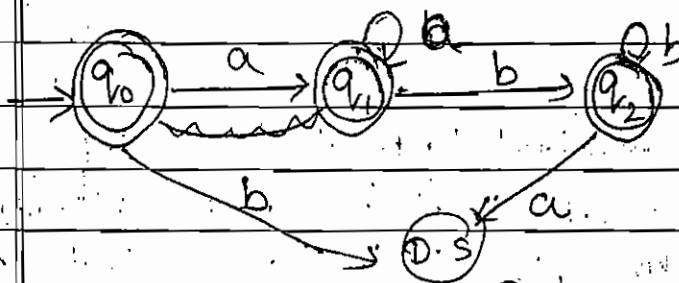
FA is not possible.

→ $L = \{a^n b^m \mid n, m \geq 1\}$
 $L = \{ab, aab, \dots\}$



→ $L = \{a^n b^m \mid n, m \geq 0\}$

$L = \{0, a, b, ab, \dots\}$



$k \geq 2$
 $n =$

aa

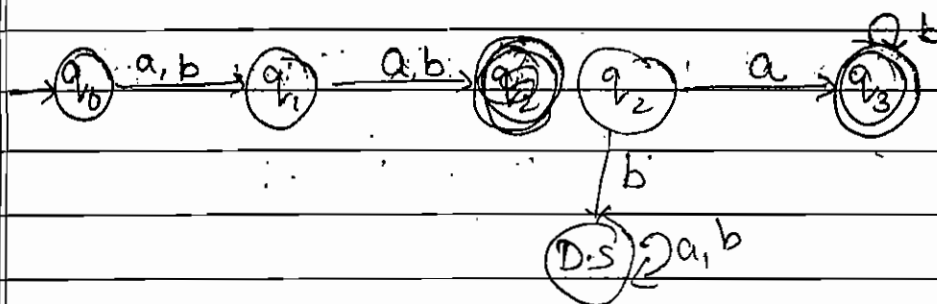
→ Min^m no. of states needed in DFA to recognize the Lang. $L = \{a^k \mid k \geq 0 \text{ \& } n \text{ is +ve int.}\}$

- A) $n+1$ B) $k+1$ C) 2^{k+1} D) 2^{n+1}

Solⁿ $L = \{a, aa, a \dots\}$
check for $n=1$

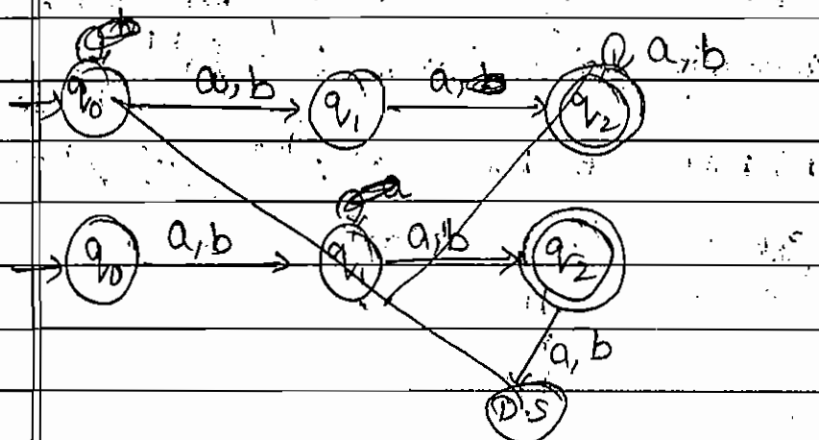
$n=2$
 $n=3$

→ $\Sigma = (a, b)$ where 3rd input symbol is a while reading the string from LHS.
 $L = \{aaa, bba, aba, baa, \dots\}$



⇒ The minimal DFA that accepts all string of a & b where nth i/p symbol is a while reading string from LHS, requires $(n+2)$ states.

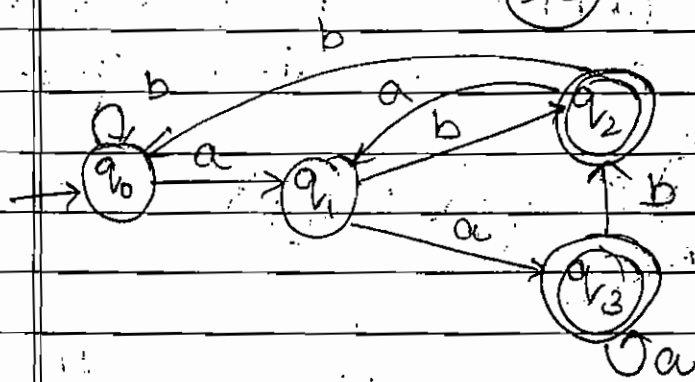
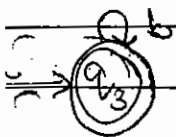
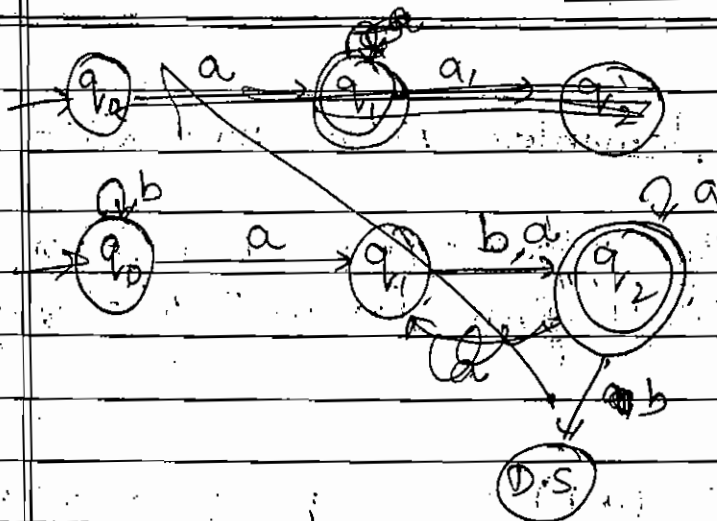
→ $\Sigma = (a, b)$ where 2nd I/P symbol is 'a' while reading string from R.H.S.
 $L = \{aa, ab, aab, \dots\}$



aaaab

bbbaa

Symbol is
string the
LHS.



all
input
string
states.

⇒ The minimal DFA that accepts all string of A & B's where nth I/P symbol is a while reading the string from RHS requires (2^n) states.

Symbol
string
RHS.

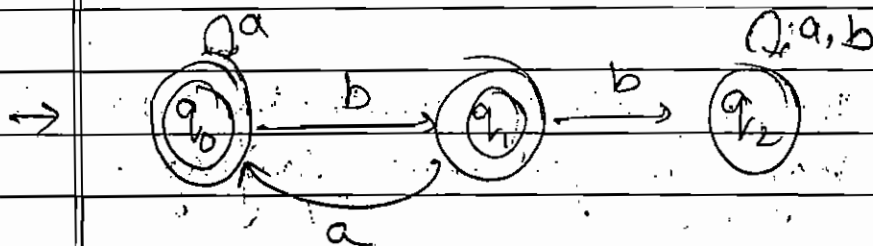
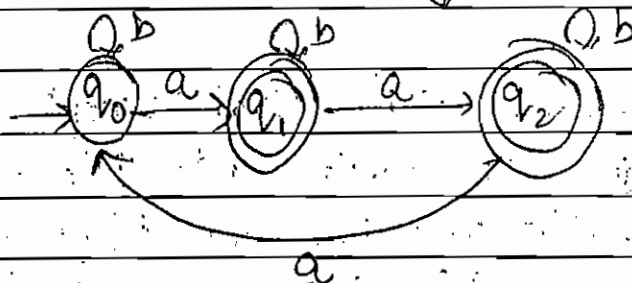
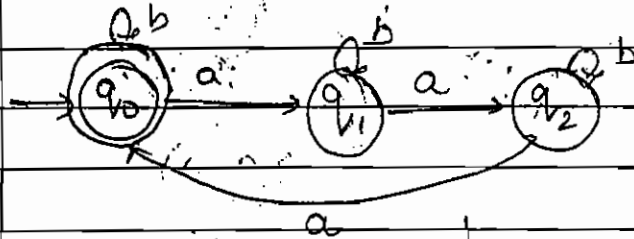
⇒ The complexity of constructing DFA is very high for some languages.

⇒ Whenever DFA designing is difficult, for such kind of lang. we have to construct NFA.

Complemented Finite Automata:-

By interchanging final & non-final states of given FA we'll get complemented FA.

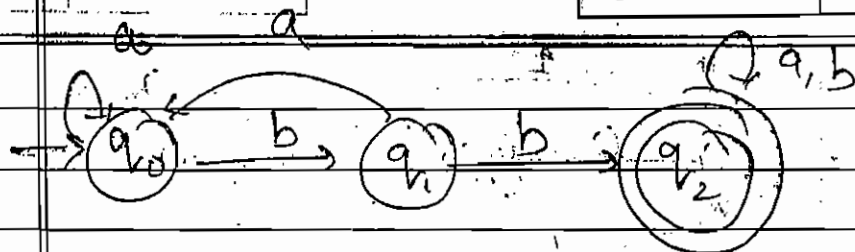
→ $\Sigma = (a, b)$ no. of a 's are not divisible by 3.



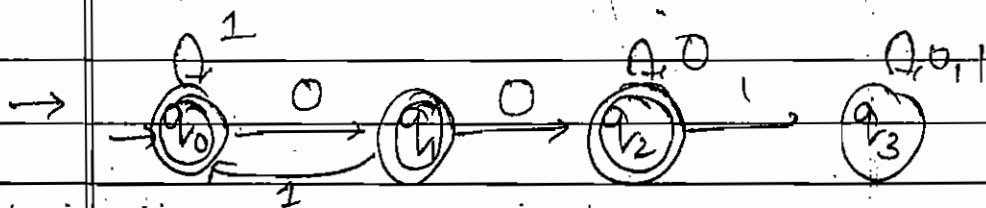
If a complement above DFA, then the lang. accepted by that DFA is -

- A) starting with a or b
- B) ending with a or b
- C) contains atleast 2 b's.

✓ NOT



final implemented

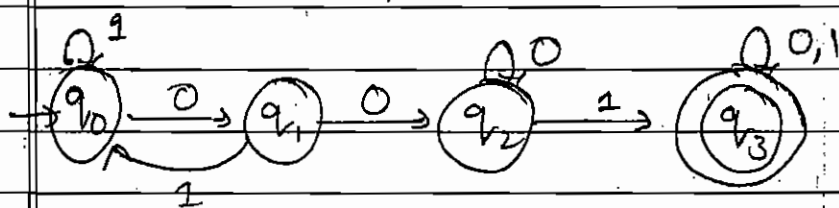


not

Complement above FA. Let

S denotes a 7bit binary string in which 1st, 4th & last bits are 1, then how many no. of strings are there in lang. accepted by above complemented DFA from S.

- A) 16 b) 8 c) 7 d) 9



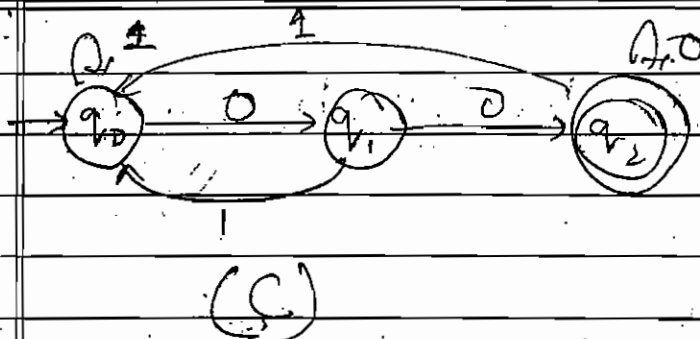
$$\frac{1}{2 \times 2} \quad \frac{1}{2 \times 2} \quad \frac{1}{2 \times 2}$$

Total string = $2 \times 2 \times 2 \times 2 = 16$

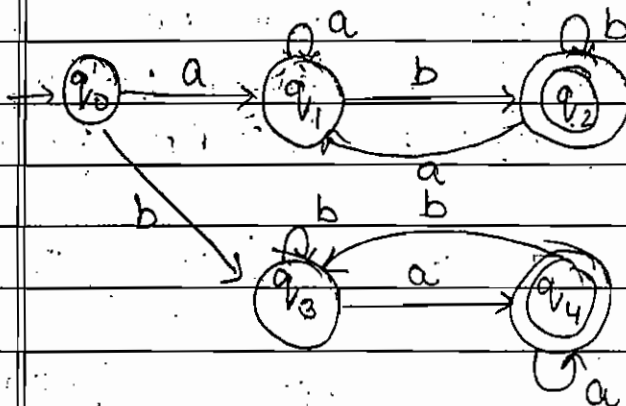
that

1 0 0 1 0 0 1 or 1 0 0 1 0 0 1
 0 1 0 1
 1 0 1 0
 1 1

$4 + 4 - 1$ (in case of set repetition is not allowed)
 $= 7$



→ $\Sigma = (a, b)$ where 1st & last symbols are diff.
 $L = \{ab, ba, abb, baa, \dots, \{abba, b\}\}$



c, d

Non deterministic FA = NFA

In NFA, from each and every state on every I/P symbol, exactly one transition exists. There may be zero or more than zero transitions exist.

* All DFAs are NFAs.

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

Q : Finite no. of states

Σ : I/P alphabet

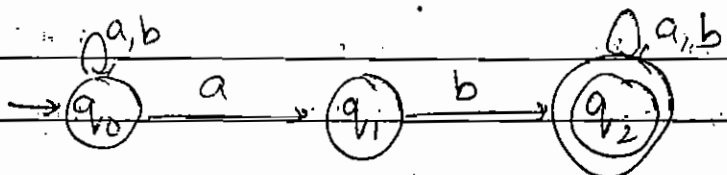
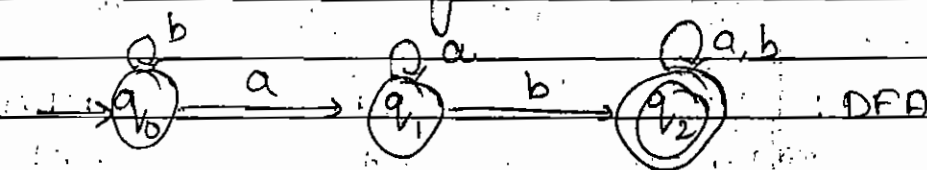
δ : transition func.

$$\delta: Q \times \Sigma = P(Q) \text{ or } 2^Q$$

q_0 : initiate state

F : set of final states

→ Construct NFA that accepts all string of as & bs where each string contain ab as substring.

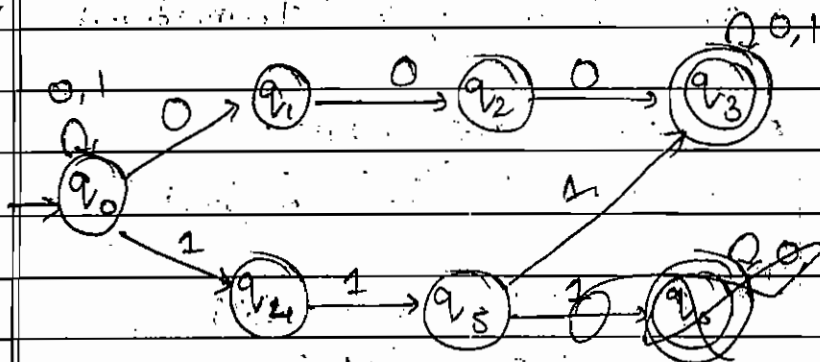


→ $\Sigma = (0, 1)$

tribit = 000 or 111

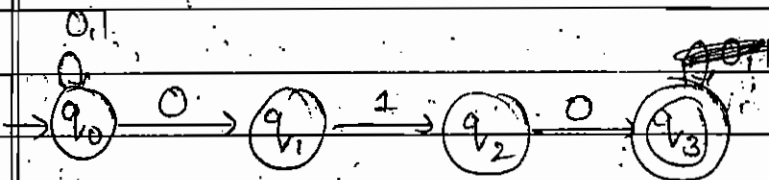
→ $\Sigma = \{0, 1\}$ where each string contains tribit as

$L = \{000, 111, 10001, \dots\}$ substring.



→ $\Sigma = \{0, 1\}$ where each string ends with 010.

$L = \{010, 1010, 0010, \dots\}$



WEL

DFA

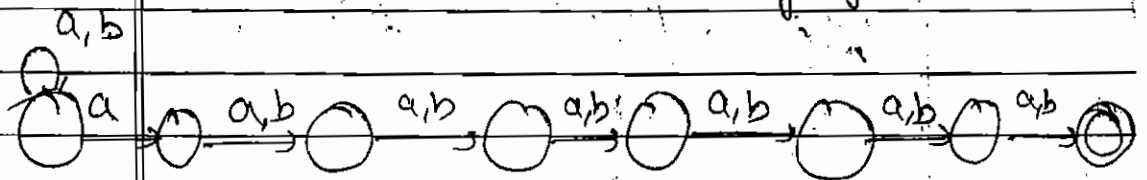
NFA

① $ w = n$	$n+2$	$n+1$
② $ w > n$	$n+1$	$n+1$
③ $ w \leq n$	$n+2$	$n+1$
④ $ w \bmod n = 0$	n	n
⑤ $ w = m \bmod n$	n	n
⑥ n^{th} I/P from L.H.S.	$n+2$	$n+1$
⑦ n^{th} I/P from R.H.L.	2^n	$n+1$

string
as

→ $\Sigma = (a, b)$

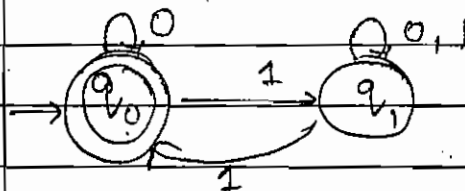
where n^{th} i/p symbol
is a while reading
string, from R.H.S.



$\Rightarrow$ for n^{th} i/p symbol from R.H.S., no. of
states $= n + 1$

string ends

→ Construct an equivalent DFA for following
NFA.



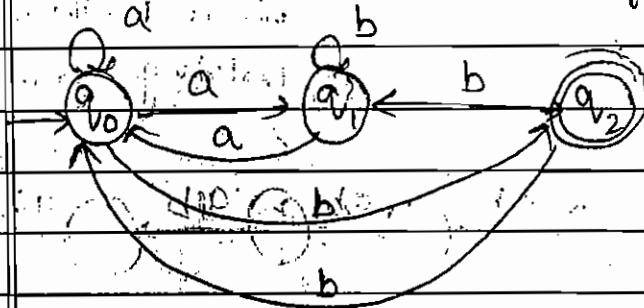
NFA:

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

DFA:

	0	ϕ	1
q_0	q_0		q_1
q_1	q_1		$[q_0, q_1]$
$[q_0, q_1]$	$[q_0, q_1]$		$[q_0, q_1]$

→ Construct equivalent DFA for NFA



NFA

	a	b
q ₀	{q ₀ , q ₁ }	q ₂
q ₁	q ₀	q ₁
q ₂	∅	{q ₀ , q ₁ }

DFA

	a	b
q ₀	[q ₀ q ₁]	q ₂
q ₁	q ₀	q ₁
q ₂	∅	[q ₀ q ₁]
[q ₀ q ₁]	[q ₀ q ₁]	{q ₁ , q ₂ }
[q ₁ q ₂]	∅	[q ₀ q ₁]

DFA:

	a	b
q_0	$[q_0 q_1]$	q_2
$[q_0 q_1]$	$[q_0 q_1]$	$[q_1 q_2]$
q_2 $[q_0 q_2]$	DS	$[q_0 q_1]$
$[q_1 q_2]$	q_0	$[q_0 q_1]$
DS	DS	DS

→ NFA

	0	1
q_0	q_1	$\emptyset$
q_1	$\{q_0, q_1\}$	$\emptyset$

DFA

	0	1
q_0	q_1	DS
DS	DS	DS
q_1	$[q_0 q_1]$	DS
$[q_0 q_1]$	$[q_0 q_1]$	DS
DS	DS	DS

→ NFA

	0	1
q_0	q_0	q_0
q_1	q_1	q_1
q_2	q_2	q_2

DFA

	0	1
q_0	q_0	q_0

⇒

While converting from NFA to DFA, no. of states possible in DFA are 1 in the best case and 2^n in the worst case.

→

	0	1
p	$\{p, q\}$	p
q	$\{r, s\}$	t
r	$\{p, r\}$	t
* s	$\emptyset$	$\emptyset$
* t	$\emptyset$	$\emptyset$

DFA:

	0	1
p	[pq]	p

	0	1
q	[rs]	r

	0	1
[pq]	[pqrs]	[pt]

	0	1
t [rs]	{ DS	DS

	0	1
[pt]	DS	DS

~~[pqrs]~~

	0	1
p	[pq]	p

	0	1
[pq]	[pqrs]	[pt]

	0	1
[pqrs]	[pqrs]	[pqrs]

	0	1
[pt]	DS	DS

AFD,
is
in

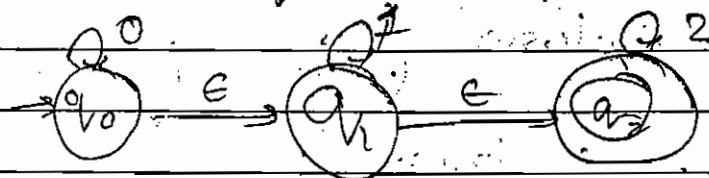
E-NFA:-

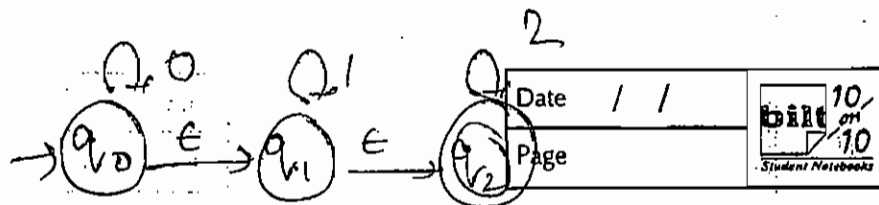
$(Q, \Sigma, \delta, q_0, F)$

$\delta: Q \times \Sigma \cup \{\epsilon\} \rightarrow P(Q)$ or 2^Q

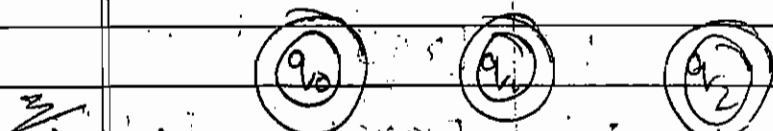
→

Construct eq. NFA for followin ENFA.





NFA:



⇒ All states which goes to final states using ϵ , will be final states in NFA.

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

i.e. all the possible states reachable through ϵ from q_0 .

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

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

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

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

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

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

$$= \epsilon\text{-closure}(q_0 \cup \emptyset \cup \emptyset)$$

$$= \epsilon\text{-closure}(q_0)$$

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

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

$$\delta(q_0, 2) = \{q_2\}$$

$$\delta(q_1, 0) = \emptyset$$

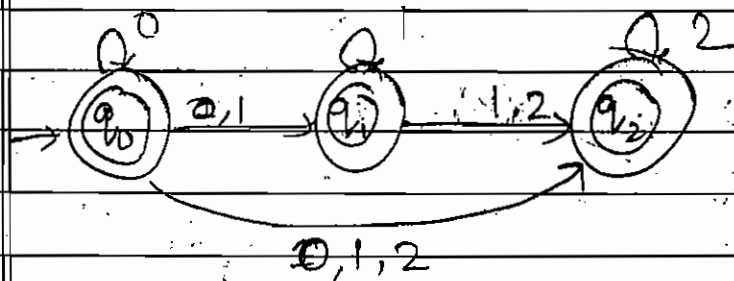
$$\delta(q_1, 1) = \{q_1, q_2\}$$

$$\delta(q_1, 2) = q_2$$

$$\delta(q_2, 0) = \emptyset$$

$$\delta(q_2, 1) = \emptyset$$

$$\delta(q_2, 2) = \{q_2\}$$

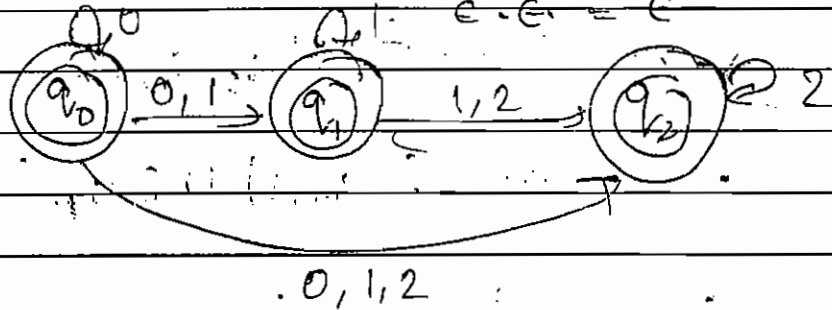


2nd Approach:

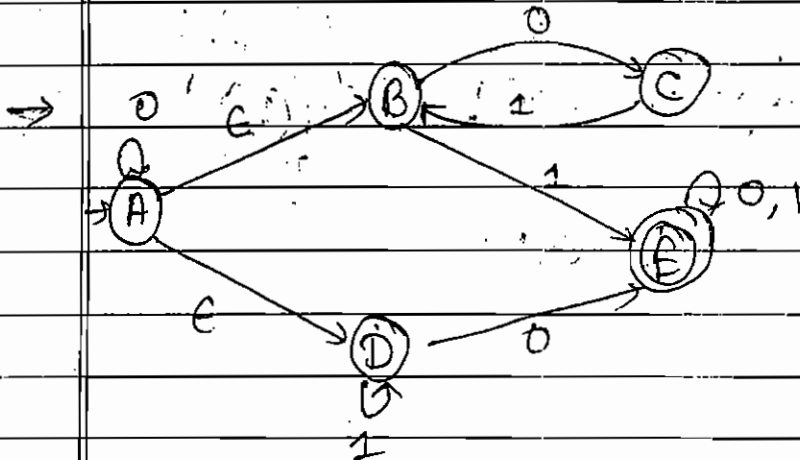
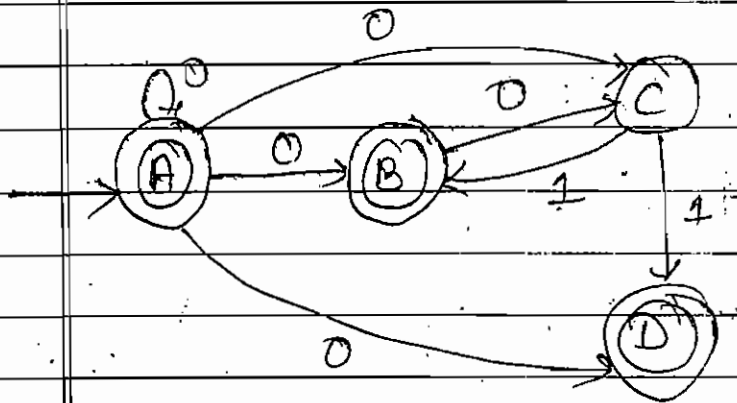
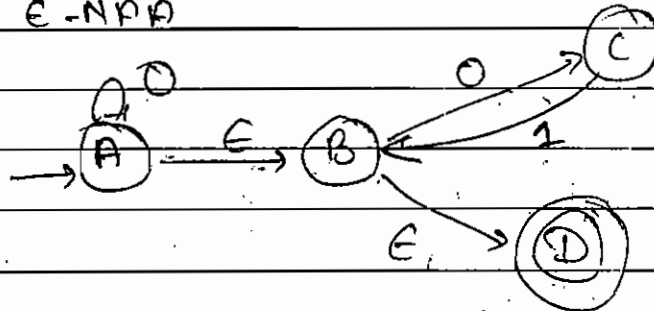
$$0 \cdot \epsilon = 0$$

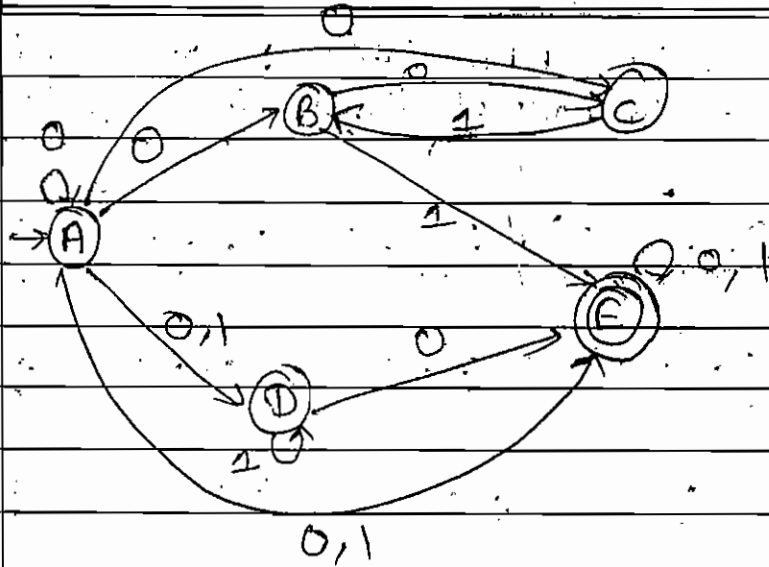
$$\epsilon \cdot 0 \cdot \epsilon = 0$$

$$\epsilon \cdot \epsilon = \epsilon$$



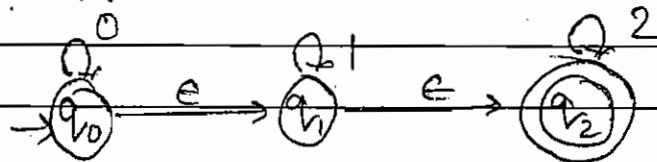
→ E-NFA





Conversion from ENFA to DFA:

→ Construct an equivalent DFA for following E-NFA.

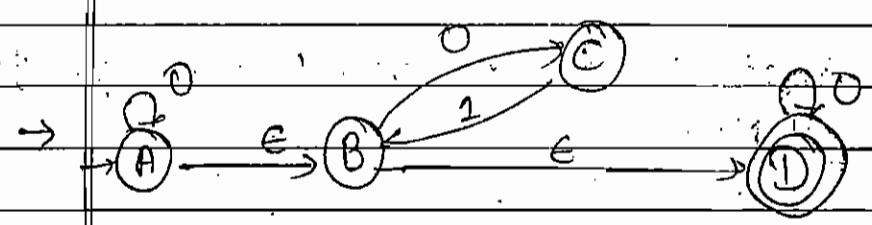


① Create NFA Table

	0	1	2
q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	$\emptyset$	$\{q_1, q_2\}$	$\{q_2\}$
q_2	$\emptyset$	$\emptyset$	$\{q_2\}$

② Create DFA table : Initial state is ϵ -closure of initial state of NFA

	0	1	ϵ
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$	$[q_1, q_2]$	$[q_2]$
$[q_1, q_2]$	$\emptyset$	$[q_1, q_2]$	q_2
q_2	$\emptyset$	$\emptyset$	q_2
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$



NFA:

	0	ϕ	1
$\textcircled{A}$	$\{A, B, C, D\}$		ϕ
$\textcircled{B}$	$\{C, D\}$		ϕ
$\textcircled{C}$	ϕ		$\{B, D\}$
$\textcircled{D}$	D		ϕ

is

DFA!

0

1

~~[A B C D]~~

[A B C D]

[B D]

[B D]

[C D]

DS

[C D]

D

[B D]

[D]

D

DS

DS

DS

DS

0

[A B D]

[A B C D]

DS

[A B C D]

[A B C D]

[B D]

[B D]

[C D]

DS

[C D]

D

[B D]

[D]

D

DS

DS

DS

DS

Decision Properties of FA:-

Decidable Problem: if there exist some generalised algorithm in order to solve the problem.

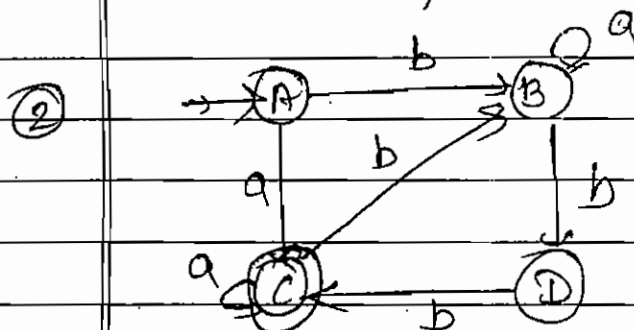
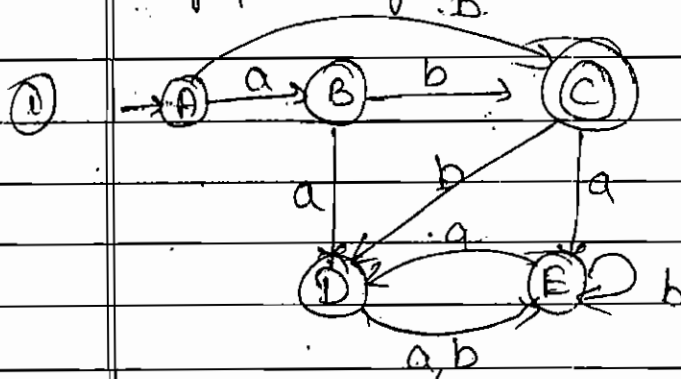
① Membership Problem

② Emptiness Problem Algo:

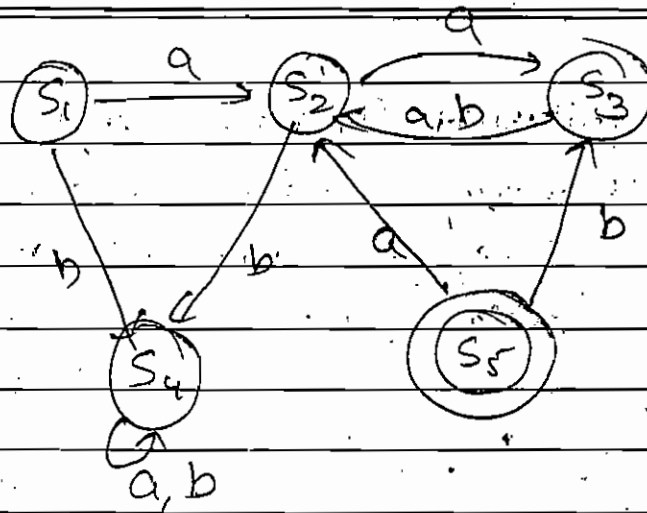
↳ Eliminate all inaccessible states.

↳ In remaining finite automata, if we find atleast one final state then the lang. accepted by that automata is non-empty otherwise empty lang.

→ Which of following Automata accepting empty lang.



③



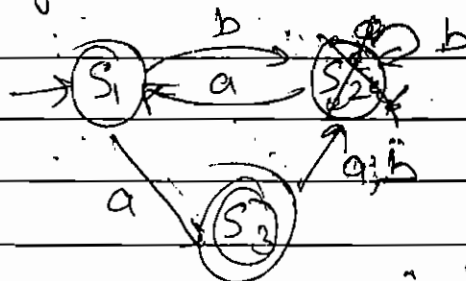
③

Finiteness Problem Algo:

↳ Eliminate all inaccessible states.

↳ Select some states from which we can't reach final state and delete those states.

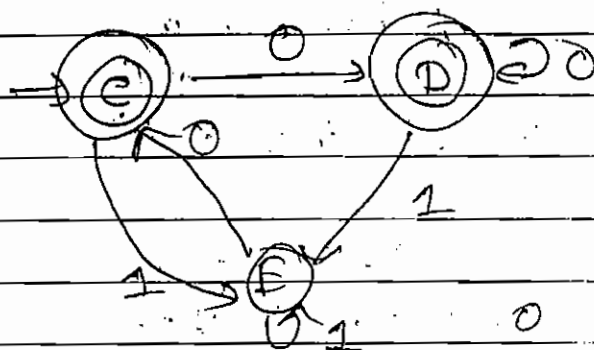
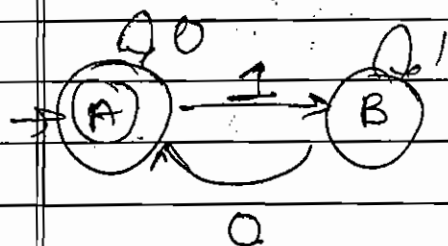
↳ In remaining finite automata if there exist any loop or cycle or self loop then the lang. accepted by finite automata is infinite.



④ Equivalence Problem Algo.

⇒ Checking whether lang. accepted by giving 2 automata are equal or non-equal.

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



Comparison Table

	0	1
→ (A, C)	(A, D)	(B, E)
(A, D)	(A, D)	(B, E)
(B, E)	(A, C)	(B, E)
(B, E)	ε	

Regular Expressions:

The declarative way of representing a Reg. Lang. ^(simplest form rather than actual)

If we can construct DFA from a lang. then it is Reg. lang.

Operators —

$+$ $\Rightarrow$ Union Op.

$\cdot$ $\Rightarrow$ Concatenation

$*$ $\Rightarrow$ Kleene closure.

We can construct R.E. for R.L. by taking symbols in I/P alphabet as operands and by using above 3 operators.

	R.L.	R.E.
1)	$L = \{ \}$	$\emptyset$
2)	$\{ \epsilon \}$	ϵ
3)	$\{ a, b \}$	$(a+b)$
4)	$\{ a \}$	a
5)	$\{ \epsilon, a, aa, \dots \}$	a^*
6)	$\{ a, aa, aaa, \dots \}$	a^+ $\rightarrow$ +ve closure (excludes ϵ)
7)	$\{ a, b, ab, ba, aa, \dots \}$	$(a+b)^+$

→ Find R.E. that generates all strings of 0s & 1s where each string starts with 0.

$$0(0+1)^* \quad \{0, 00, 01, \dots\}$$

→ starting with 0 ending with 1

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

→ where each string start & ends with diff. symbol.

$$\cancel{0(0+1)^*} (0+1)^* + 1(0+1)^* 0$$

→ string start & ends with same symbol. $L = \{0, 1, 00, 11, \dots\}$

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

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

→ when LOS is exactly 4.

$$L = \{0000, 0111, 1111, \dots\}$$

$$(0+1)(0+1)(0+1)(0+1)$$

(a, b, c)

→ How many no. of states in DFA that accepts lang. generated by following R.E.

$$(a+ b+ c) (a+ b+ c) \dots (n-2) \text{ times}$$

$$\text{length} = (n-2) = \text{LOS}$$

$$\text{exactly } (n-2)$$

$$\text{no. of state} = N + 2$$

$$= (n+2) - 2 = n$$

→ then LOS is atleast 3.

$$L = \{ \epsilon, a, b, c, ab, ba, bc, cb, ca, ac, \dots \}$$

$$\text{no. of state} = n+1$$

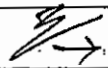
$$(0+1)(0+1)(0+1)(0+1)^+$$

or

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

→ How many no. of states in DFA that accepts Lang. generated by foll. R.E.
 $(a+b)(a+b) \dots (n-2 \text{ times}) (a+b)^+$
 LOS $\geq n-1$

$$\therefore \text{no. of state} = N+1 = (n-1)+1 = n$$



LOS $\neq 5$

$$L = \{0, 00, 000, \dots, 00000\}$$

$$\times (0+1)(0+1) \dots (0+1)$$

$$\times (0+1) + (0+1)(0+1)$$

$$(0+1+1)(0+1+1)(0+1+1)(0+1+1)(0+1+1)$$

→ LOS is divisible by 4.

$$L = \{0, 4, 8, 12, \dots\}$$

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

→ LOS is odd.

$$L = \{1, 3, 5, \dots\}$$

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

→ Each string contains even no. of 0s followed by odd no. of 1s.

$$L = \{1, 001, 111, 00001, \dots\}$$

$$[(00)^* (11)^* 1]$$

→ where no. of 0s are divisible by 3.

$$L = \{0, 000, 0000001, \dots\}$$

$$(000)^* 1^*$$

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

(0+1)*

→ Find R.E for R.L.

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

Not possible.

$$\rightarrow L = \{0^n 1^m \mid n, m \geq 1\}$$

$$L = \{01, 001, 011, \dots\}$$

$$0^+ 1^+$$

$$\rightarrow L = \{0^n 1^m \mid n+m \text{ is even}\}$$

$$L = \{001, 0001, 0111, \dots\}$$

$$0^+ 1^+$$

$$\rightarrow (0^* 1^*) (0^* 1^*)$$

$$\times [(0^+ 1^+) (0^+ 1^+)]^*$$

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

→ $L = \{0^n 1^m \mid n+m \text{ is odd}\}$
 even + odd = odd
 odd + even = odd

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

→ where No. of 0s are exactly 4 in each string

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

→ where No. of 0s are atleast 2 in each string

$$1^* 0 1^* 0 1^* 0^* 1^*$$

or

$$1^* 0 1^* 0^+ 1^*$$

or

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

→ No. of 0s are at most 3.

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

→ How many no. of states req. in DFA that accepts lang. gener. by Reg. E.

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

4th element from R.H.S is 1
so no. of state = $2^4 = 16$

each

Identities of R.E. :-

① $\emptyset + R = R + \emptyset = R$

② $\emptyset \cdot R = R \cdot \emptyset = \emptyset$

③ $\epsilon \cdot R = R \cdot \epsilon = R$

④ $\epsilon^* = \epsilon$

~~⑤~~ $\emptyset^* = \epsilon$

⑥ $\epsilon + RR^* = R^*R + \epsilon = R^*$

⑦ $(a+b)^* = (a^* + b^*)^*$
 $= (a^* b^*)^*$
 $= (a^* + b)^*$
 $= (a + b^*)^*$
 $= a^*(b a^*)^*$

→ Construct the following R.E from P & Q
 where

$$P = (01)^* + 1)^*$$

$$Q = (01)^* (1(01)^*)^*$$

✓ a) $P = Q$ b) $P \in Q$

c) $P > Q$ d) none.

$$P = ((01)^* + 1)^*$$

$$Q = ((01) + 1)^*$$

①

→ The R.E having minimal * heights
 is preferable.

→ Construct which of these 2 are equal

① $(0+e)(00)^*$ $0(00)^* + 00^*$
 A

② $0(00)^*$

③ $(00)^*$

④ 0^*

2006

A) I, II B) II, IV C) I, III D) I, IV

P & Q

→

Construct following R.E. - P & Q -

$$P = (111^*)^*$$

$$Q = (11 + 111)^* (A+B)^*$$

$$L_1 = \{0, 2, 3, \dots\}$$

$$L_2 = \{0, 2, 3, 4, \dots\}$$

which is true about P & Q

~~A)~~

~~B)~~

~~C)~~

- ① a) P = Q b) P ⊂ Q c) P ⊃ Q d) none.

Is

②

how many no of strings are there in minimal D.F.A that accepts P & Q.

A) 4, 3

B) 3, 3

C) 3, 4

d) none

equal

*

4

2006

→

$$\text{Let } L = (111 + 1111)^*$$

How many no of states are there in minimal D.F.A that accepts L.

a) 15

b) 8

c) 9

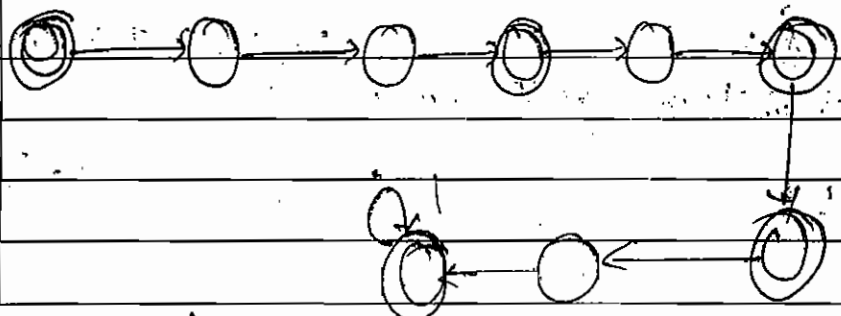
d) 5

$$L = \{0, 3, 5, 6, 8, 9, 10, 11, 12, 13, 14, 15, \dots\}$$

8 states

I, IV

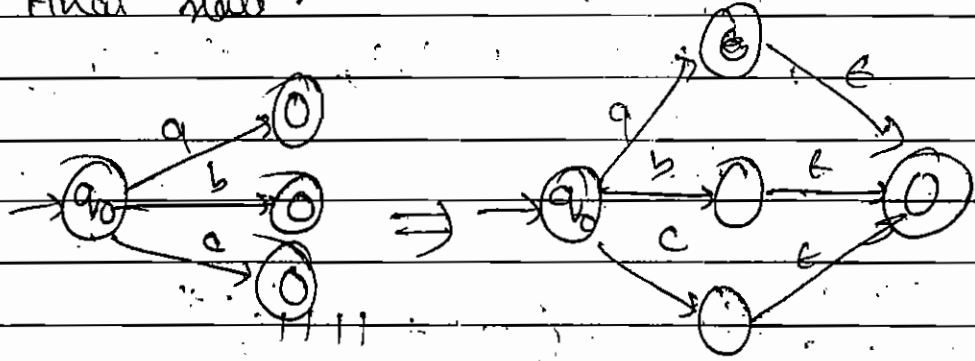
solⁿ → Untill it create DFA upto the length after which it is accepting all the lengths then for that state makes self loops.



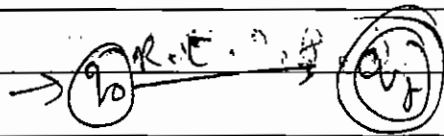
9 states.

Conversion from FA to RE :-

→ Convert any no. of ^{final} states into single final state.

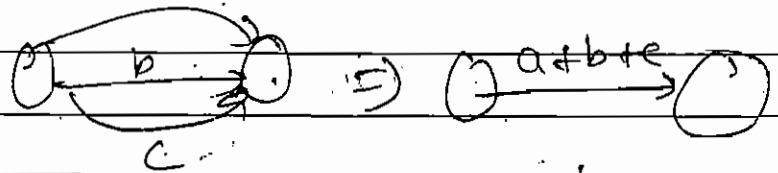


→ Add only ^{is to} ~~to~~ edge from initial to final

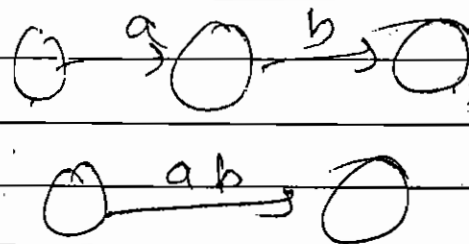


Conditions:-

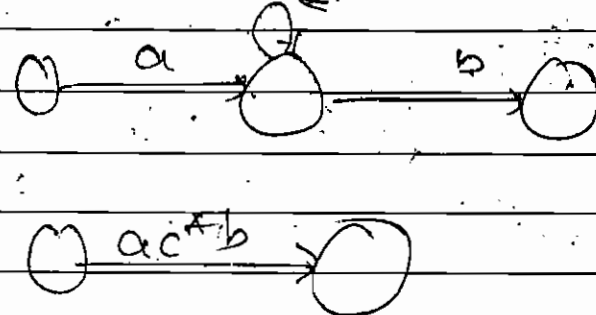
(1)



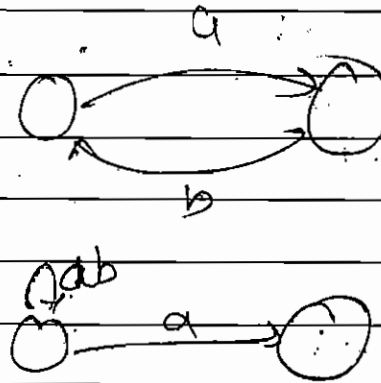
(2)



(3)



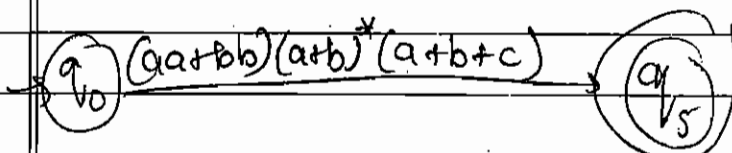
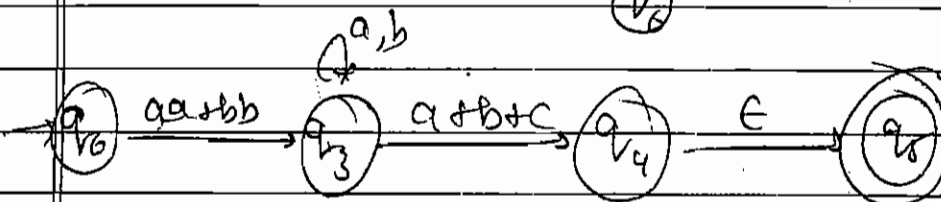
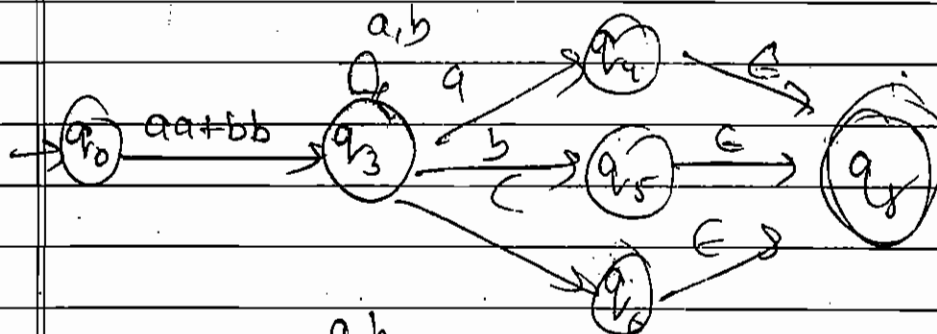
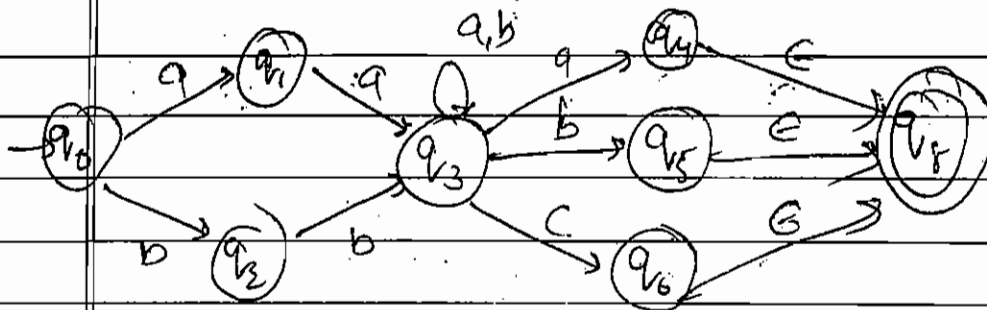
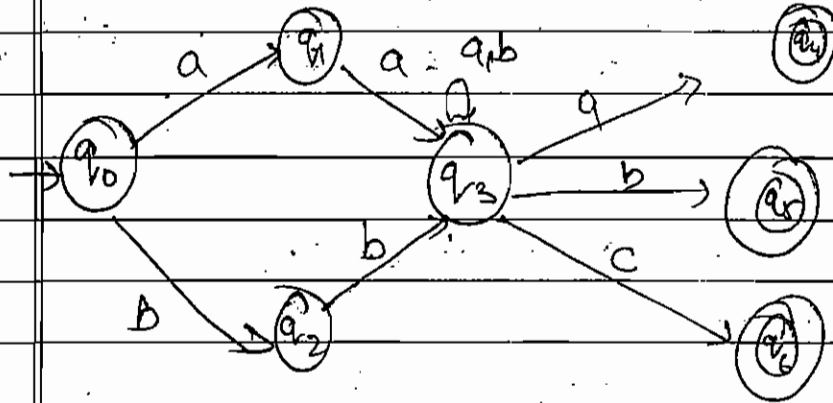
(4)



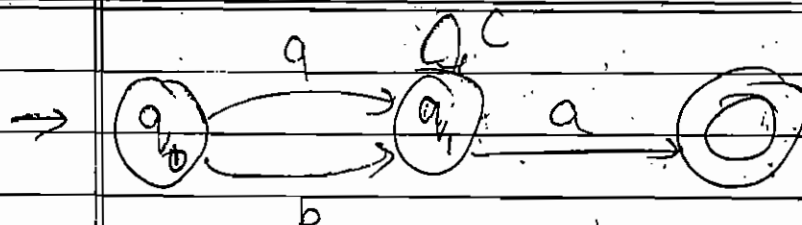
single

if find

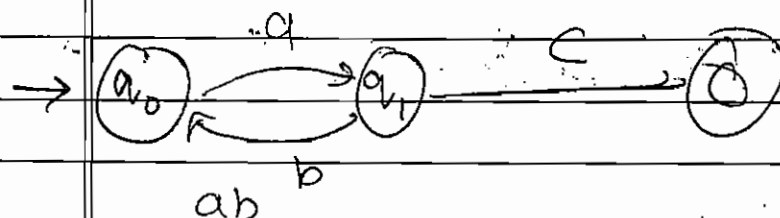
→ Construct RE for following NFA -



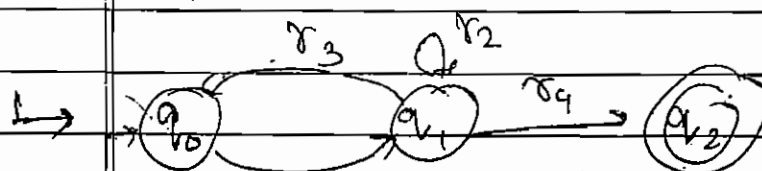
$(aa+bb)(a+b)^*(a+b+c)$



$$(q_0) (a+b)^* c^* a \rightarrow (q_2)$$



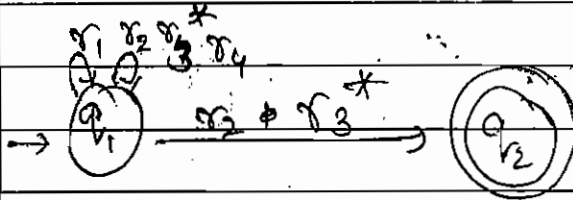
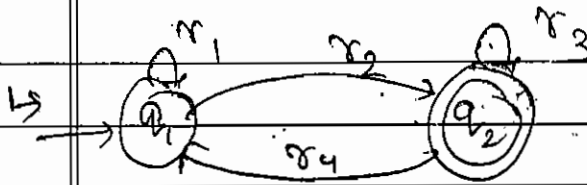
$$(q_0) (ab)^* ac \rightarrow (q_2)$$



$$(q_0) (r_1 r_2^* r_3)^* r_1 r_2^* r_4 \rightarrow (q_2)$$

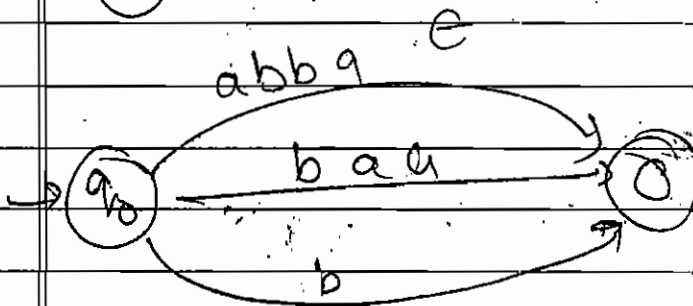
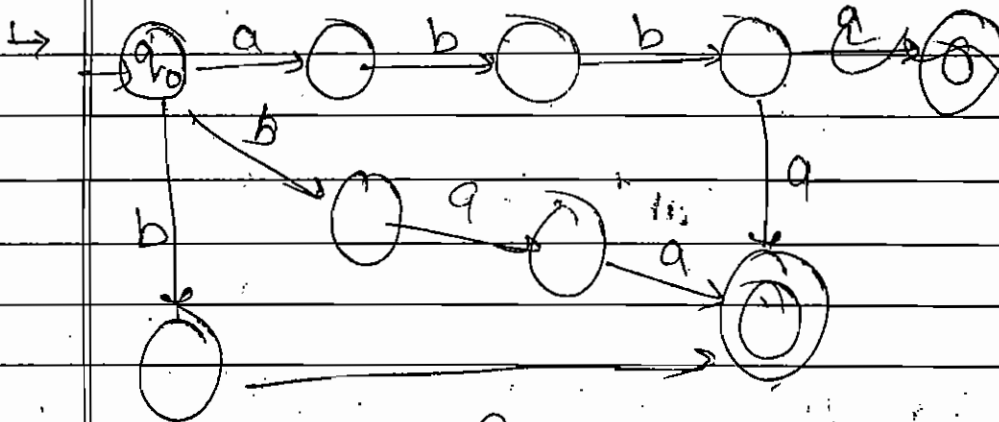
$$X(r_1 r_2^* r_3)^* r_1 r_2^* r_4$$

$$(r_1 r_2^* r_3)^* r_1 r_2^* r_4$$

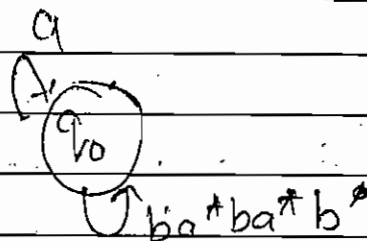
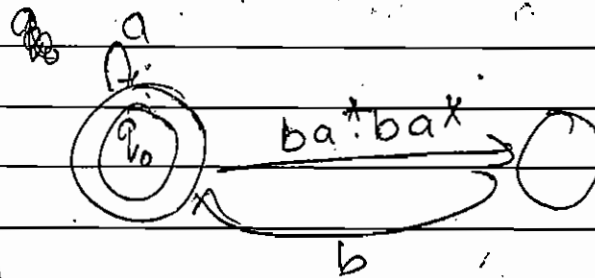
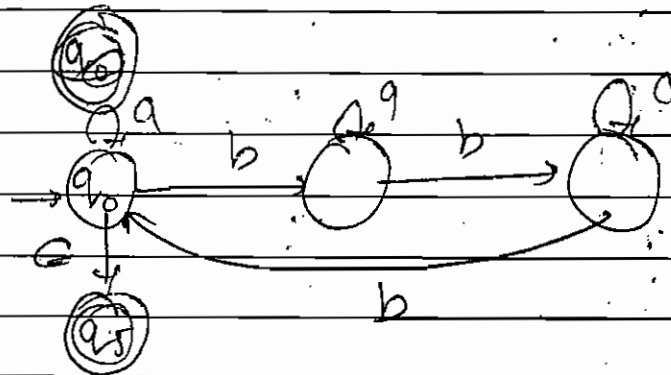
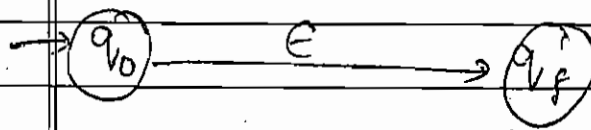
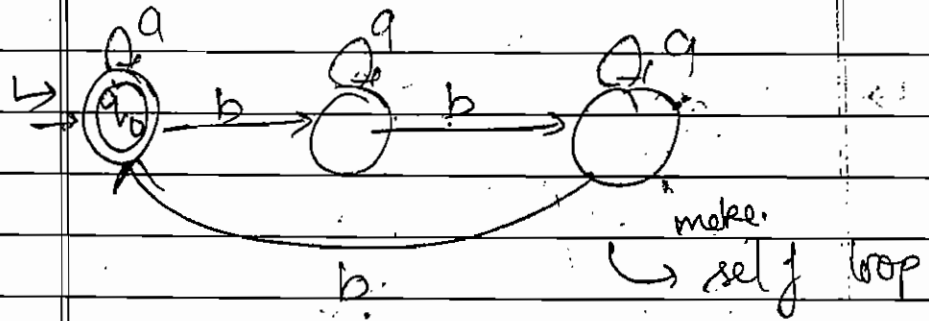


a, b
Q

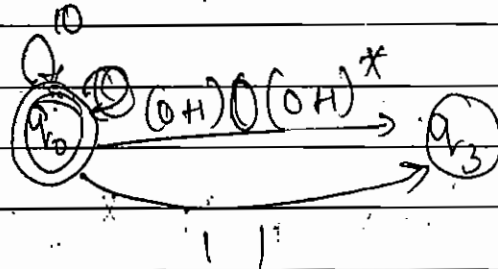
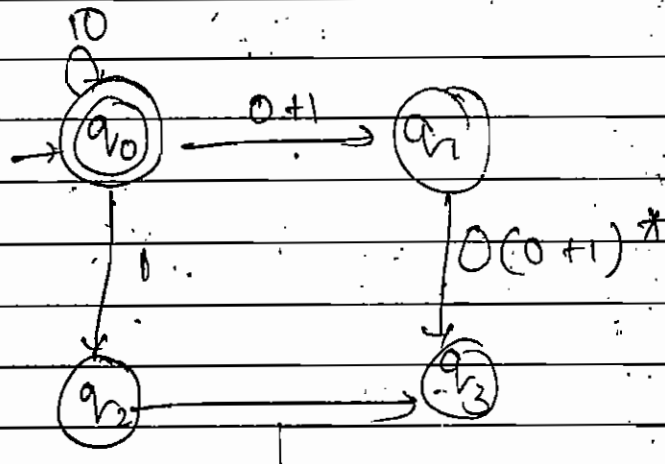
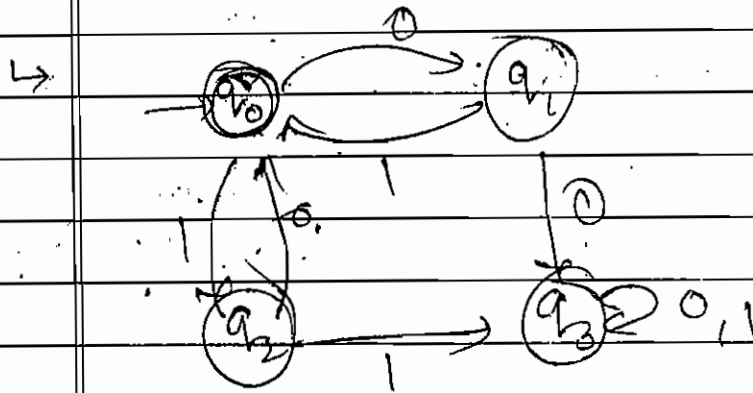
$$R = (r_1 + r_2 r_3^* r_4)^* r_2 r_3^*$$



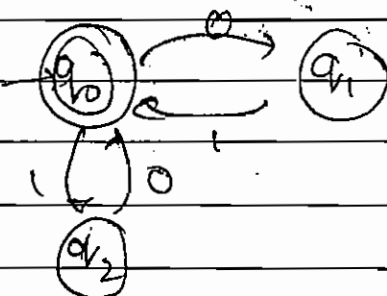
$$abba + baa + b$$



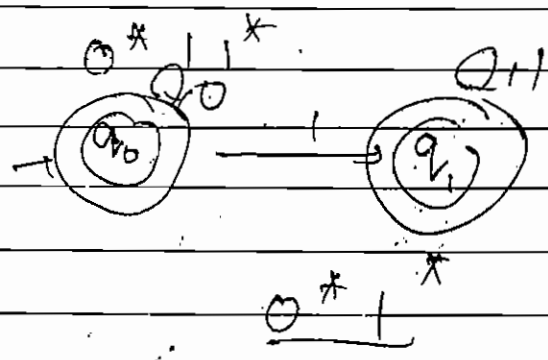
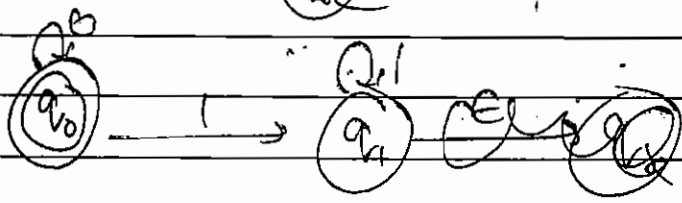
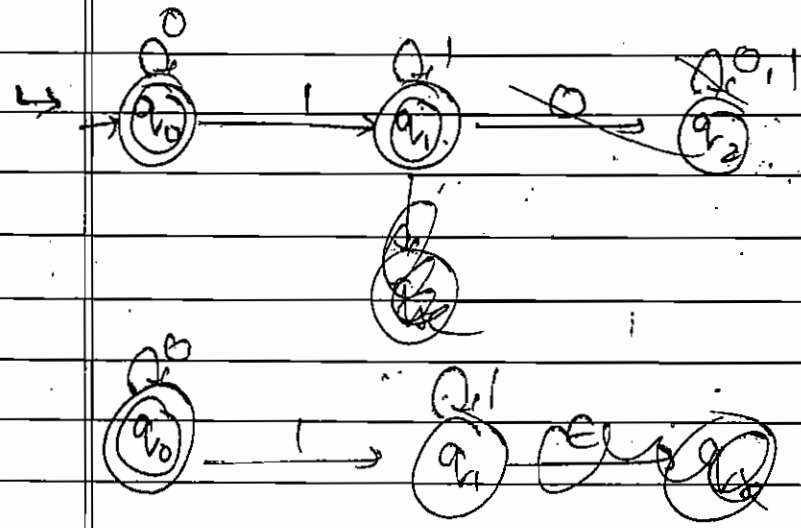
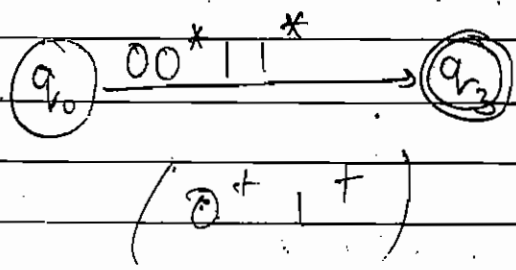
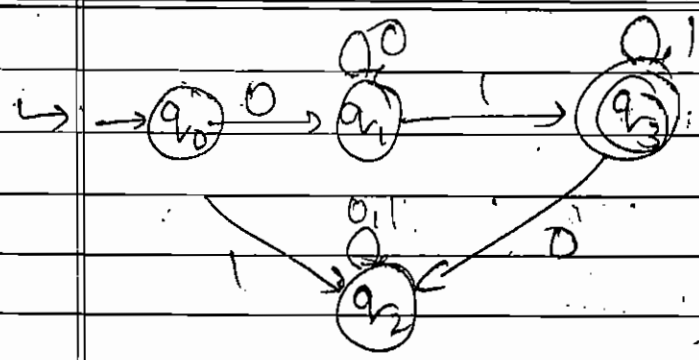
$a + ba^*ba^*b$



Note! If there is any dead lock state,
delete it



$\Rightarrow (01+10)^*$

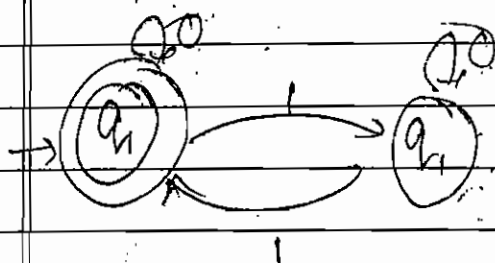


Stali,

→ Find R.E for ab. (0's & 1's
where no. of 1s are even in
every string.

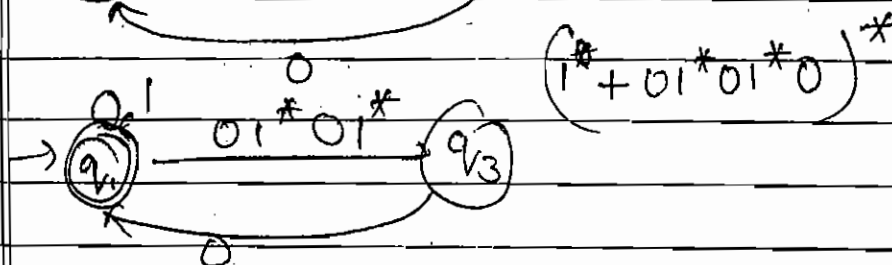
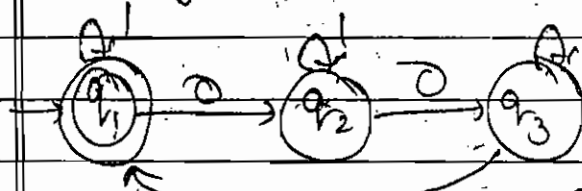
$$L = \{0, 01, 110, 01110\}$$

$$X \quad (0^* 1 0^* 1 0^*)^*$$



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

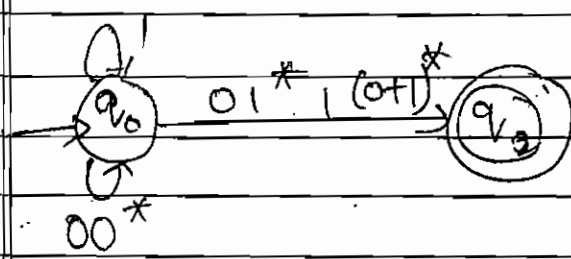
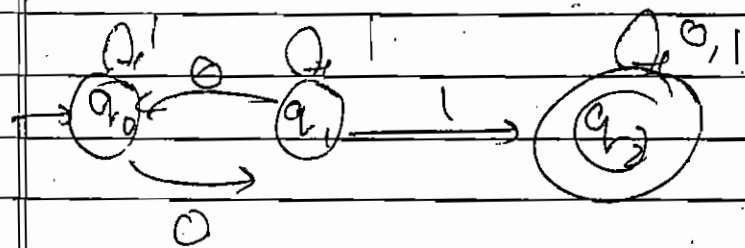
→ No. of 0s are divisible by 3



Note: First create FA then convert into R.E.

2- L's
in

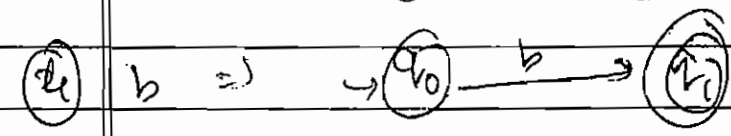
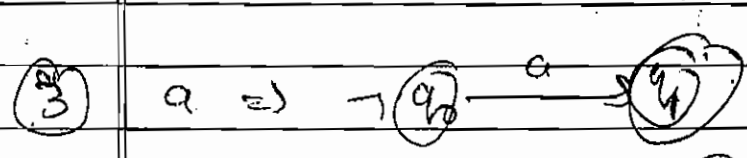
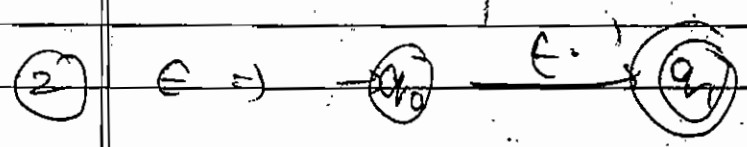
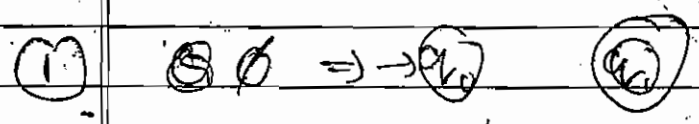
→ Convert R.E. into NFA. into RE.

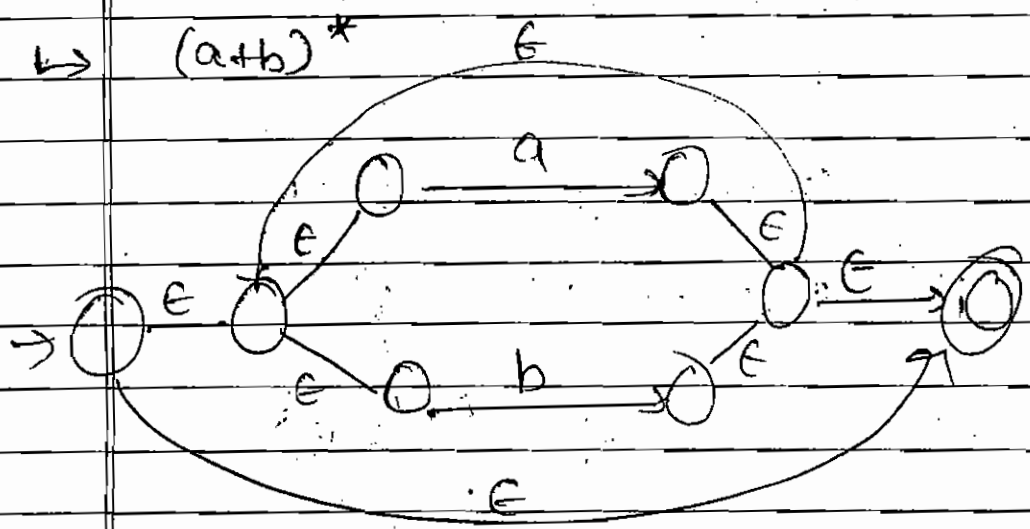
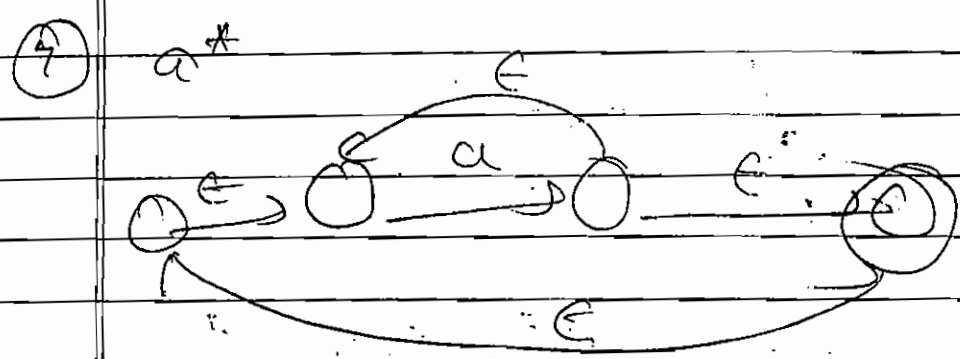
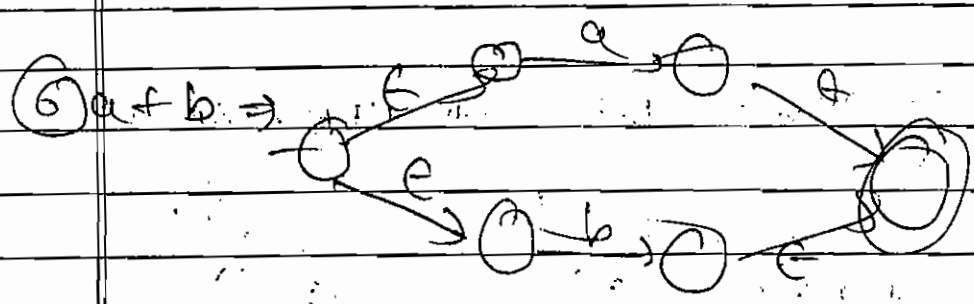
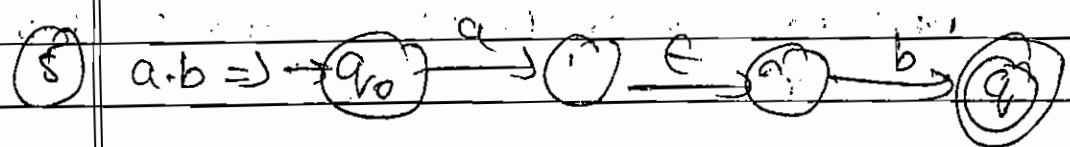


$$X (1 + 010^*)^* 01^* 1(0+1)^*$$

$$(1 + 01^*0)^* 01^* (0+1)^*$$

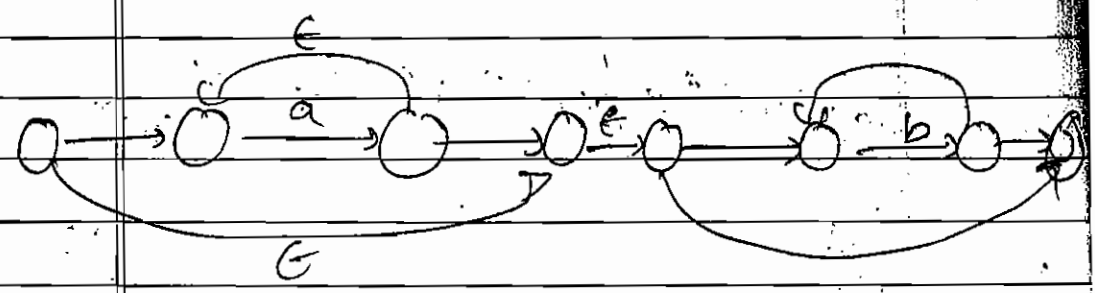
Conversion of R.E. to FA:-





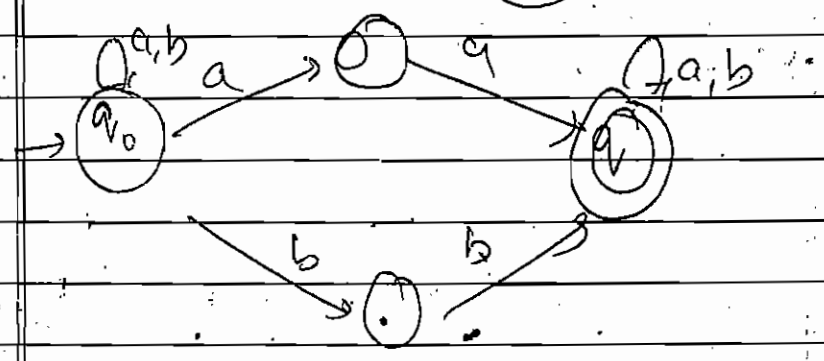
Q. 1

$a^* b^*$

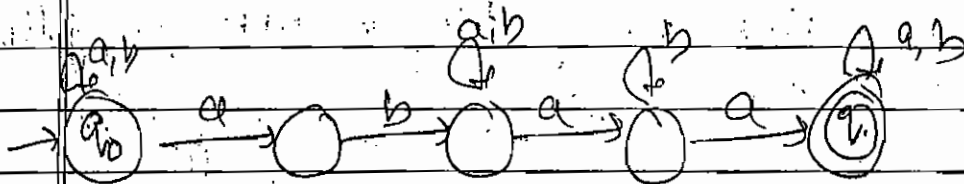
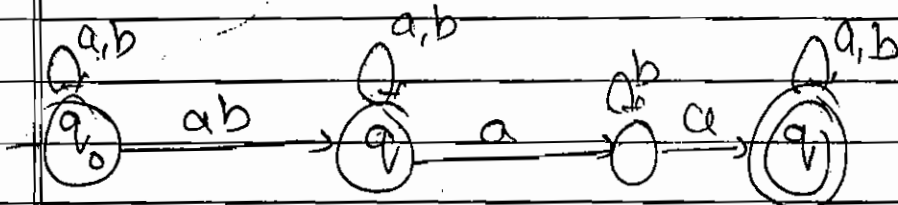
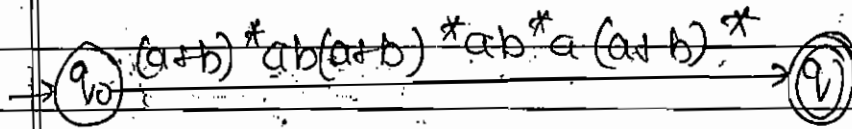


Q. 2 Find no. of strings in NFA for following R.E.
 $(a+b)^* (aa+bb) (a+b)^*$

Q. 3 $(a+b)^* (aa+bb) (a+b)^*$

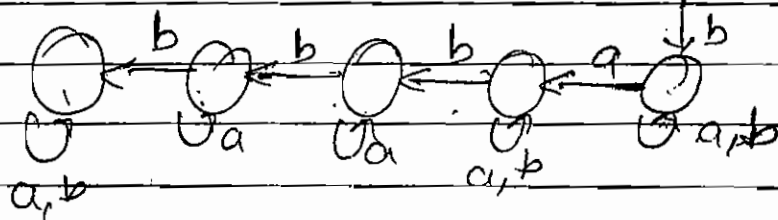
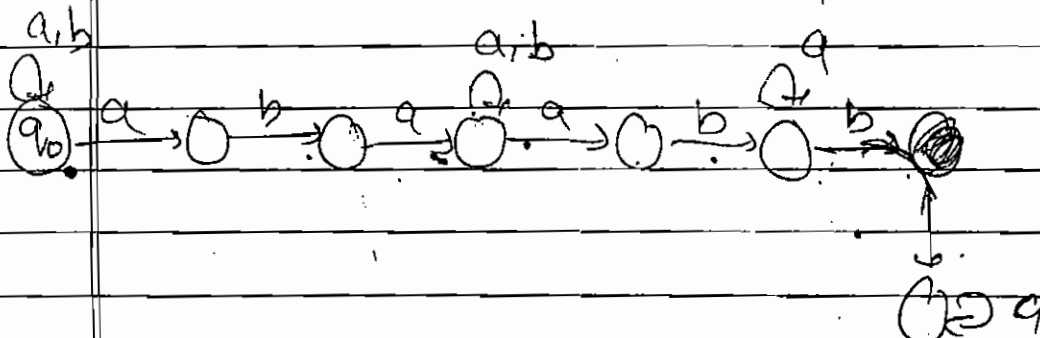


→ $(a+b)^* ab(a+b)^* ab^* a(a+b)^*$



5 states

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

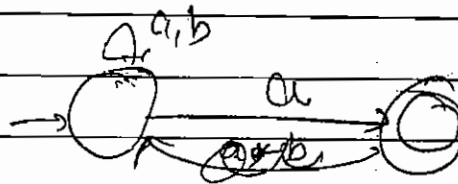


= 12

$(a+b)^*$

→ No. of state = No. of individual symbol + 1

→ $(a+b)^* a$

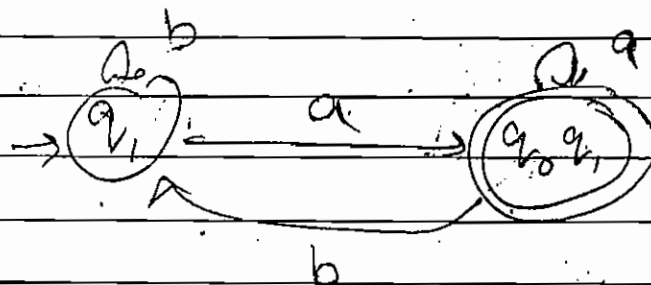


	a	b
q1	(q1, q1) q1	
q2	⊘	⊘

DFA

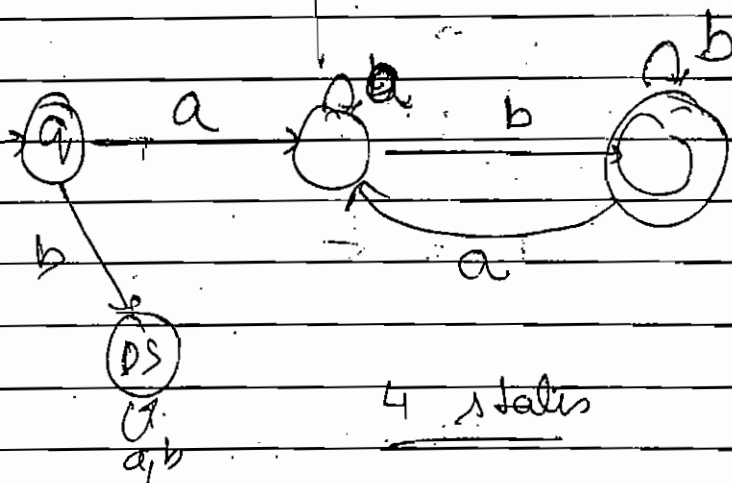
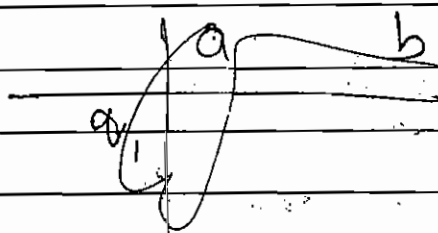
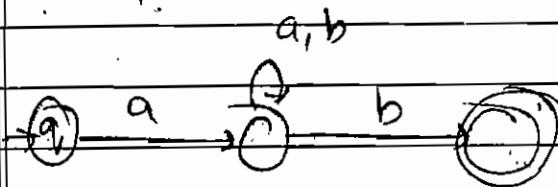
	a	b
q1	(q1, q2) q1	

(q1, q2) q2	(q1, q2)	(q1, q2)
-------------	----------	----------

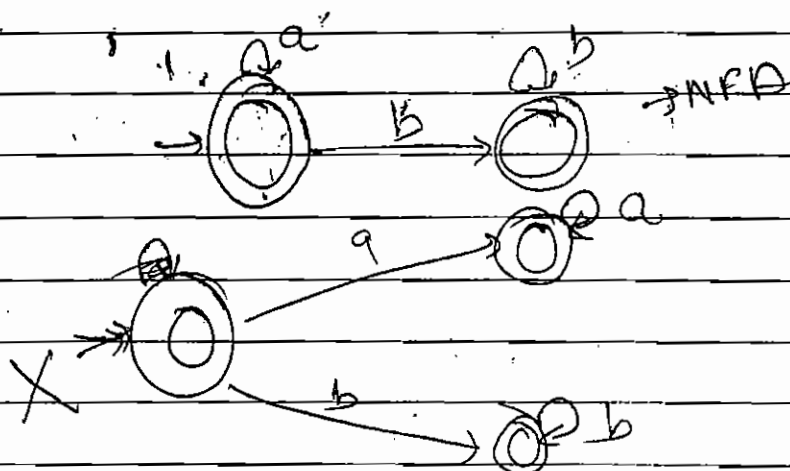


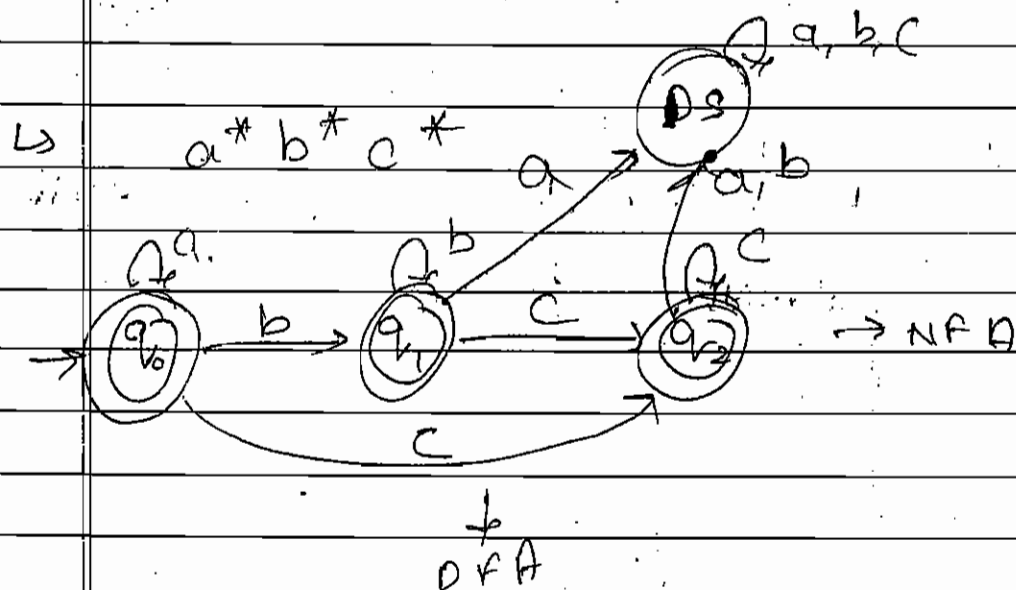
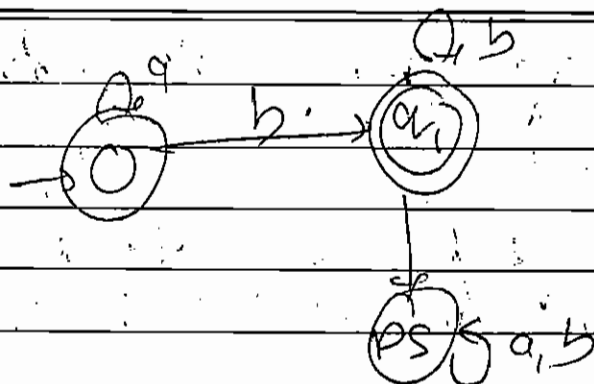
2 states

→ $a(a+b)^*b$



→ a^*b^* $\emptyset, a, b$





↳ $a^* b^* c^* d^*$

in NFA

Note: No. of state = No. of symbols

NOS in DFA = No. of symbols + 1

Conversing many states into one:

Equivalent States:-

Two state (Q_1, Q_2) are said to be equivalent $\forall x \in \Sigma^*$ both Q_1, Q_2 has to go to either final or -

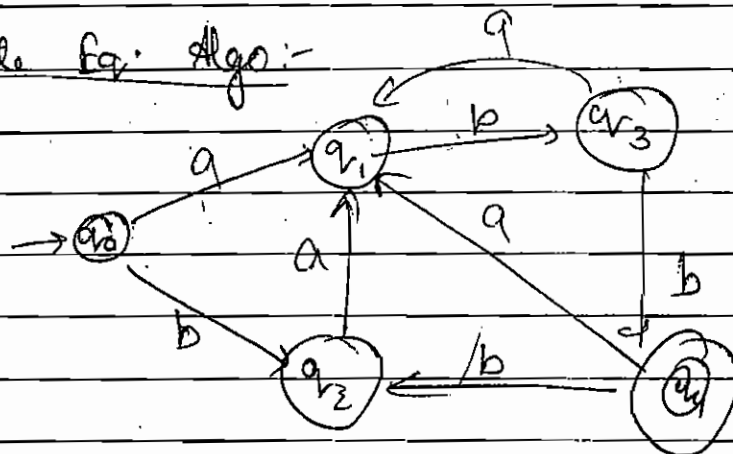
non-final. But it is difficult to proof for infinite no. of alphabets.

hence we have to prove equivalent algorithms for identifying equivalent state.

① State Equivalence algo.

② Table Filling algo.

State Eq. Algo:-



me!

2

2

me

ut

alphabet

ivalent

ivalent

	a	b
q_0	q_1	q_2
q_1	q_1	q_3
q_2	q_1	q_4
q_3	q_1	q_4
q_4	q_1	q_2

0. Equivalence - $\{q_0, q_1, q_2, q_3\} \{q_4\}$

1. Equivalence = $\{q_0, q_1, q_2\} \{q_3\} \{q_4\}$

2. Equivalence = $\{q_0, q_2\} \{q_1\} \{q_3\} \{q_4\}$

3. Equivalence = $\{q_0, q_2\} \{q_1\} \{q_3\} \{q_4\}$

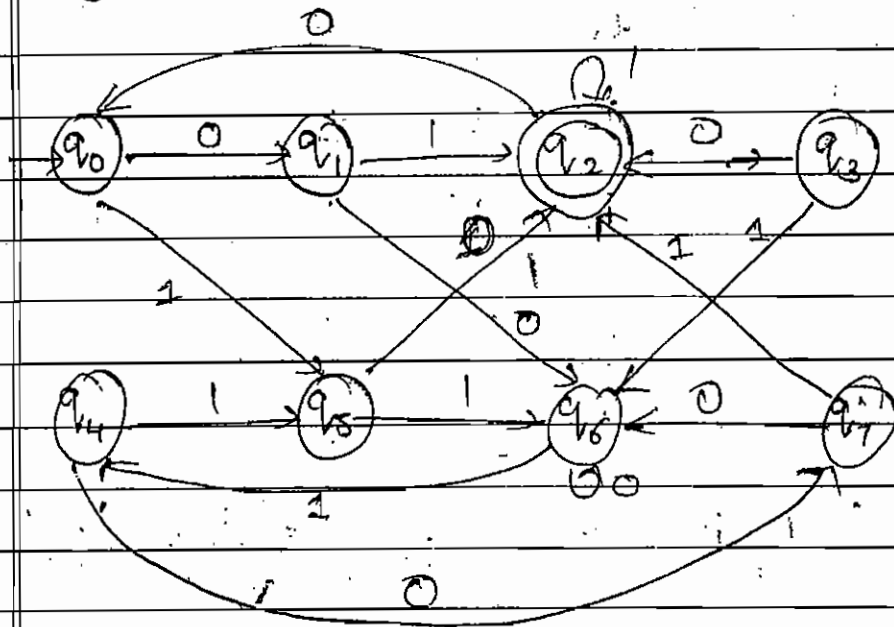
4. Equivalence =

4 states

	a	b
$[q_0, q_2]$	$[q_1]$	$[q_0, q_2]$
q_1	q_1	q_3
q_3	q_1	q_4
q_4	q_1	q_0, q_2

→ direct equivalence.

→ Construct minimal DFA for following DFA.



	0	1
q ₀	q ₁	q ₅
q ₁	q ₀	q ₂
q ₂	q ₀	q ₂
q ₃	q ₂	q ₆
q ₄	q ₇	q ₅
q ₅	q ₂	q ₆
q ₆	q ₃	q ₄
q ₇	q ₄	q ₂

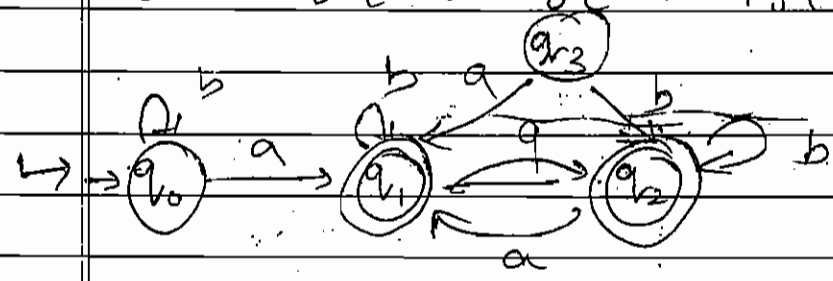
following

$$O = \text{Equi} = \{q_0, q_1, q_5, q_6, q_4, q_7, q_2\}$$

$$= \{q_0, q_1, q_5, q_6, q_7\} \cup \{q_2\}$$

$$\{q_3, q_5\} \cup \{q_1, q_7\} \cup \{q_0\} \cup \{q_2\} \cup \{q_4\} \cup \{q_6\}$$

$$\{q_1, q_7\} \cup \{q_3, q_5\} \cup \{q_0, q_4\} \cup \{q_6\} \cup \{q_2\}$$



① How many no. of states in minimal states DFA?

- Ⓐ a) 2 b) 3 c) 4 d) none

②

$$O = \text{Eq.} \{q_0, q_1, q_2\} - q_0$$

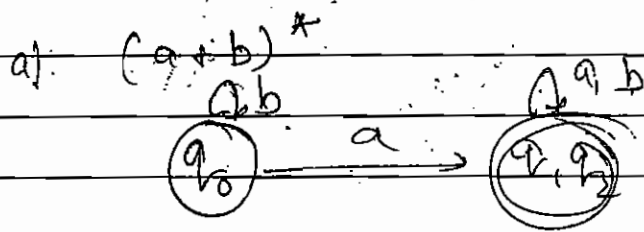
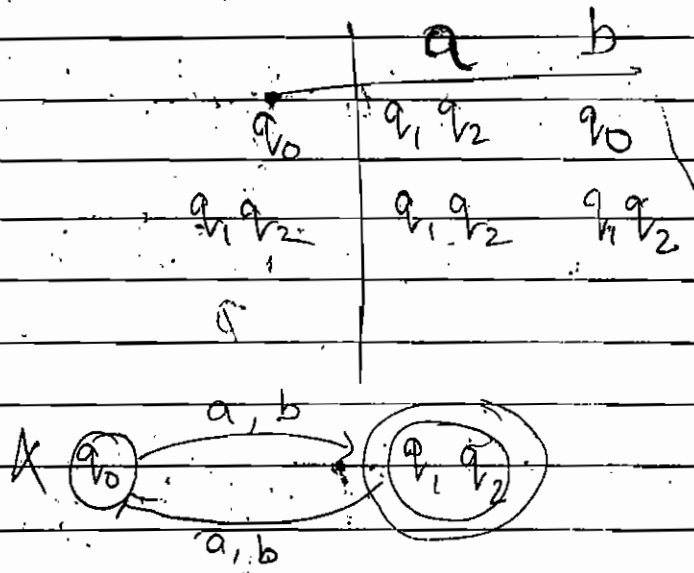
$$\{q_2, q_3\}$$

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

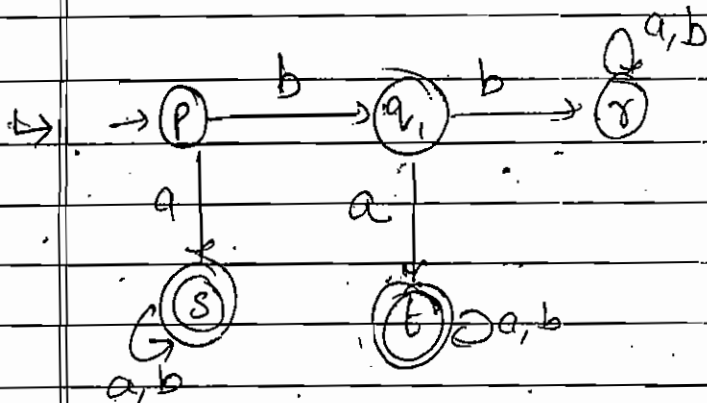
$$\{q_0\} \cup \{q_1, q_2\}$$

	a	b
q ₁	q ₁	q ₀
q ₂	q ₂	q ₁
q ₃	q ₁	q ₂
	q ₁	q ₂

② Which lang. it accepting...



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

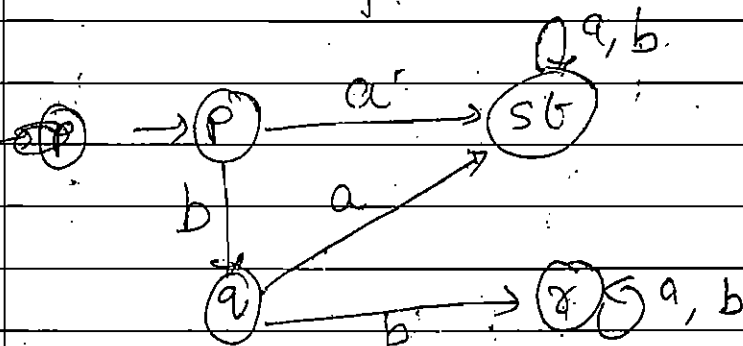


$$Q = \{p, q_1, r, s, t\}$$

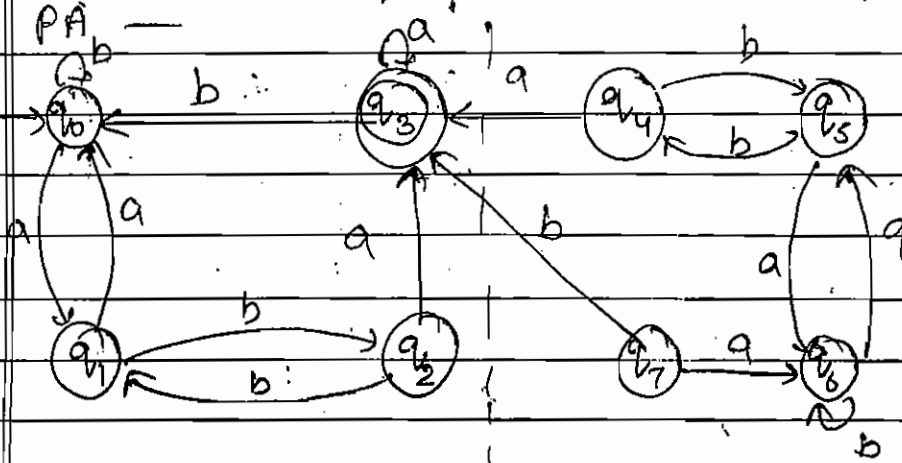
	a	b
p	s	q
q	t	r
r	r	r
s	s	s
t	t	t

$\{p\} \quad \{q\} \quad \{r\} \quad \{s, t\}$

	a	b
p	st	q
st	st	st
q	st	r
r	r	r



→ Construct an equi. minimal DFA for following



$$\text{Equi} = \{q_0, q_1, q_2, q_4, q_5, q_6, q_7\} \{q_3\}$$

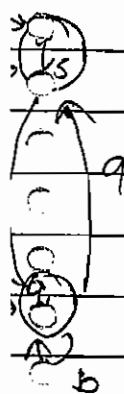
	a	b
q_0	q_1	q_0
q_1	q_0	q_2
q_2	q_3	q_1
q_3	q_3	q_0
q_4	q_3	q_5
q_5	q_6	q_4
q_6	q_5	q_6
q_7	q_6	q_3

ex following

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

$\{q_2\} \{q_4\} \{q_3\}$

	a	b
q_0	q_1	q_0
q_1	q_0	q_2
q_2	a	



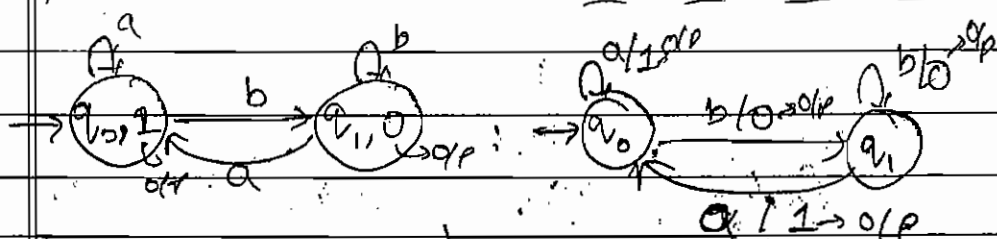
F/A with O/P :-

Moore M/C:

It is a mathematical model in which o/p is associated with state.

Mealy M/C:

It is a mathematical model in which o/p is associated with transition.



$(Q, \Sigma, \delta, q_0, A, \lambda)$

$\Rightarrow$ finite no. of states

$\Sigma \rightarrow$ i/p alphabet

$\delta : T.F. Q \times \Sigma \rightarrow Q$

q_0 : initial state

Δ : o/p alphabet

λ : o/p func.

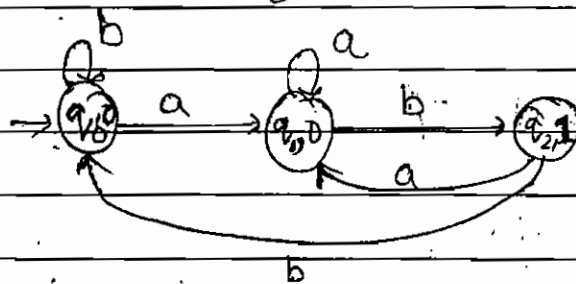
$\lambda : Q \rightarrow \Delta$

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

→ Moore & Mealy M/C's are DFA.

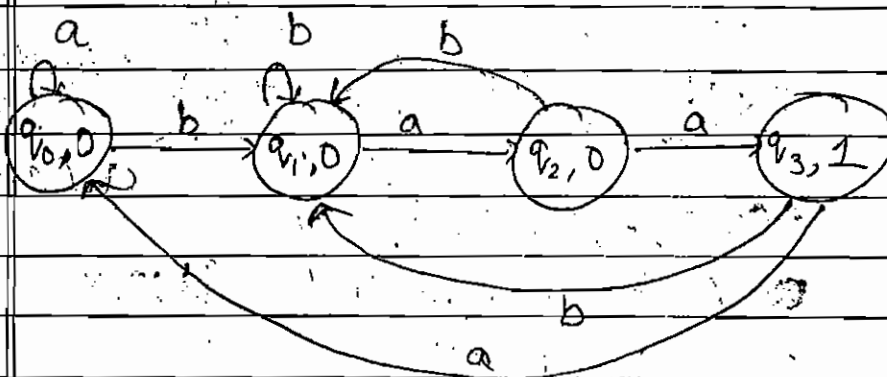
↳ Construct ^{Moore} ~~Mealy~~ M/C that takes all strings of a's & b's as i/p and counts no. of occurrences of substring 'ab' for given string.

$$\Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$



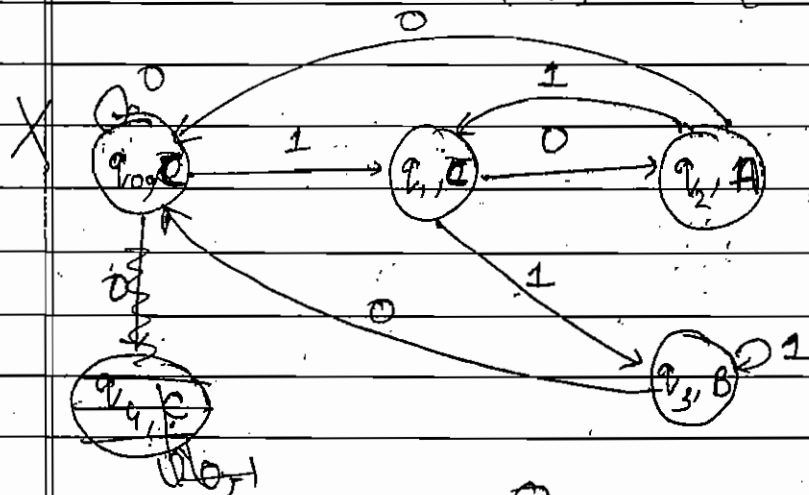
↳ Construct Moore M/C that counts no. of occ. of substring 'baa' over the given string.

$$\Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$

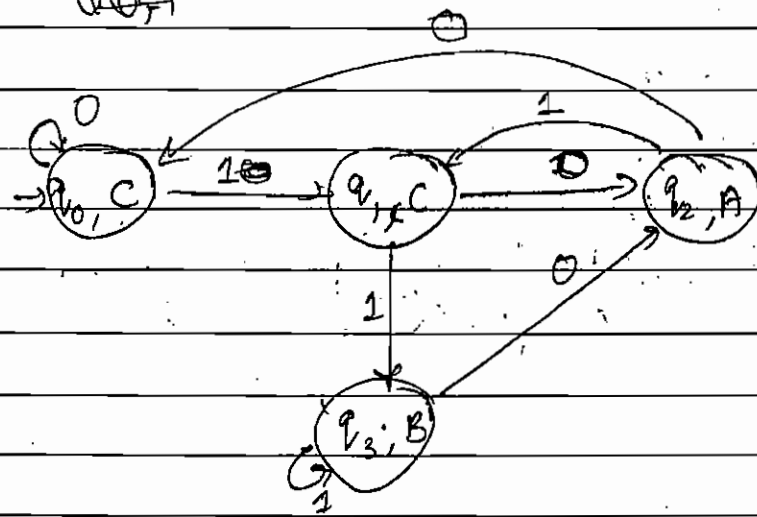


all strings
m of
given

→ that takes all strings of 0s & 1s as i/p
and produces A as o/p if i/p ends
with 10 or produces B as o/p if i/p
ends with 11 otherwise produces
C as o/p. $\Sigma = \{0, 1\}$ $\Delta = \{A, B, C\}$



counts
ex the



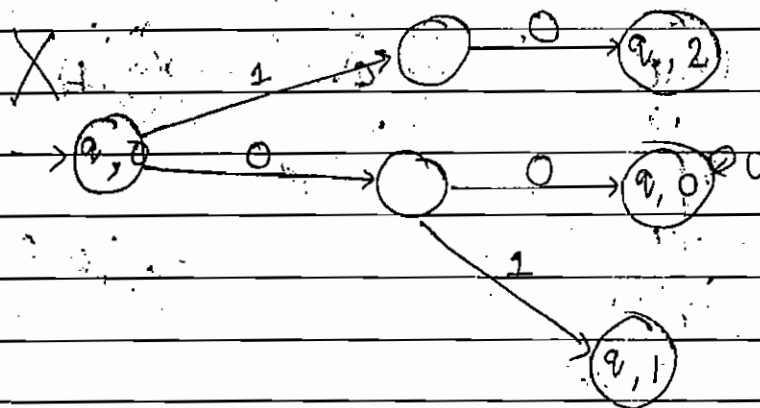
1

↳ that takes all binary nos. as i/p and produces Residue modulo 3 as o/p.

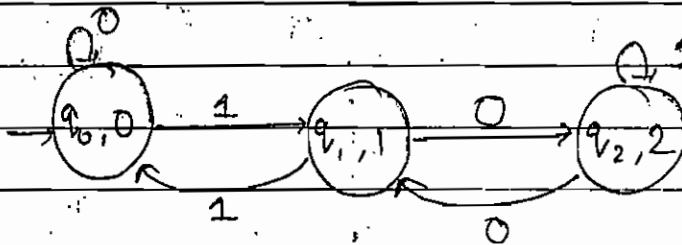
remainder

00, 01, 10

$$\Sigma = \{0, 1\} \quad \Delta = \{0, 1, 2\}$$



0011



↳ Takes all base 3 nos. as i/p and produces Residue modulo 4 as o/p.

$$\Sigma = \{0, 1, 2\}$$

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

$$0 \rightarrow 0$$

$$1 \rightarrow 1$$

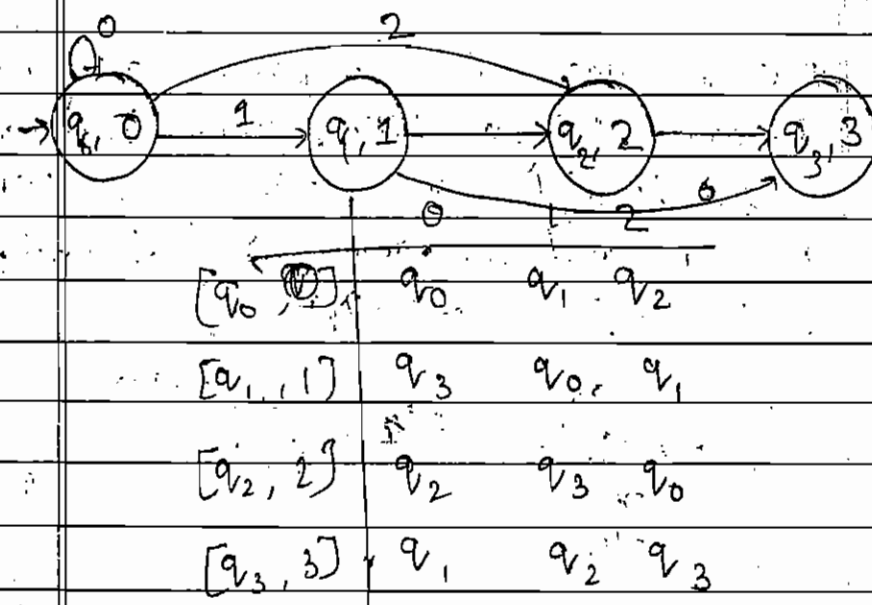
$$2 \rightarrow 2$$

$$3 \rightarrow 0$$

$$4 \rightarrow 1$$

$$5 \rightarrow 2$$

10 and



~~q0~~

Note The moore m/c that takes all base m no. as i/p and produces residue modulo n as o/p requires 'n' no. of states.

↳ Construct Mealy m/c that produces 1's complement of given binary string as o/p.

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

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

o/p produces

0 → 0

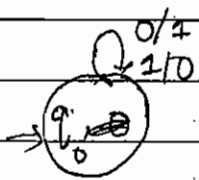
1 → 1

2 → 2

3 → 3

4 → 4

5 → 5

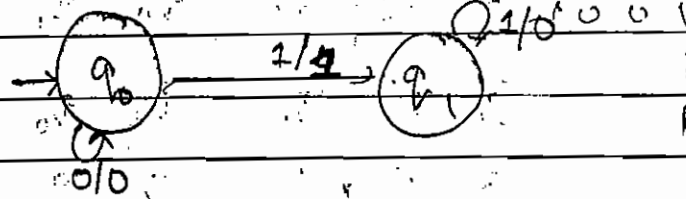


2005

↳ that produces 2's complement of given binary string as O/P. Assume that we are ~~Σ = {0,1}~~ reading string from LSB to MSB, and end carry is discard.

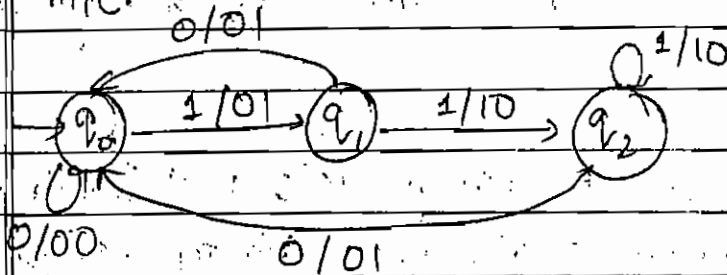
$$\Sigma = \{0,1\}$$

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



Logic: from MSB LSB to MSB 0 will be 0 as it is, until first 1 comes. after 1 it will be like 1's complement.

↳ What is O/P produced by follow state m/c.



a) 11 → 01

b) 10 → 00

c) sum of a & b present & previous bit.

d) NOT

$$01 \Rightarrow 01$$

$$10 \Rightarrow 01$$

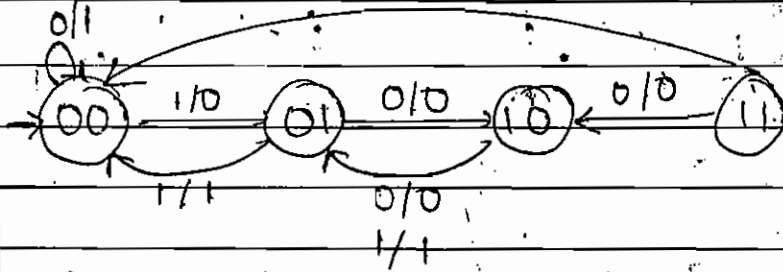
$$11 \Rightarrow 10$$

$$00 \Rightarrow 00$$

given
we are
LSB
rd.

2009

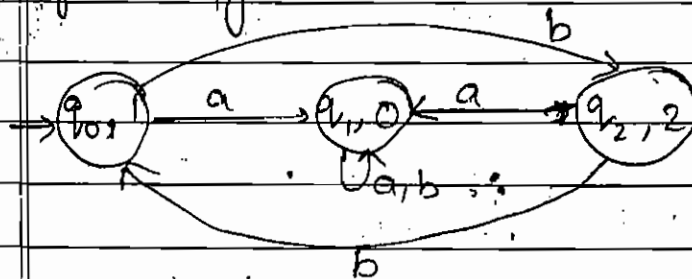
L



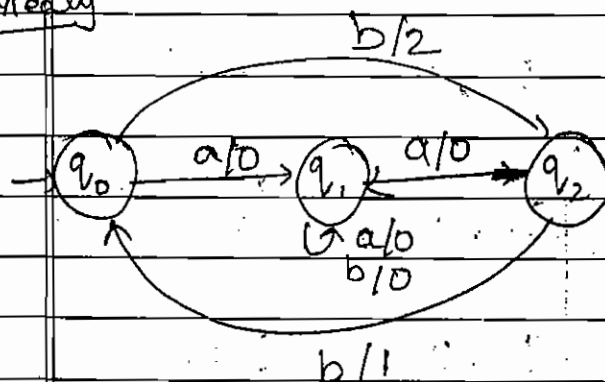
By reading 101, it produces 1 as O/P,
so min^m. length = 3:

Mealy to Moore to Mealy M/C:

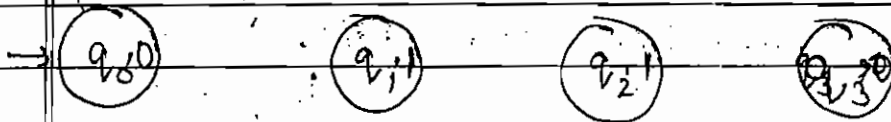
L Construct an equi mealy M/C for
following Moore M/C.



Mealy



t_s	Present State	$i=0$ N.S.	$i=1$ N.S.	O/P
	q_0	q_3	q_2	0
	q_1	q_1	q_1	1
	q_2	q_2	q_2	1
	q_3	q_0	q_3	0

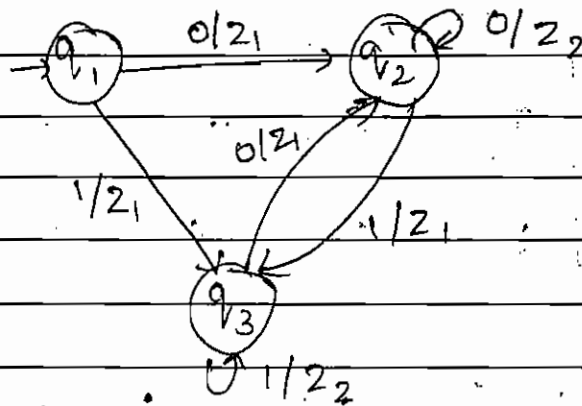


Mealy

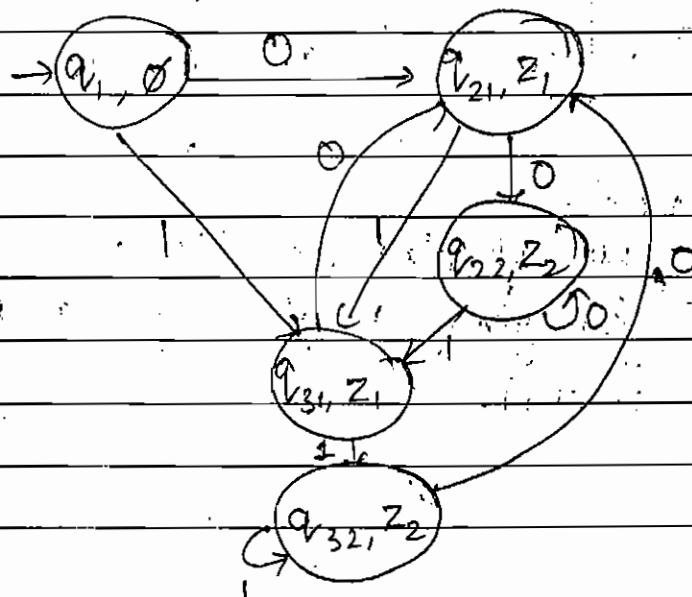
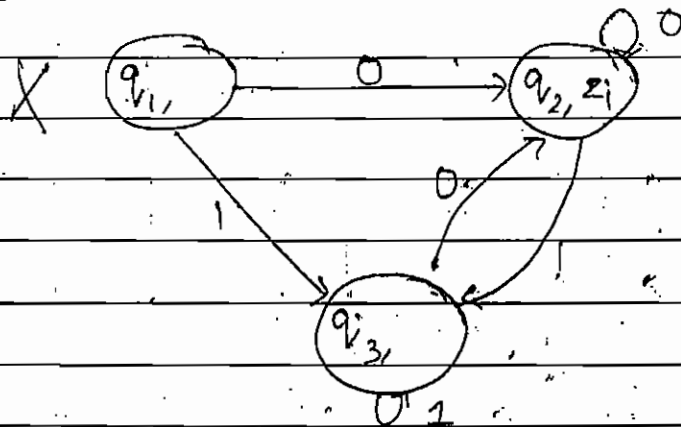
P.S.	$i=0$ N.S. O/P	$i=1$ N.S. O/P
q_0	$q_3, 0$	$q_2, 1$
q_1	$q_1, 1$	$q_1, 1$
q_2	$q_2, 1$	$q_2, 1$
q_3	$q_0, 0$	$q_3, 0$

Mealy $\Rightarrow$ Moore

$\rightarrow$ Construct equ. Moore for Mealy -



Moore:



→

PS	C=0		C=1	
	NS	O/P	NS	O/P
q_1	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

PS	C=0		C=1	
	NS	O/P	NS	O/P
q_1	q_3	1	q_{20}	1
q_{20}	q_1	0	q_{40}	0
q_{21}	q_1	1	q_{40}	1
q_3	q_{21}	0	q_1	0
q_{40}	q_{41}	0	q_3	0
q_{41}	q_{41}	1	q_3	1

max^m

Note: The no. of states possible in a Moore M/C for given n state, m O/P Mealy M/C is $(n \times m)$.

→ How many Mealy M/Cs are possible with two states x & y for the i/p alphabet $\{a, b\}$, o/p alpha $\{0, 1\}$, where x is always initial state.

	i = a o/p	i = b o/p
x	$2 \times 2 = 4$	$2 \times 2 = 4$
y	$2 \times 2 = 4$	$2 \times 2 = 4$
	$\Rightarrow 4^4 = 256$	

→ How many Moore M/C are possible for above ques?

	i = 0	i = 1	O/P
x	2	2	3 → may be 0 also
y	2	2	2
	$= 2^5 \times 3 = 64 \times 3 = 192$		

Moore
Mealy M/C

Regular Languages:-

↳ which of the following lang. are regular and which are non-regular?

① $L = \{a^n b^n \mid n \leq 1\}$

② $L = \{a^n b^n c^n \mid n \leq 1\}$

③ $L = \{a^n b^m \mid n, m \leq 1\}$
 $n < m$

④ $L = \{a^i b^j \mid i, j \leq 1\}$
 $i > j$

⑤ $L = \{a^n b^m c^k \mid n, m, k \geq 1\}$

⑥ $L = \{a^i b^{2j} \mid i, j \geq 1\}$

⑦ $L = \{a^n b^{4m} \mid n > m\}$

⑧ $L = \{a^i b^j \mid i \neq j\}$

⑨ $L = \{w \in \{a, b\}^*\}$

⑩ $L = \{ww^R \mid w \in \{a, b\}^*\}$

$w^R \rightarrow$ reverse.

Regular

$$(10) L = \{ x.y \mid x, y \in \{0,1\}^* \}$$

$$(11) L = \{ x x^R \mid x \in \{0,1\}^*, |x| = 2 \}$$

↪ length of string.

$$(12) L = \{ x x \mid x \in \{a,b\}^* \}$$

$$(13) L = \{ w \mid w \in \{a,b\}^*, |w| \geq 10 \}$$

$$(14) L = \{ w \mid w \in \{a,b\}^*, |w| \bmod 15 = 0 \}$$

$$(15) L = \{ a^{2^n} \mid n \geq 1 \}$$

$$(16) L = \{ a^{2^n} \mid n \leq 1 \}$$

$$(17) L = \{ a^{2^n} \mid n \geq 1 \}$$

$$(18) L = \{ a^{2^n} \mid n \geq 1 \}$$

$$(19) L = \{ a^{n^2} \mid n \geq 1 \}$$

$$(20) L = \{ a^p \mid p \text{ is a prime no.} \}$$

$$21) L = \{ a^k \mid k \text{ is a odd no.} \}$$

$$22) L = \{ a^m \mid m \text{ is an even no.} \}$$

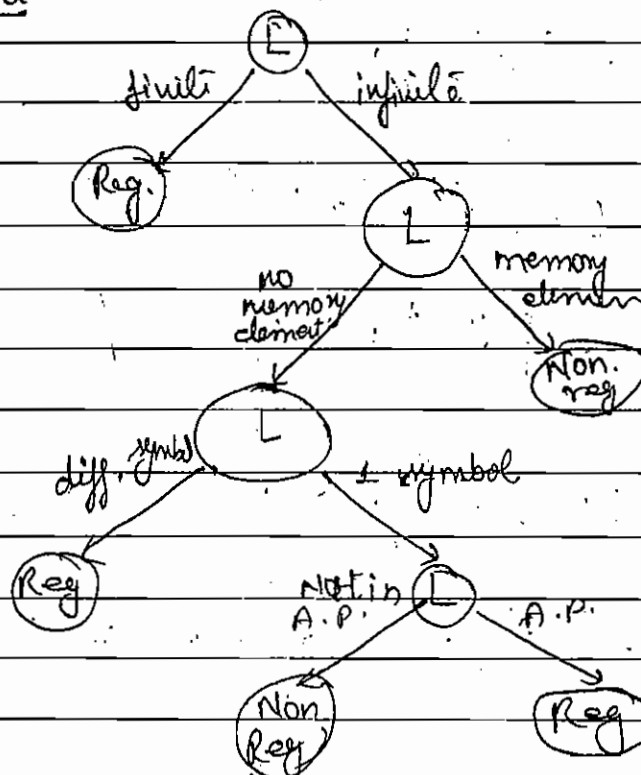
$$23) L = \{ \text{total population of INDIA} \}$$

$$24) L = \{ \text{Property of Indian Cricketers} \}$$

$$25) L = \{ \text{Total no. of java programs written in the year 2006} \}$$

$$26) L = \{ a^i b^j \mid i, j \geq 1 \}$$

Method:



→ Powers should be in A.P. for regularity.

$$27) L = \{a^m b^k c^l \mid m = k + l\}$$

$$28) L = \{a^i b^j c^k \mid i > j > k\}$$

$$29) L = \{x \mid x \in \{a, b\}^* \text{ and } n_a(x) > n_b(x)\}$$

$$30) L = \{x \mid x \in \{0, 1\}^* \text{ and } n_0(x) = 2n_1(x)\}$$

$$31) L = \{a^i b^j \mid i \neq j\}$$

$$32) L = \{a^2, a^5, a^8, \dots\}$$

$$33) L = \{0^i \mid i \leq 1\}$$

$$34) L = \{a^m b^{n+m} c^n \mid m, n \geq 1\}$$

$$35) L = \{x \mid x \in \{0, 1\}^* \text{ and } x \text{ has equal no. of 0s \& 1s}\}$$

$$36) L = \{x \mid x \in \{a, b\}^* \text{ and } |x| \equiv 3 \pmod{7}\}$$

$$37) L = \{ \cancel{w} w w w w w w^R \mid w \in \{0,1\}^*, |w| = 1 \}$$

$$38) L = \{ a^n b^n c^n d^n \mid n \geq 1 \}$$

$$39) L = \{ a^{2n+1} \mid n \geq 1 \}$$

Closure Properties of Regular Languages:-

① Union operation

Union of two regular lang. is always regular hence R.L. are closed under union operation.

$$R_1 \cup R_2 = R_3$$

$$\{ a^n b^m \mid n, m \geq 0 \} \cup \{ c^n d^m \} \Rightarrow$$

$\downarrow$
 R_1
 $\downarrow$
 R_2
 $\downarrow$
 R

② Concatenation Op:-

Conc. of two R.L. is always regular hence R.L. are closed under conc. operation.

$$R_1 \cdot R_2 = R_3$$

$$\{ a^n \} \{ b^m \}$$

③ Kleen Closure

Kleen closure of R.L. is always regular closure hence R.L. are closed under Kleen closure op.

$$L_1 \Rightarrow \gamma_1$$

$$L_1^* \Rightarrow \gamma_1^*$$

④ Positive Closure

the closure of R.L. is always regular hence R.L. are closed under the closure.

$$L_1 \Rightarrow \gamma$$

$$L_1^+ \Rightarrow \gamma^+$$

⑤ Complement Op:-

Complement of R.L. is always regular hence regular R.L. are closed under complement op.

$$L_1 \Rightarrow \text{Reg.}$$

$$\bar{L}_1 \Rightarrow \Sigma^* - L_1 \Rightarrow \text{Reg.}$$

⑥ Intersection Op:-

$$L_1 : \text{Reg.}$$

$$L_2 : \text{Reg.}$$

$$\bar{L}_1$$

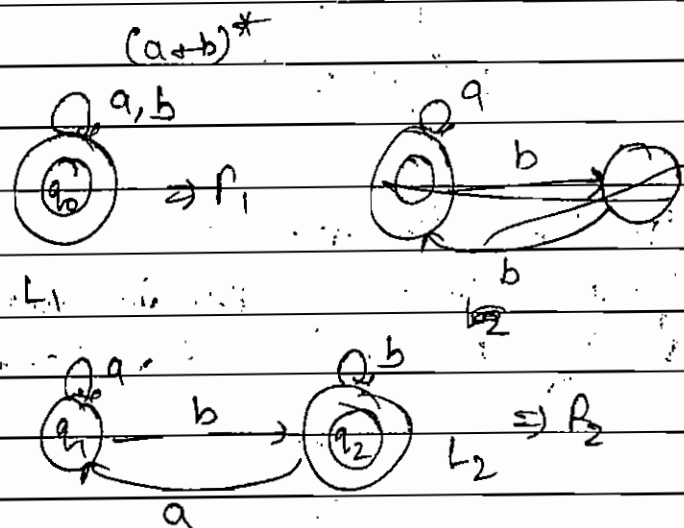
$$\bar{L}_2$$

$$\bar{L}_1 \cup \bar{L}_2 \Rightarrow \overline{L_1 \cap L_2}$$

$$\Rightarrow L_1 \cap L_2$$

R.L. are closed under Intersection op.

Q How many no. of states there in min.
 DFA that accept $L_1 \cap L_2$ where L_1 is
 $(a+b)^*$ & L_2 is $(a+b)^*b$



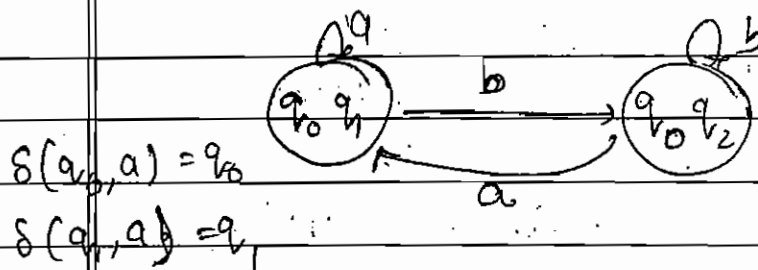
$$L_1 \cap L_2 = F_1 \times F_2 = \{Q, \Sigma, \delta, q_0, f\}$$

$$Q = \{q_0\} \times \{q_1, q_2\} = \{q_0q_1\} \cup \{q_0q_2\}$$

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

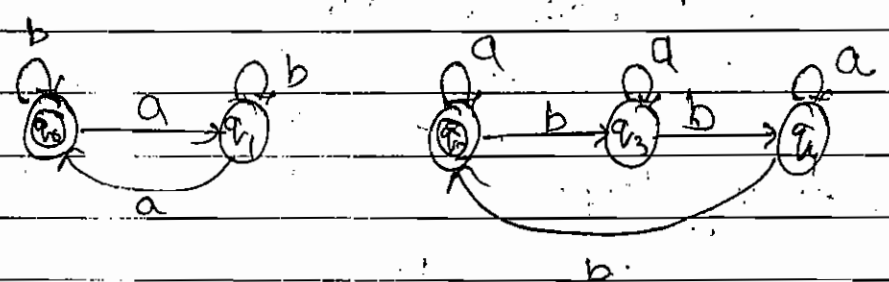
$$q_0 = \{q_0q_1\}$$

$$q_f = \{q_0q_2\}$$



Ques.

Construct DFA that accepts all strings of
as & bs where no. of as are divisible
by 2 & no. of bs are divisible by 3
in every string.



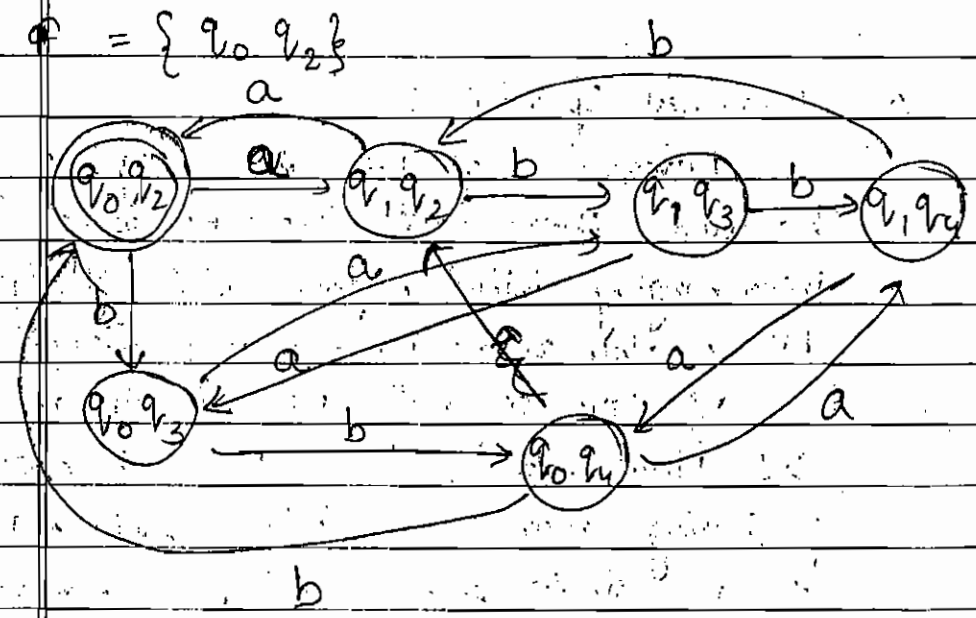
$$Q = \{q_0, q_1\} \times \{q_2, q_3, q_4\}$$

$$= \{q_0, q_2\} \{q_0, q_3\} \{q_0, q_4\} \{q_1, q_2\} \{q_1, q_3\} \{q_1, q_4\}$$

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

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

$$q = \{q_0, q_2\}$$



↳ How many no. of states in min^m DFA $\Sigma = \{a, b\}$ where no. of a 's are divisible by 6 & b 's are divisible 8.

$$48 \text{ states} = 6 \times 8$$

$$\begin{array}{r} 2 \overline{) 16, 8} \\ 2 \overline{) 3, 4} \\ 2 \overline{) 1, 2} \\ 5 \end{array}$$

⑦ Diff. O/P

$$\begin{array}{cc} L_1 & L_2 \\ R L & R L \end{array}$$

$$L_1 - L_2 = (L_1, \overline{L_2})$$

Let L_1 & L_2 are R.L. & diff. of L_1 & L_2 is always reg., Hence reg. R.L. are closed under diff. operation.

⑧ Reversal Operation:

Reversal of R.L. is always regular because there exist some FA which is interchanging initial & final states of original FA & also reversing the edge directions.

If the original FA contain more > 1 Final state, then there is chance of having more than 1 initial state. Make it as single initial state, by taking new initial state with ϵ transition.

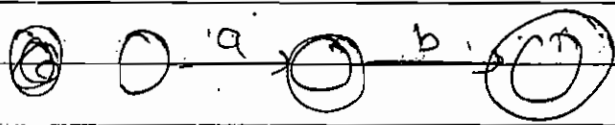
DFA

exists

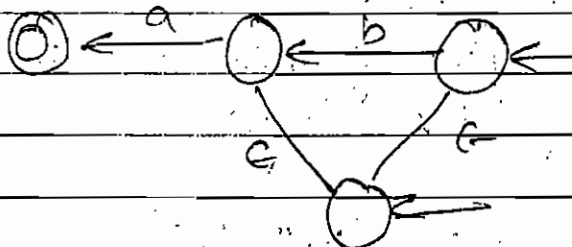
$$L = \{a^n b^m\}$$

$$L^R = \{b^m a^n\}$$

ab



ab^R



L^R is closed

⑨ Subset Operation :-

Subset of R.L. is may or may not regular hence R.L. are not closed under subset operation.

Regular

is

original

operations

more

chance

Make

easy

$$a^* b^* \subseteq (a+b)^* \quad \{a^n b^n \mid n \geq 1\} \subseteq a^* b^*$$

$$\{a^n b^m \mid n, m \geq 1\} \subseteq (ab)^* \quad \{a^{2^n} \mid n \geq 1\} \subseteq a^*$$

$\downarrow$
R.L.

$$\{a^n b^n \mid n \geq m\} \subseteq (a+b)^*$$

$\downarrow$
Non R.L.

⑩ Infinite Union & Infinite Intersection.

R.L. are not closed under above two operations. Because

$$L_1 \cup L_2 \cup L_3 \dots \cup L_\infty \Rightarrow \text{Non Reg.}$$

$$L_1 \cap L_2 \cap L_3 \dots \cap L_\infty \Rightarrow \text{Non Reg.}$$

Grammar :-

Set of rules use to describe strings of the lang.

$$G = (V, T, P, S)$$

$V \rightarrow$ Set of variables or Non terminals

$T \rightarrow$ set of terminal

$P \Rightarrow$ No. of production

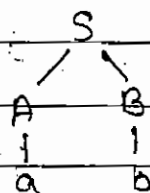
$S \rightarrow$ Starting Symbol

$$S \rightarrow AB$$

$$A \rightarrow a$$

$$B \rightarrow b$$

$\Rightarrow$



Parse Tree:

Tree representation of derivation.

→ Identify the lang. represented by following grammar.

$$S \rightarrow OS | I / \epsilon$$

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

be strings

$$S \rightarrow OS | S / ISOS / \epsilon$$

$$OISOS | ISOS$$

$$OIOIIO$$

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

$$\epsilon, 01, 10, 1100, \dots$$

$$S \rightarrow OS || / 0 ||$$

$$00011111$$

$$L = \{0^n 1^{2n} \mid n \geq 1\}$$

is.

$$S \rightarrow aSa / bsb / \epsilon$$

$$L = \{\epsilon, aa, bb, abba, baab, abaaba, \dots\}$$

Palindrom.

→ Find the grammar that generate all string of a's & b's including ϵ .

$(a+ b)^*$

$S \rightarrow a s a / b s b / a s b / b s a / \epsilon$ X

$S \rightarrow a s / b s / a / b / \epsilon$

↳ Each string starting with a ending with b.

$S \rightarrow a A b$

$A \rightarrow a A / b A / \epsilon$

↳ Each string starting & ending with diff symbol.

$S \rightarrow a A b / b A a$

$A \rightarrow a A / b A / \epsilon$

↳ Each string starts & ends with same symbol

$S \rightarrow a A a / b A b / a / b$

$A \rightarrow a A / b A / \epsilon$

↳ length of string is exactly 3.

X $S \rightarrow a A b / b A a /$

X $A \rightarrow a / b / \epsilon$

$S \rightarrow A A A$

$A \rightarrow a / b$

↳ At least 3

$$S \rightarrow AAA|B$$

$$A \rightarrow a|b$$

$$B \rightarrow aB|bB|E$$

↳ At most 3.

$$S \rightarrow A|AA|AAA$$

$$A \rightarrow a|b$$

$$S \rightarrow AAA$$

$$A \rightarrow a|b|E$$

↳ L is divisible by 3.

$$S \rightarrow AAAS|E$$

$$S \rightarrow a|b$$

$$S \rightarrow AS|E$$

$$A \rightarrow BBB$$

$$B \rightarrow a|b$$

↳

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

$$S \rightarrow asb|ab$$

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

$$S \rightarrow a s_1 b s_2 c$$

$$s_1 \rightarrow a s$$

$$S \rightarrow AB$$

$$A \rightarrow aAb|ab$$

$$B \rightarrow cB|c$$

$$\rightarrow L = \{a^n b^n c^m d^m / n, m \geq 1\}$$

$$S \rightarrow AB$$

$$A \rightarrow aAb / ab$$

$$B \rightarrow cBd / cd$$

$$\rightarrow L = \{a^n b^m c^m d^n / n, m \geq 1\}$$

$$S \rightarrow A$$

$$A \rightarrow aBd / \text{ad}$$

$$B \rightarrow bBc / bc$$

$$abBcd$$

$$a$$

$$S \rightarrow aAd / asd$$

$$A \rightarrow bAc / bc$$

$$\rightarrow L = \{a^m b^{m+n} c^n / m, n \geq 1\}$$

$$a^m b^m b^n c^n$$

$$S$$

$$S \rightarrow AB$$

$$A \rightarrow aAb / ab$$

$$B \rightarrow bBc / bc$$

$$\rightarrow L = \{a^{m+n} b^n c^m / n, m \geq 1\}$$

$$a^m a^n b^n c^m$$

$$S \Rightarrow aAc / aSc$$

$$A \Rightarrow aAb / ab$$

$$\hookrightarrow S \rightarrow aas/sas/sbs/sab/asa/abs/a/b$$

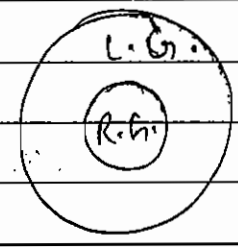
identify total no. of strings of length 4, given is -

- a) 3 b) 5 c) 8 d) None.

Type of Grammars

① Reg. lan. (3)

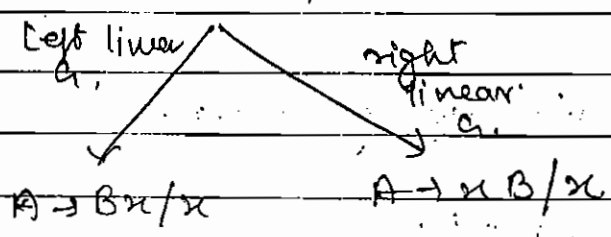
$\Downarrow$
R.G.
Type G.



$$A \rightarrow xB/Bx/x$$

$$x \in \Sigma^*$$

$$A, B \in V$$



Note: L.H.S. exactly 1 variable, R.H.S. atmost 1 Var.
Linear G. - Same, but Linear Gm. need not to be R.G.

② CFL (2)
 $\Downarrow$
CFG
2G

$$A \rightarrow \alpha$$

$$\alpha \in (V+T)^*$$

③ CSL (1)

$\Downarrow$
 CSG

$\uparrow$

$\alpha \rightarrow \beta \quad \alpha, \beta \in (V \cup T)^+$
 $|\alpha| \leq |\beta|$

NOTE

④ R.E.L. (0)

$\Downarrow$

Unrestricted G.

Type 0

$\alpha \rightarrow \beta$
 $\alpha \in (V \cup T)^+$
 $\beta \in (V \cup T)^*$

$\rightarrow$ Ty. Tell the type of grammar

1) $S \rightarrow ABC/E$

$A \rightarrow BC/b$

$B \rightarrow AB/b$

$C \rightarrow a$

An C.F.G.

$$\begin{aligned} 2) \quad & S \rightarrow ABC / \epsilon \\ & aA \rightarrow BC / bb \\ & bB \rightarrow aBa / ba \\ & acb \rightarrow abab \end{aligned}$$

Note: S derives ϵ in a valid csa production, in that case S should not be presented on RHS part of any other grammar productions.

↳ Identify type of grammar, possible type no & highest type no, satisfied by given grammar -

$$S \rightarrow ABC$$

$$A \rightarrow ab$$

$$B \rightarrow b$$

$$C \rightarrow ac / A$$

Ans: CFG, possible type 2, 1, 0, highest 2.

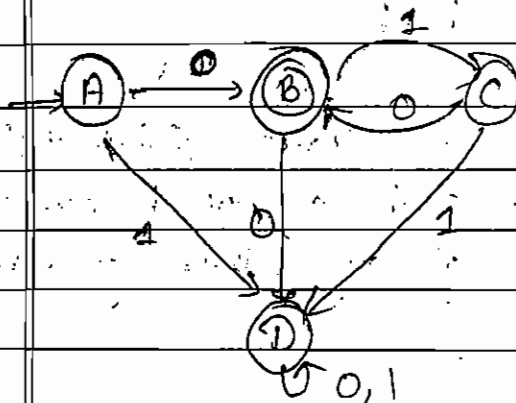
Linear Grammar:

In any grammar if we find exactly one variable on LHS and atmost 1 variable on RHS, or

All RG are linear, all L.G. need not to be RG.

FA $\rightarrow$ RG

$\rightarrow$ Construct equivalent RG for following FA!



$G = (V, T, P, S)$

$V =$ no. of states

$T = \{0, 1\}$

P

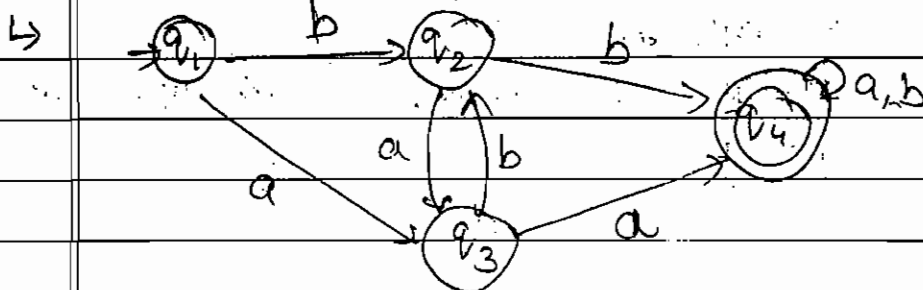
$S = A$

$A \rightarrow 0B / 1D / 0$

$B \rightarrow 0D / 1C$

$C \rightarrow 0B / 1D / 0$

$D \rightarrow 0D / 1D$



$$q_1 \rightarrow a q_3 / b q_2$$

$$q_2 \rightarrow a q_3 / b q_4 / b$$

$$q_3 \rightarrow a q_4 / b q_2 / a$$

$$q_4 \rightarrow a q_4 / b q_4 / a / b$$

→ How many max^m no. of productions are possible in regular grammar equivalent to given n state DFA, over the i/p alphabet $\{a, b, c\}$, where q_1 is always initial state.

$$V = n$$

$$T = \{a, b, c\}$$

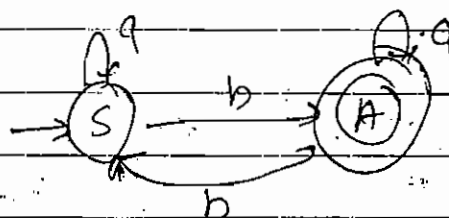
$$S = q_1$$

Ans: $6n + 1$ (← also because initial state is final state also)

→ No. of states for following R.G. —

$$S \rightarrow aS / bA / b$$

$$A \rightarrow aA / bS / a$$



In RHS a or b are present so we can create P.A.

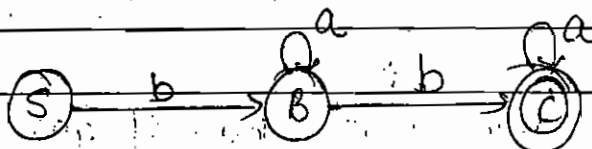
→

$S \rightarrow bB$

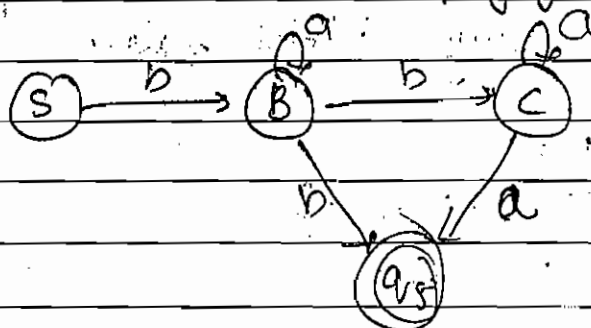
$B \rightarrow bC \mid aB \mid b$

$C \rightarrow AC \mid a$

DFA not possible



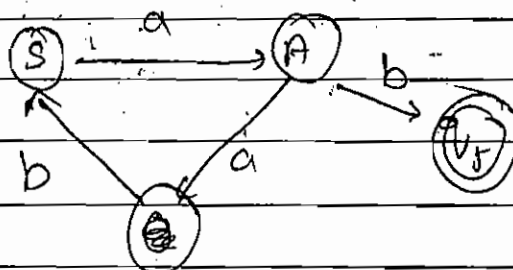
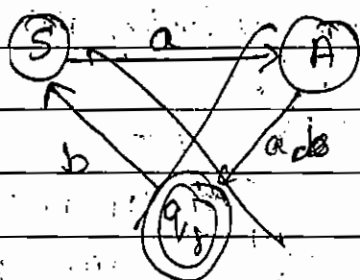
Note: while converting to NFA always take a new state. True only for Right Linear G.



→

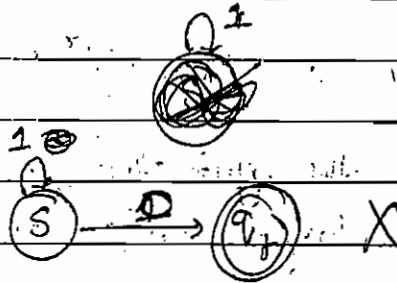
$S \rightarrow aA$

$A \rightarrow abS \mid b$



→ $S \rightarrow S1 \mid 0$

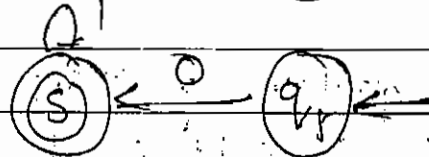
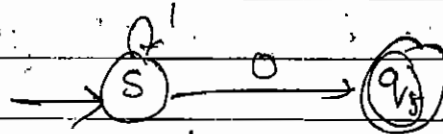
possible



Note: For Right Linear Grammar, 1st reverse the grammar, create FA, then reverse the FA.

up take
Linear G.

$S \rightarrow 1S \mid 0$

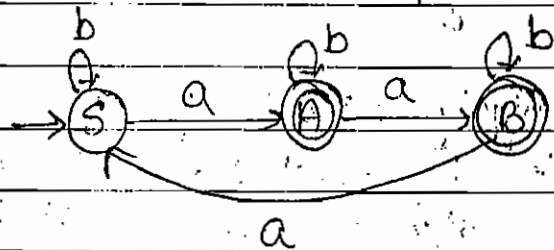


→ Lang. given by foll. grammar.

$S \rightarrow bS \mid aA \mid a$

$A \rightarrow bA \mid aB \mid b \mid a$

$B \rightarrow bB \mid aS \mid b$



Ans: no. of a's not divisible by 8.

Context Free Grammar :-

$$A \rightarrow \alpha$$

$$\alpha \in (V+T)^*$$

To show the syntactic rules for programming lang. while designing compiler.

⇒ Any grammar of the form

$$A \rightarrow \alpha$$

$$\alpha \in (V+T)^*$$

is known as CFG.

⇒ CFG is used to represent syntactic rules of P.L.

⇒ Ambiguity problem of CFG :-

A CFG is said to be ambiguous for atleast one string if there exist

↳ more than 1 Left Most Derivation

↳ ———— 1 RMD

↳ ———— 1 Parse Tree

↳ only 1 L.M.D & only 1 R.M.D., both are producing diff. Parse Trees.

⇒ The Ambiguity problem of CFG is undecidable because there is no generalised algo to check ambiguity.

⇒ Elimination of ambiguity of CFG is

also undecidable coz it is impossible to eliminate ambiguity from every ambiguous cfn.

⇒ Inherently ambiguous cfn :-

The cfn from which elimination of ambiguity is not possible.

2009

Find cfn that generates all strings of a's & b's where each string is odd length palindrome.

$S \rightarrow a S a \mid b S b \mid a \mid b$

→ Check following is ambiguous or unambiguous

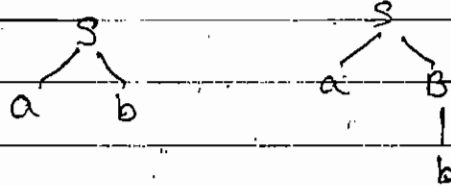
$S \rightarrow aB \mid ab$

$A \rightarrow aAB \mid a$

$B \rightarrow ABb \mid b$

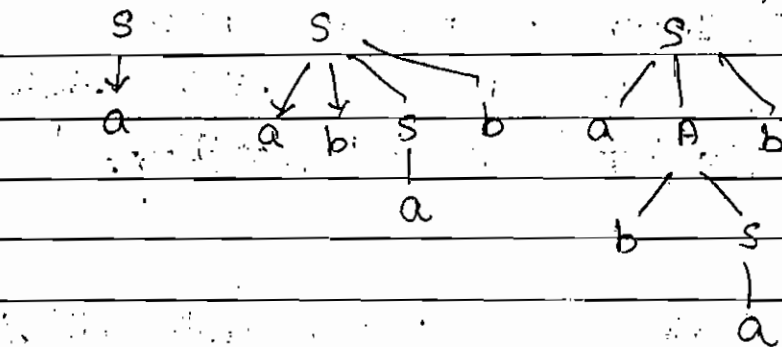
$S \rightarrow aABb \mid aBb \mid ab$

$S \rightarrow aaABbb \mid aaBbb \mid abb \mid ab$



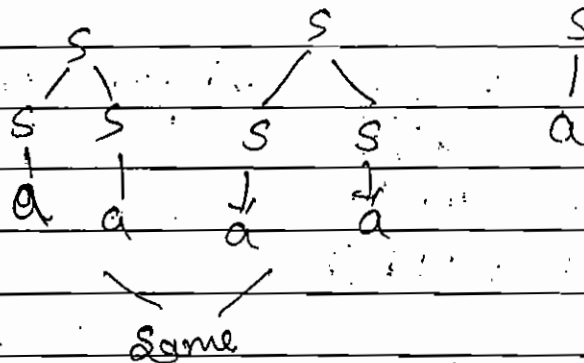
Since there are two parse trees, it is ambiguous.

→ $S \rightarrow a / absb / aAb$
 $A \rightarrow bs / aAAb$

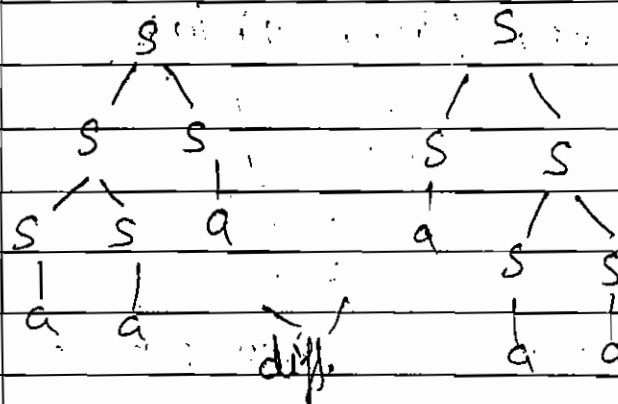


Ambiguous.

→ $S \rightarrow ss / a$

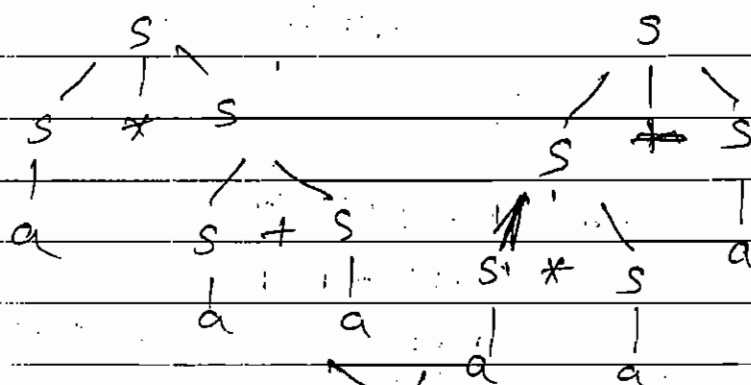
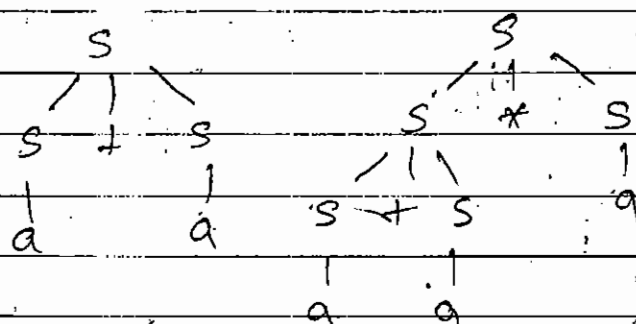
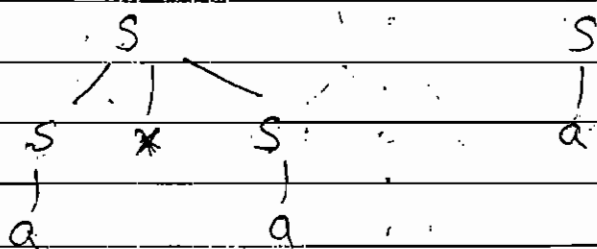


Same

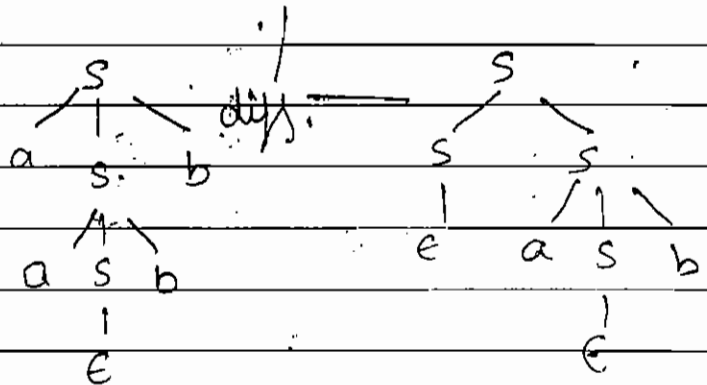
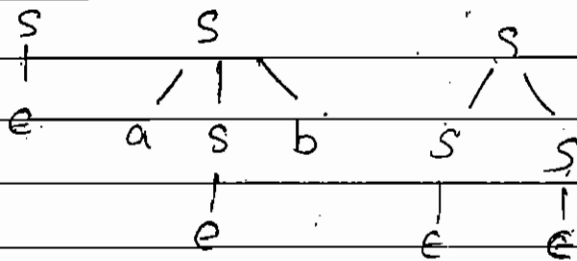


Ambiguous.

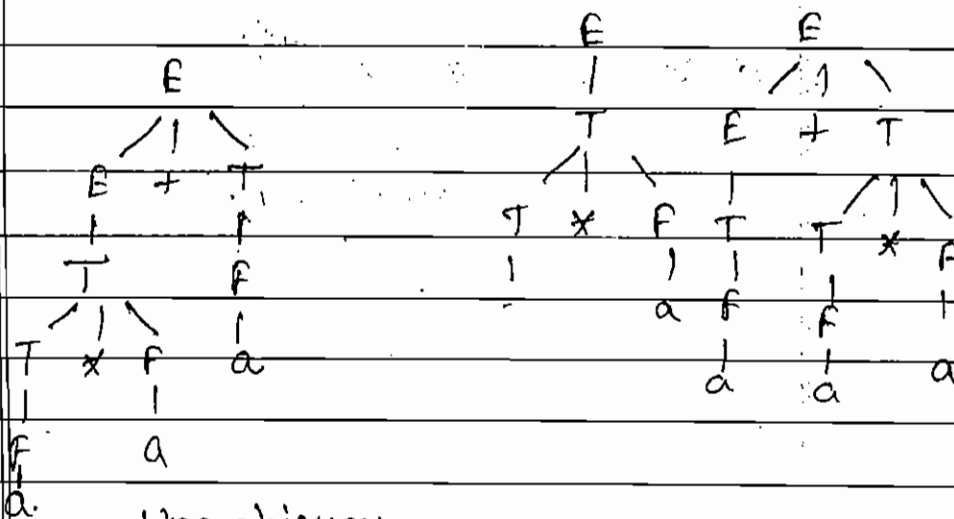
$\rightarrow S \rightarrow S * S \mid S + S \mid a$



check for $a * a + a$
+
ambiguous.

$\rightarrow S \rightarrow asb / ss / \epsilon$


Ambiguous...

 $\rightarrow$ Check $E \rightarrow E + T / T$
 $T \rightarrow T * F / F$
 $F \rightarrow a$
 $E + T * F$


Unambiguous

Simplification of CFG:

① Useless Symbol:

It means variables which are not deriving any terminals and variable which are not necessary for derivation.

→ Eliminating useless symbol -

$$S \rightarrow AB/a$$

$$A \rightarrow BC/b$$

$$XB \rightarrow aB/c$$

$$XC \rightarrow aC/B$$

= Eliminate productions which are having B & C.

$$S \rightarrow a$$

$$A \rightarrow b$$

= Since A is useless

$$S \rightarrow a$$

Only 1 production

→ $S \rightarrow \overline{AB} / \overline{AC}$

$$\checkmark A \rightarrow aAb/bAa/a$$

$$\checkmark B \rightarrow \overline{bBA} / \overline{aAB} / \overline{AB} \quad B \rightarrow bBA/aAB/AB$$

$$X C \rightarrow abCa/aDb$$

$$X D \rightarrow bD/aC$$

7 Productions.

$\rightarrow S \rightarrow \cancel{ABC} / BaB$
 $LA \rightarrow aA / \cancel{BAC} / aaa \quad \times \text{ (useless) }$
 $LB \rightarrow bBb / a$
 $LC \rightarrow \cancel{CA} / AC$

$S \rightarrow BaB$

$B \rightarrow bBb / a$

② Unit Productions:-

Any production of the form

$A \rightarrow B$, is

$\Rightarrow$ 1st Eliminate Unit then useless prod.

$\rightarrow$ Eliminate Unit Production

$S \rightarrow \underline{Aa} / B$

$\underline{B} \rightarrow \underline{A} / \underline{bb}$

$\underline{A} \rightarrow \underline{a} / \underline{bc} / B$

From Down to Top

$S \rightarrow Aa$

$S \rightarrow Aa / a / bc / bb$

$B \rightarrow a / bc / bb$

$A \rightarrow a / bc / bb$

$\rightarrow S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow C / b$

$C \rightarrow D$

$D \rightarrow E$

$E \rightarrow a$

Solⁿ: $S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow a/b$

$C \rightarrow a$

$D \rightarrow a$

$E \rightarrow a$

Useless

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow a/b$

③ Null Productions or ϵ -prod:-

Any prod. of the form

$A \rightarrow \epsilon$

The original & resultant grammar need not to be equivalent.

→ Eliminate ϵ -prod.

$S \rightarrow aS, b$

$S_1 \rightarrow aS, b/\epsilon$

Solⁿ:

$S \rightarrow aS, b/ab$

$S_1 \rightarrow aS, b/ab$

→

$S \rightarrow AB$

$A \rightarrow aAA/\epsilon$

$B \rightarrow bBB/\epsilon$

$S \rightarrow AB/B/A$

$A \rightarrow aAA/aA/a$

$S \rightarrow AB/B/A/\epsilon$

$A \rightarrow aA/a$

$B \rightarrow bB/b$

$B \rightarrow bBB/bB/b$

⇒ Elimination Sequence -

Null prod. → Unit prod → Useless Symbols.

→ Simplify following CFG

$$S \rightarrow AbaC$$

$$A \rightarrow BC$$

$$B \rightarrow b/E$$

$$C \rightarrow D/E$$

$$D \rightarrow d$$

$$S \rightarrow AbaC / Aba$$

$$A \rightarrow BC / C / B / E$$

$$B \rightarrow b$$

$$C \rightarrow D$$

$$D \rightarrow d$$

$$S \rightarrow AbaC / Aba / baC / ba$$

$$A \rightarrow BC / C / B$$

$$B \rightarrow b$$

$$C \rightarrow D$$

$$D \rightarrow d$$

$$S \rightarrow AbaC / Aba / baC / ba$$

$$A \rightarrow BC / C / B$$

$$B \rightarrow b$$

$$C \rightarrow d$$

Note:

Symbols-

$S \rightarrow AbaC / Aba / baC / ba$

$A \rightarrow \cancel{ac}b / d / b$

$B \rightarrow b$

$C \rightarrow d$

$P \rightarrow q$

Normal Form of CFG:

Applying condition or restriction on R.H.S. of CFG, v

N.F.

Chomsky
N.F.C

Greibach N.F.

① Chomsky N.F.:

Any CFG of the form

$A \rightarrow BC$

$A \rightarrow a$

② Greibach N.F.:

Any CFG of form

$A \rightarrow a\alpha$

$\alpha \in V^*$

Note: For the given E-CFG only we can construct an equivalent CNF or GNF CFG.

→ Construct an equiv. CNF-GFN for following —

$$S \rightarrow bA | aB$$

$$A \rightarrow bAA | aS | a$$

$$B \rightarrow aBB | bS | b$$

Solⁿ

$$S \rightarrow bA$$

↓

$$S \rightarrow C_b A \quad 1$$

$$C_b \rightarrow b \quad 2$$

$$S \rightarrow aB$$

↓

$$S \rightarrow C_a B \quad 3$$

$$C_a \rightarrow a \quad 4$$

$$A \rightarrow bAA$$

↓

$$A \rightarrow C_b A \quad 5$$

$$~~E \rightarrow b~~$$

$$E \rightarrow AA \quad 6$$

$$A \rightarrow aS$$

↓

$$A \rightarrow C_a S \quad 7$$

$$A \rightarrow aS$$

$$B \rightarrow aBB$$

↓

$$B \rightarrow C_a F \quad 9$$

$$~~G_a \rightarrow a~~$$

$$F \rightarrow BB \quad 10$$

$$B \rightarrow bS$$

↓

$$B \rightarrow C_b S \quad 11$$

$$B \rightarrow b$$

↓

$$B \rightarrow b \quad 12$$

for

→ Construct an equi. GNF-CFG for following -

$$S \rightarrow AB$$

$$A \rightarrow aA \mid bB \mid b$$

$$B \rightarrow b$$

Soln.

replace A by its productions

$$\checkmark S \rightarrow aAB \mid bBB \mid bB$$

$$\checkmark A \rightarrow aA \mid bB \mid b$$

$$\checkmark B \rightarrow b$$

aB

→ Construct equi GNF.

$$S \rightarrow asb \mid \epsilon$$

$$S \rightarrow asb \mid ab$$

$\Downarrow$

$$S \rightarrow asB \mid aB$$

$$B \rightarrow b$$

→ GNF

Decision Properties of CFG's

① Emptiness Problem —

Means checking whether the lang. generated by given CFG is empty or non-empty.

Algo:

① Check whether starting symbol derives any terminal or not.

② If S.S. derives atleast one terminal then the lang. generated by grammar is non-empty, otherwise empty.

→ $S \rightarrow XY$
 $X \rightarrow AX / AA$
 $A \rightarrow a$
 $Y \rightarrow BY$
 $B \rightarrow b$

It will produce empty lang.

→ $S \rightarrow XY$
 $X \rightarrow AX / AA$
 $A \rightarrow a$
 $Y \rightarrow BY / BB$
 $B \rightarrow b$

↳ Non-empty

→ $S \rightarrow XY$
 $X \rightarrow AA / XY / b$
 $A \rightarrow BC$
 $B \rightarrow AC$
 $C \rightarrow AB$
 $Y \rightarrow aY / b$

↳ Non-empty

② Finiteness Problem :-

Algo:

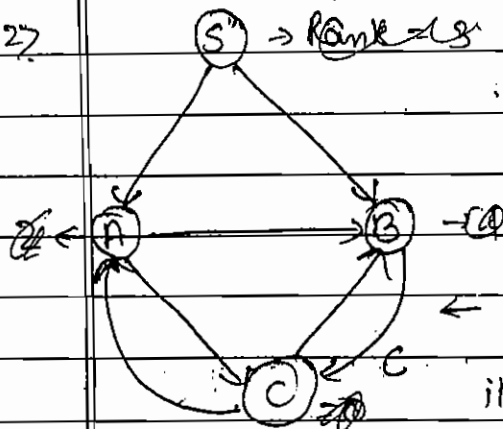
- 1> Simplify CFG.
- 2> Construct transition graph by taking variables as vertices and production as directed edges.
- 3> In resultant T.G, if there exist any self loop or cycle then the lang. generated by that grammar is infinite lang. otherwise finite lang.

→ $S \rightarrow AB$
 $A \rightarrow BC / a$
 $B \rightarrow CC / b$
 $C \rightarrow AB / a$

also find rank of variable
 S, A, B, C

1> It is simplified.

27

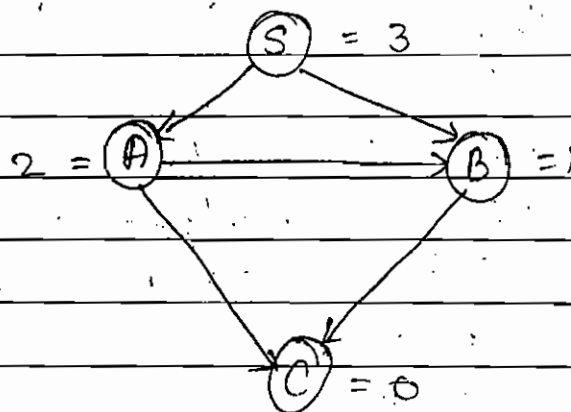


← cycle is present, so
it is infinite.

Rank of Variables:

Length of the longest path starting from that variable in T.G.

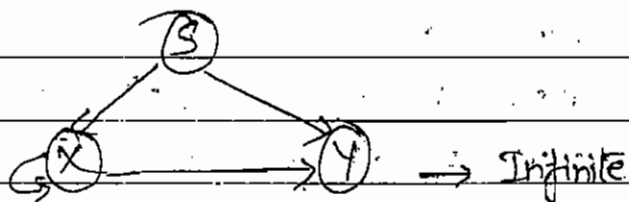
→

 $S \rightarrow AB$
 $A \rightarrow BC/a$
 $B \rightarrow CC/b$
 $C \rightarrow a$


→ $S \rightarrow XY$
 $X \rightarrow AA/XY/b$
 $XA \rightarrow BC$
 $XB \rightarrow AC$
 $XC \rightarrow BA$
 $Y \rightarrow a$

⇓

$S \rightarrow XY$
 $X \rightarrow XY/b$
 $Y \rightarrow a$



③

Membership Problem:-

Means checking whether the given string x is valid member of given CFG or not.

CYK Algo:-

C → Cocke

Y → Younger

K → Kasami

→ Dynamic Programming

→ Bottom Up Approach

→ $O(n^3)$

→ P-class

→ Chart parsing

→ ① Check whether the string baaba is a valid member of following CFG

$$S \rightarrow AB/BC$$

$$A \rightarrow BA/a$$

$$B \rightarrow CC/b$$

$$C \rightarrow AB/a$$

② Check whether the following strings are valid members of given grammar or not

X (1) b

X (5) aab

✓ (2) ba

X (6) abq

X (3) baa

✓ (7) aaba

X (4) baab

X (8) aa

③ How many substrings of given string baaba are valid members of given grammar? 5

④ How many diff. substrings of given string are valid members of given grammar? 4

⑤ How many substrings of the given string are not valid members? 10

⑥ How many substrings of the given string are only generated from variable B? 5

⑦ How many ^{diff.} substrings of given string are only generated from variable: A & C? 1

Solⁿ:

b a a b a

①

B	A, C	A, C	B	A, C
A, S	B	S, C	A, S	
∅	B	B		
∅	S, A, C			
A, S, C				

- * 1st row filled with corresponding R.H.S. Terminal.
- * 2nd row calculated by diagonal combinations.

Note: If in last block, starting symbol appears then it is member of given CPA otherwise not.

* For substring, check diagonally according to the length of substring.

③ No. of boxes containing starting symbol is the no. of ~~its~~ substrings of given string, which are valid member.

⑤ NO. of boxes containing only B.

sol

(ii) diff. = 2

⑥ 1

→ Check whether the string $aabb$ is a valid member of foll. grammar or not —

$S \rightarrow AB$

$A \rightarrow BB/a$

$B \rightarrow AB/b$

2 → Check whether foll. string are member or not.

$\times 1 > a$

$\times 4 > bb$

$\checkmark 2 > ab$

$\checkmark 3 > abb$

$\times 3 > aa$

$\times 6 > aabb$

$\checkmark 5 > bbb$

3 → How many diff. substring of given string are valid members? 5

4 → How many diff. substrings of given string are not valid members? 10

5 → How many diff. substring of given string are generated from only variable A? 4

Solⁿ:

a	a	b	b	b
A	A	B	B	B
∅	S, B	A	A	
S, B	A	S, B		
A	S, B			
S, B				

① Yes, Valid member

3 → aab
→ ab
bbb
- abbb
aabb

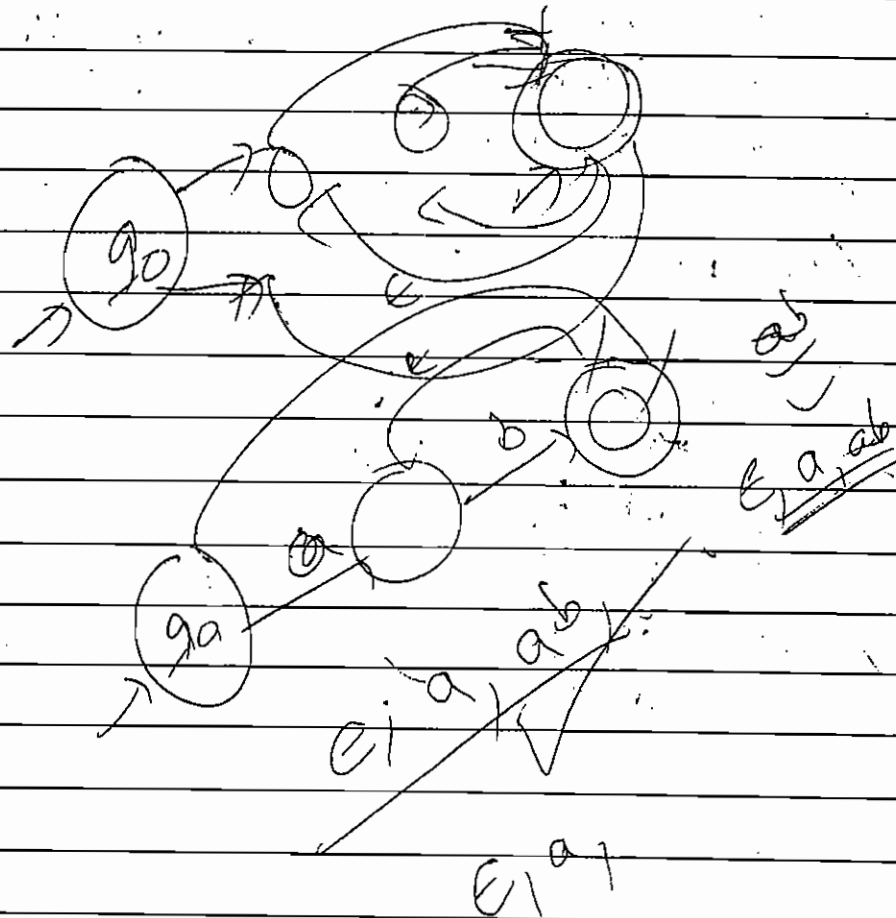
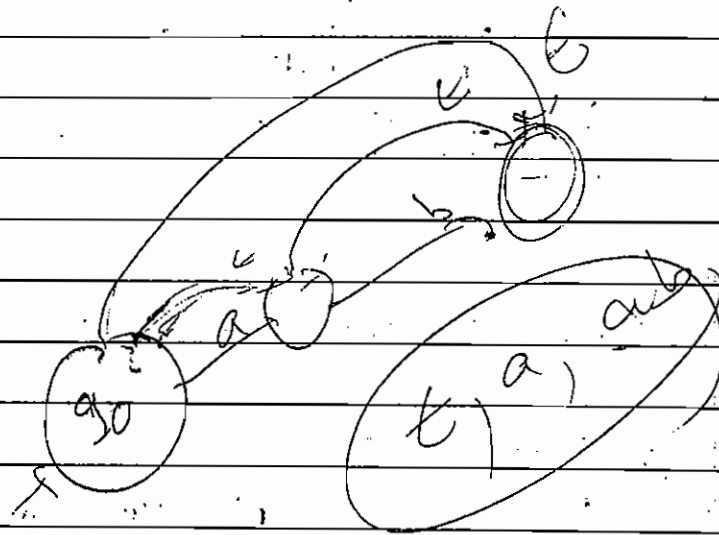
All are diff. so
all are different
& valid.

4 → 10

5 → a
a
bb
bb
abb
aabb

4

string
A ?
4



4 11, 111, 1111, 11111

10

23

2 → 10

3 → 10

4 → 10

5 → 10

6 → 10

7 → 10

8 → 10

9 → 10

10 → 10

11 → 10

12 → 10

13 → 10

14 → 10

15 → 10

6

10

10

11

12

13

14

15

16

17

18

19

20

21

22

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

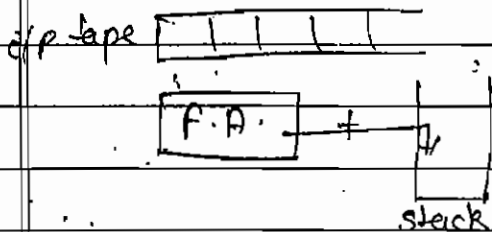
57

58

59

60

Pushdown Automata :-



$(Q, \Sigma, \delta, q_0, F, Z_0, \Gamma)$

$Q \rightarrow$ finite set of states

$\Sigma \rightarrow$ i/p alphabet

$\delta \rightarrow$ transition func.

$$\delta: Q \times \Sigma \cup \{\epsilon\} \times \Gamma \rightarrow Q \times \Gamma^*$$

$q_0 \rightarrow$ initial state

$F \rightarrow$ Set of final states

$Z_0 \rightarrow$ Initial stack element

$\Gamma \rightarrow$ stack alphabet.

Acceptance Mechanism:

① By Empty stack

By reading the complete string from left to right if the stack is empty and irrelevant about no. of final states, then given string is said to be accepted.

② By Final State

By reading string from left to right

if PDA enters into final state & irrelevant about no. of stack symbols, given string will be accepted.

Note: No. of lang. accepted by empty stack mechanism is equal to no. of lang. accepted by final states mechanism.

Notations

Transition Diagram

Transition Function

Note: If we restrict the size of stack in PDA then the lang. accepted by that PDA is all R.L. (not only finite.)

⇒ PDA fails to recognize lang. where more than 1 memory element required.

⇒ Construct PDA for the lang.

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

$$\delta: Q \times \Sigma \cup \{\epsilon\} \times \Gamma \Rightarrow Q \times \Gamma^*$$

ng from
pg and
les,
be

to right

aabb

Date / /

Page

bill 10/05/10
Student Notebooks

$$\delta(q_0, a, z_0) = (q_0, az_0)$$

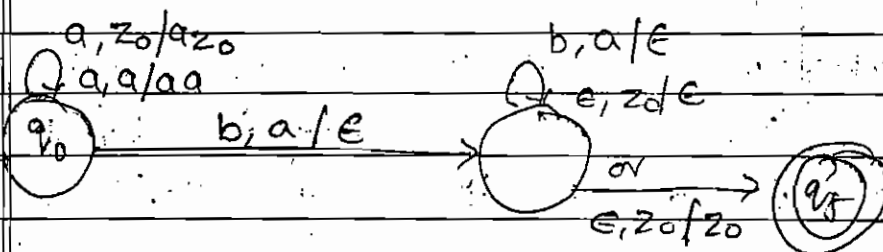
$$\delta(q_0, a, a) = (q_0, aa)$$

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

$$\delta(q_1, b, a) = (q_1, \epsilon)$$

$$\delta(q_1, \epsilon, z_0) = (q_1, \epsilon) \text{ or } (q_f, z_0)$$

by empty stack. by final state



→ Construct PDA that accept all string of a's & b's where each string contains equal no. of a's & equal no. b's.

$L = \{ab, ba, abab, aabb\}$

Logic:

Some symbols - push
diff. - pop.

$$\delta(q_0, a, z_0) = (q_0, az_0)$$

$$\delta(q_0, a, a) = (q_0, aa)$$

$$\delta(q_0, b, a) = (q_0, \epsilon)$$

$$\delta(q_0, b, z_0) = (q_0, bz_0)$$

$$\delta(q_0, b, b) = (q_0, bb)$$

$$\delta(q_0, a, b) = (q_0, \epsilon)$$

$$\delta(q_0, \epsilon, z_0) = (q_0, \epsilon)$$

$$\rightarrow L = \{a^n b^n c^m / n, m \geq 1\}$$

$$\delta(q_0, a, z_0) = (q_0, az_0)$$

$$\delta(q_0, a, a) = (q_0, aa)$$

$$\delta(q_0, b, a) = (q_0, \epsilon)$$

$$\delta(q_1, b, a) = (q_1, \epsilon)$$

$$\delta(q_1, c, z_0) = \delta(q_2, z_0)$$

$$\delta(q_2, c, z_0) = (q_2, z_0 c)$$

$$\delta(q_2, \epsilon, z_0) = (q_2, \epsilon)$$

(q_0, z_0)

final states

(q_0)

String contains

$$\rightarrow L = \{a^{m+n} b^n c^m \mid n, m \geq 1\}$$

$$\delta(q_0, a, z_0) = (q_0, a z_0)$$

$$\delta(q_0, a, a) = (q_0, aa)$$

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

$$\delta(q_1, b, a) = (q_1, \epsilon)$$

$$\delta(q_1, c, a) = (q_2, \epsilon)$$

$$\delta(q_2, c, a) = (q_2, \epsilon)$$

$$\delta(q_2, \epsilon, z_0) = (q_2, \epsilon z_0)$$

$$\rightarrow L = \{a^m b^{m+n} c^n \mid m, n \geq 1\}$$

$$\delta(q_0, a, z_0) = (q_0, a z_0)$$

$$\delta(q_0, a, a) = (q_0, aa)$$

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

$$\delta(q_1, b, a) = (q_1, \epsilon)$$

$$\delta(q_0, b, z_0) = (q_1, bz_0)$$

$$\delta(q_1, b, b) = (q_1, bb)$$

$$\delta(q_1, c, b) = (q_2, \epsilon)$$

$$\delta(q_2, c, b) = (q_2, \epsilon)$$

$$\delta(q_2, \epsilon, z_0) = (q_2, \epsilon)$$

→ $L = \{ a^m b^n c^m d^n \mid m, n \geq 1 \}$
Not possible

→ Identify the lang. generated by following PDA.

$$\delta(q_0, a, z_0) = (q_0, az_0)$$

$$\delta(q_0, a, a) = (q_0, aa)$$

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

$$\delta(q_1, b, b) = (q_1, bb)$$

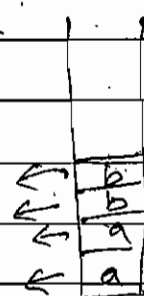
$$\delta(q_1, c, b) = (q_2, \epsilon) \quad L = \{ a^n b^m c^m d^n \mid m, n \geq 1 \}$$

$$\delta(q_2, c, b) = (q_2, \epsilon)$$

$$\delta(q_2, d, a) = (q_3, \epsilon)$$

$$\delta(q_3, d, a) = (q_3, \epsilon)$$

$$\delta(q_2, \epsilon, z_0) = (q_2, \epsilon)$$



→ Construct PDA -

$$L = \{a^m b^n \mid m > n\}$$

$$\delta(q_0, a, z_0) = (q_0, a z_0)$$

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

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

$$\delta(q_1, b, a) = (q_1, \epsilon)$$

$$\delta(q_1, \epsilon, a) = (q_2, \epsilon)$$

$$\checkmark \delta(q_2, \epsilon, a) = (q_2, \epsilon)$$

$$\checkmark \delta(q_2, \epsilon, z_0) = (q_2, \epsilon)$$

→ $L = \{a^n b^{2n} \mid n \geq 1\}$

$$\delta(q_0, a, z_0) = (q_0, a z_0)$$

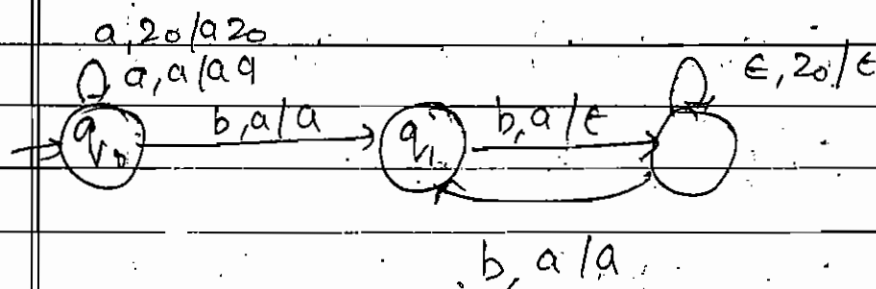
$$\delta(q_0, a, a) = (q_0, a a)$$

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

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

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

$$\delta(q_2, \epsilon, z_0) = (q_2, \epsilon)$$



$$\rightarrow L = \{a^n b^{2^n} \mid n \geq 1\}$$

Note: If symbols are not in A.P., then FA & PDA both are not possible.

$$\rightarrow \text{Construct } L = \{wcw^R \mid w \in \{a, b\}^*\}$$

abcba $\rightarrow$ palindrom

$$\delta(q_0, a, z_0) = (q_0, az_0)$$

$$\delta(q_0, b, z_0) = (q_0, bz_0)$$

$$\delta(q_0, a, a) = (q_0, aa)$$

$$\delta(q_0, a, b) = (q_0, ab)$$

$$\delta(q_0, b, a) = (q_0, ba)$$

$$\delta(q_0, c, b) = (q_0, b)$$

$$\delta(q_0, b, b) = (q_2, \epsilon)$$

$$\delta(q_2, b, b) = (q_2, \epsilon)$$

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

$$\delta(q_2, \epsilon, z_0) = (q_2, \epsilon)$$

$$\delta(q_0, a, z_0) = (q_0, a z_0)$$

$$\delta(q_0, b, z_0) = (q_0, b z_0)$$

$$\delta(q_0, a, a) = (q_0, aa)$$

$$\delta(q_0, a, b) = (q_0, ab)$$

$$\delta(q_0, b, a) = (q_0, ba)$$

$$\delta(q_0, b, b) = (q_0, bb)$$

$$\delta(q_0, c, z_0) = (q_1, z_0)$$

$$\delta(q_0, c, a) = (q_1, a)$$

$$\delta(q_0, c, b) = (q_1, b)$$

$$\delta(q_1, a, a) = (q_1, \epsilon)$$

$$\delta(q_1, b, b) = (q_1, \epsilon)$$

$$\delta(q_1, \epsilon, z_0) = (q_1, \epsilon)$$

$$\rightarrow L = \{w w^R \mid w \in \{a, b\}^*\}$$

$$\delta(q_0, \epsilon, z_0) = (q_0, \epsilon)$$

$$\delta(q_0, a, z_0) = (q_0, a z_0)$$

$$\delta(q_0, b, z_0) = (q_0, b z_0)$$

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

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

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

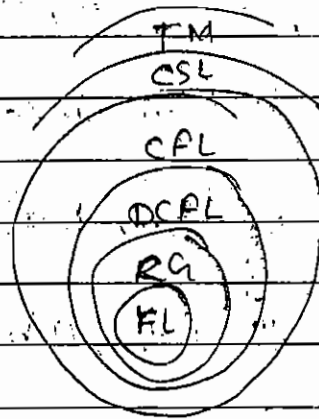
$$\delta(q_0, b, b) = (q_0, b b) \text{ or } (q_1, \epsilon)$$

$$\delta(q_1, a, a) = (q_1, \epsilon)$$

$$\delta(q_1, b, b) = (q_1, \epsilon)$$

$$\delta(q_1, \epsilon, z_0) = (q_1, \epsilon)$$

⇒ All DCFLs are CFL but ^{no. of} all CFL need not be DCFL. Hence the lang. accepted by NPDA is more than DPDA.



→ which of the following is CFL but not DCFL.

a) $L = \{w c w^R \mid w \in \{0,1\}^*\}$

b) $L = \{0^n 1^n 2^n \mid n > 1\}$

c) $L = \{0^n 1^m 2^{n+m} \mid n, m \leq 10\} \rightarrow \text{Finite lang.}$

✓ d) None.

→ $L = \{w c w \mid w \in (a,b)^*\}$ PDA not possible
 abcab abacaba bbacbbba

$\delta(q_0, \epsilon, z_0) = (q_0, \epsilon)$

$\delta(q_0, a, z_0) = (q_0, a)$

$\delta(q_0, b, z_0) = (q_0, b)$

need
accepted

$$\times \delta(q_0, a, a) = (q_0, aa)$$

$$\delta(q_0, a, b) = (q_0, ab)$$

$$\delta(q_0, b, a) = (q_0, ba)$$

$$\delta(q_0, b, b) = (q_0, bb)$$

$$\delta(q_0, c, \epsilon) = (q_0, \epsilon z_0)$$

$$\delta(q_0, c, a) = (q_0, \epsilon a)$$

$$\delta(q_0, c, b) = (q_0, \epsilon b)$$

not

$$\rightarrow L = \{w w \mid w \in (a, b)^*\}$$

PDA not possible

note
eng.

$\rightarrow$ identify the lang. accepted by following PDA.

$$\delta(q_0, a, z_0) = (q_1, z_0)$$

$$\delta(q_0, b, z_0) = (q_0, z_0)$$

$$\delta(q_1, a, z_0) = (q_2, z_0)$$

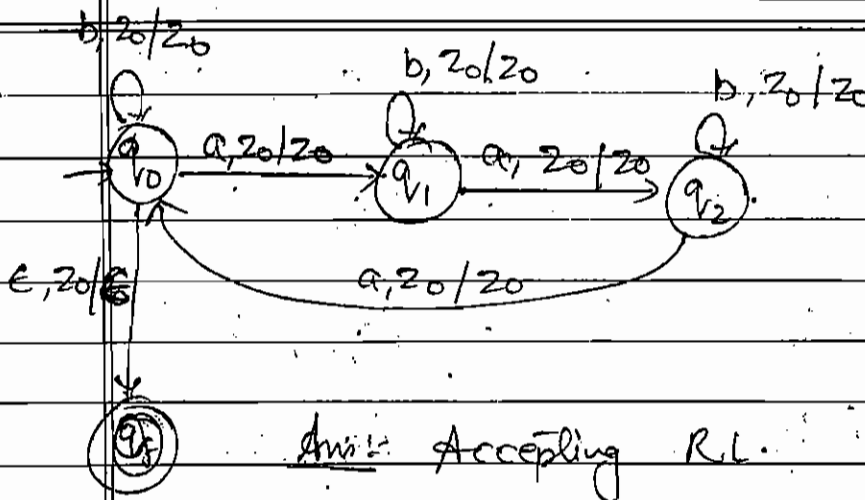
$$\delta(q_1, b, z_0) = (q_1, z_0)$$

$$\delta(q_2, a, z_0) = (q_0, z_0)$$

$$\delta(q_2, b, z_0) = (q_2, z_0)$$

$$\delta(q_0, \epsilon, z_0) = (q_f, \epsilon)$$

is possible



Ans: Accepting RL.

a^*b^*

70%

Context Free Languages:

→ Which of the following lang. are
 a) CFL b) non-CFL, c) CFL but not regular
 d) non CFL - non regular.

~~CFL~~ Non-CFL

① $L = \{a^n b^n c^m \mid n > m\}$

a) ② $L = \{a^n b^n c^n d^n \mid n \leq 1\}$

c) ③ $L = \{a^n b^m \mid n < m\}$

d) ④ $L = \{a^n b^n c^n d^n \mid n \geq 1\}$

c) ⑤ $L = \{a^n b^m c^m d^n \mid m, n \geq 1\}$

a) ⑥ $L = \{a^n b^m c^n d^m \mid n, m \leq 1\}$

c) ⑦ $L = \{0^n 1^m 2^{n+m} \mid m, n \geq 1\}$

$0^n 1^m 2^{n+m}$

? d) $L = \{0^n 1^{2n} 2^{3n} \mid n \geq 1\}$

a) 9) $L = \{x \in \Sigma^* \mid x, y \in \{0,1\}^*\}$

a) 10) $L = \{xx^R \mid x \in \{a,b\}^*, |x| = 1\}$

d) 11) $L = \{x \in \Sigma^* \mid x \in \{0,1\}^*\}$

d) 12) $L = \{www^R \mid w \in \{a,b\}^*\}$

d) 13) $L = \{a^n b^{3n} \mid n \geq 1\}$

c) 14) $L = \{a^m b^n \mid m \neq n\}$

c) 15) $L = \{a^m b^n \mid m = 2n+1\}$

c) 16) $L = \{a^i b^j \mid i \neq 2j+1\}$

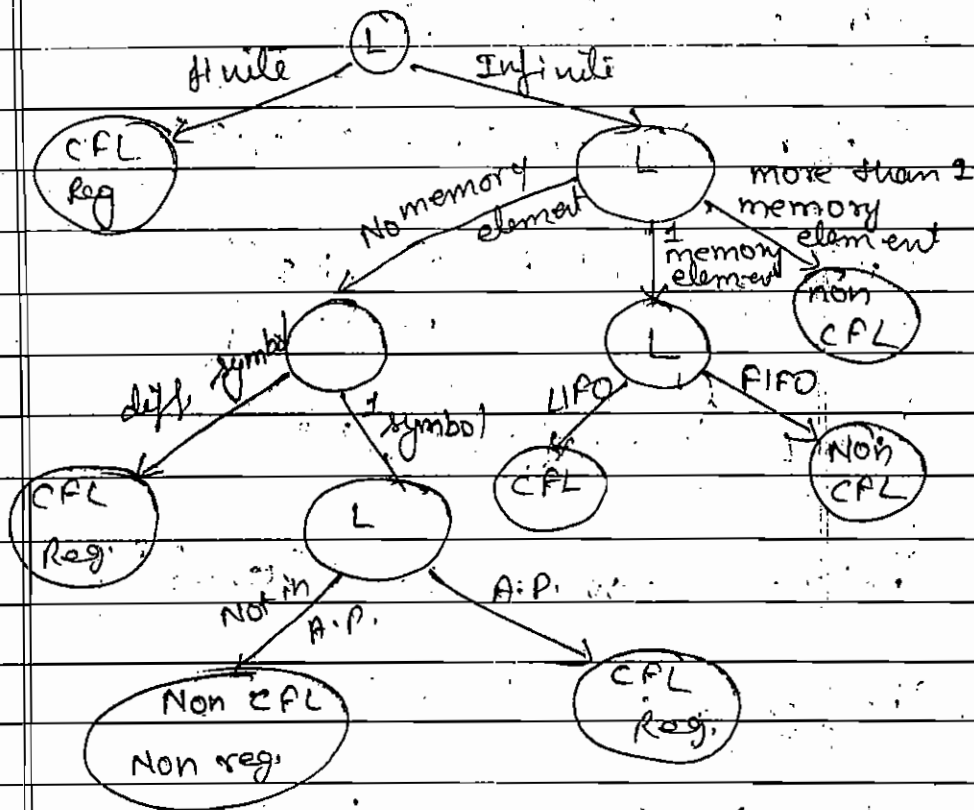
d) 17) $L = \{a^{n^2} \mid n \geq 1\}$

a) 18) $L = \{a^{2n} \mid n \geq 1\}$

d) 19) $L = \{b^{n!} \mid n \geq 1\}$

d) 20) $L = \{a^m \mid m \text{ is a prime no.}\}$

a) 21) $L = \{a^k \mid k \text{ is even no.}\}$



d (22) $L = \{a^i b^j c^k \mid i > j > k\}$

c (23) $L = \{a^i b^j c^k \mid j = i + k\}$

c (24) $L = \{a^i b^j c^k d^l \mid i = k \text{ or } j = l\}$

c (25) $L = \{a^i b^j c^k d^l \mid i \geq j \text{ and } k = l\}$

d (26) $L = \{a^i b^j c^k d^l \mid i = k \text{ \& } j = l\}$

c (27) $L = \{a^i b^j c^k d^l \mid i = l \text{ \& } j = k\}$

a (28) $L = \{a^m b^l c^k d^n \mid m, l, k, n \geq 1\}$

a. (29) $L = \{a^n b^{4m} \mid n, m \geq 1\}$

a. (30) $L = \{a^3, a^8, a^{13}, \dots\}$

a. (31) $L = \{1^{2n+1} \mid n \geq 1\}$

d. (32) $L = \{0^n \mid n \geq 1\}$

a. (33) $L = \{w \mid w \in \{a, b, c\}^* \text{ and } |w| \geq 100\}$

d. (34) $L = \{w \mid w \in \{a, b, c\}^* \text{ and } n_a(w) = n_b(w) = n_c(w)\}$

c. (35) $L = \{w \mid w \in \{a, b, c\}^* \text{ and } n_a(w) \geq n_b(w) + 1\}$

a. (36) $L = a^* b^* c^* d^*$

more than 2
non
element

non
CFL

Non
CFL

Context Free $\rightarrow$ PDA

Let GNF

$$A \rightarrow a\alpha / b\gamma$$

where $\alpha, \gamma \in V^*$

then we can construct PDA as follows steps:

① Let push the starting symbol Let A.

$$\delta(q_0, \epsilon, z_0) = (q_1, A z_0)$$

② Push R.H.S. of A.

$$\delta(q_1, a, A) = (q_1, \alpha)$$

$$\delta(q_1, b, A) = (q_1, \gamma)$$

③ Final State

$$\delta(q_1, \epsilon, z_0) = (q_f, \epsilon)$$

$\rightarrow$ Construct equivalent PDA for following CFG:

$$S \rightarrow aSb \mid ab$$

$$\delta(q_0, \epsilon, z_0) \rightarrow (q_1, S z_0)$$

$$\delta(q_1, a$$

$$S \rightarrow aSb \mid ab$$

$$Sb \rightarrow$$

$$S \rightarrow aSb \mid ab$$

$$B \rightarrow b$$

① $\delta(q_0, \epsilon, z_0) \rightarrow (q_1, S z_0)$

$$\textcircled{2} \delta(q_1, a, s) = (q_1, sB) \text{ or } (q_1, B)$$

$$\delta(q_1, b, B) = (q_1, \epsilon)$$

follows

$$\textcircled{3} \delta(q_1, \epsilon, z_0) = (q_f, \epsilon)$$

Note:- Grammar to PDA is by default NPDA.

$$\begin{aligned} \rightarrow S &\rightarrow aAA \\ A &\rightarrow aS/bS/a \end{aligned}$$

$$\textcircled{1} \delta(q_0, \epsilon, z_0) \rightarrow (q_1, s z_0)$$

$$\begin{aligned} \textcircled{2} \delta(q_1, a, s) &\rightarrow (q_1, AA) \\ \delta(q_1, a, A) &\rightarrow (q_1, s) \text{ or } (q_1, \epsilon) \\ \delta(q_1, b, A) &\rightarrow (q_1, s) \end{aligned}$$

eg CFG:

$$\textcircled{3} \delta(q_1, \epsilon, z_0) = (q_f, \epsilon)$$

$$\begin{aligned} \rightarrow S &\rightarrow 0A \\ A &\rightarrow 0AB/1 \\ B &\rightarrow 1 \end{aligned}$$

$$\textcircled{1} \delta(q_0, \epsilon, z_0) = (q_1, s z_0)$$

$$\textcircled{2} \quad \delta(q, 0, s) = (q, A)$$

$$\delta(q, 0, A) = (q, AB)$$

$$\delta(q, 1, A) = (q, \epsilon)$$

$$\delta(q, 1, B) = (q, \epsilon)$$

$$\textcircled{3} \quad \delta(q, \epsilon, Z_0) = (q, \epsilon)$$

Closure Properties of DCFL & CFL

① Union Op:-

$$L_1 \cup L_2$$

Let L_1 & L_2 are any two CFLs, & S_1 & S_2 are CFGs for L_1, L_2 respectively. Then ' $L_1 \cup L_2$ ' is

$$S \rightarrow S_1 \mid S_2$$

We can construct the PDP for this hence $L_1 \cup L_2$ is always CFL. Hence CFLs are closed under union-op.

* DCFL are not closed under Union, coz we can't construct PPDA from union grammar.

② Concatenation:

$$S \rightarrow S_1 S_2$$

We can construct PDA for this hence $L_1 L_2$ are always CFL. So, CFLs are closed under Union Concatenation

Note

* DCFLs are not closed under Concatenation op.

③ Kleene Closure:

$$L_1 \rightarrow \text{CFL} \rightarrow S_1$$

$$L_1^* \rightarrow S_1^*$$

~~Kleene~~ CFLs are closed under Kleene closure, but DCFLs are not closed.

④ Positive Closure:

$$L_1 \rightarrow \text{CFL} \rightarrow S_1$$

$$L_1^+ \rightarrow S_1^+$$

CFLs are closed under Positive closure but DCFLs are not closed.

⑤ Intersection Closure:

Let L_1 & L_2 are any two CFLs and Intersection of L_1 & L_2 may or may not Context free. Hence CFL are not closed under Intersection op.

Note Intersection of two DCFLs may or may not DCFL hence DCFLs are not closed under Intersection op.

⑥ Intersection with R.L.:

Let L_1 be CFL & L_2 be R.L., then $L_1 \cap L_2$ is always CFL, need not be regular. Hence, CFL are closed under intersection with

R.L. operation. Hence $\rightarrow$

Note: Intersection of DCFL & R.L. is always DCFL & need not be R.L. hence DCFL are closed under Intersection with R.L. op.

⑦ Complement Op:-

Let L_1 & L_2 are two CFLs. Assume that L_1 & L_2 are closed under Complement, $\Rightarrow$

$$\begin{aligned} \overline{L_1 \cup L_2} &= \text{CFL} & \overline{L_1} &\rightarrow \Sigma^* - L_1 \\ \overline{L_1 \cap L_2} &\rightarrow \text{CFL} \end{aligned}$$

$$L_1 \cap L_2 \rightarrow \text{NCFL}$$

So, CFLs are not closed under Complement Op.

Note: Complement of DCFL is always DCFL hence DCFLs are closed under complement op. (Complemented DPDA is possible.)

⑧ Diff. Op:-

$$\begin{array}{cc} L_1 & L_2 \\ \text{CFL} & \text{CFL} \end{array}$$

$$L_1 - L_2 = L_1 \cap \overline{L_2} \rightarrow \text{NCFL}$$

$$L_1 - L_2 = \text{NCFL}$$

So, CFLs are not closed under Diff. op.

Note: DCFLs are not closed under Diff. op.

⑨ Diff. with Reg. lang.:-

$$\begin{array}{cc} L_1 & L_2 \\ \text{CFL} & \text{Reg.} \end{array}$$

- ① Uni.
- ② Inter
- ③ Krec
- ④ Posi
- ⑤ ~~Diff~~
- ⑥ Inte
- ⑦ Comple
- ⑧ Diff.
- ⑨ Diff.
- ⑩ Revers

$$L_1 \cup L_2 = L_1 \cap L_2 \rightarrow \text{CFL}$$

R.O.

Hence, CFL & DCFL are closed under

Diff. with R.L.

(10) Reverse Op.

$$L_1 \rightarrow \text{CFL } \{a^n b^n | n \geq 1\}$$

$$S \rightarrow a s b / a b \rightarrow \text{PDA}$$

$$L_1^R \rightarrow \text{CFL } \{b^n a^n | n \geq 1\}$$

$$S \rightarrow b s a / b a \rightarrow \text{PDA}$$

CFLs are closed under Reverse Op.

DCFLs are not closed under Reverse Op.

CFL

DCFL

① Union

✓

X

② Intersection

X

X

③ Kleen

✓

X

④ Positive

✓

X

⑤ Concatenation

✓

X

⑥ Intersection with R.L.

✓

✓

⑦ Complement

X

✓

⑧ Diff.

X

X

⑨ Diff. with R.L.

✓

✓

⑩ Reversal

✓

X

→ Let L_1 & L_2 are two CFLs, L_3 & L_4 are two RLs then which of following is CFL or not may or may not CFL —

- ① $L_1 \cup L_2 \rightarrow$ CFL A) CFL
B) may or may not CFL
- ② $L_3 \cup L_4 \rightarrow$ CFL
- ③ $L_1 \cup L_2 \cup L_3 \cup L_4 \rightarrow$ CFL
- ④ $(L_1 \cup L_2) \cap (L_3 \cap L_4) \rightarrow$ CFL
- ⑤ $(L_1 \cap L_3) \cup (L_2 \cap L_4) \rightarrow$ ~~not CFL~~ A
- ⑥ $(L_1 \cap L_4) \cap (L_2 \cap L_3) \rightarrow$ B
- ⑦ $(L_1 \cap L_2) \cap (L_3 \cup L_4) \rightarrow$ B
- ⑧ $(L_1 \cap L_3) \cup (\bar{L}_2 \cup L_4) \rightarrow$ B
- ⑨ $(L_2 - \bar{L}_4) \rightarrow$ A
- ⑩ $(\bar{L}_3 - \bar{L}_4) \cap (L_2 - L_3) \rightarrow$ A
- ⑪ $(L_2 - L_4) - (\bar{L}_3 - \bar{L}_4) \rightarrow$ ~~not CFL~~ A
- ⑫ $(L_1 - L_2) - (L_3 - \bar{L}_4) \rightarrow$ B

2004

→

Let $L = L_1 \cap L_2$

where $L_1 = \{a^n b^n c a^m b^m \mid n, m \geq 0\} \rightarrow$ CFL

$L_2 = \{a^i b^j c^k \mid i, j, k \geq 1\} \rightarrow$ ~~not CFL~~ Reg.

then L is

- ✓ a) CFL but not R.L. b) CFL & R.L.
- c) CFL but not CSL d) Not Recursive Enum.

$$L_1 \cap L_2 = a^n b^n c$$

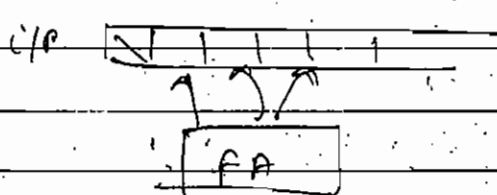
Q. Care
CFL

→ Consider two lang. P & Q. where P is R.L. & Q is CFL. then which of the following is always R.L. -

or may not CFL

- a) $P \cap Q$ b) $Q - P$ c) $\Sigma^* - P$ d) $\Sigma^* - Q$

TURING MLC:-



- ① Infinite
- ② Read & Write

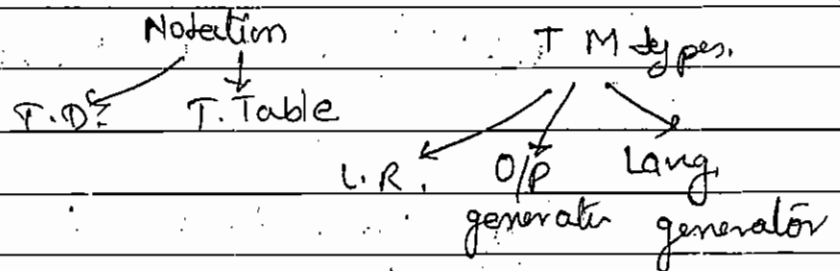
$$T.M. = (Q, \Sigma, \delta, q_0, F, B, \Gamma)$$

δ : transition func.

$$Q \times \Sigma \rightarrow Q \times \Sigma \times \{L, R\}$$

B: Blank Symbol

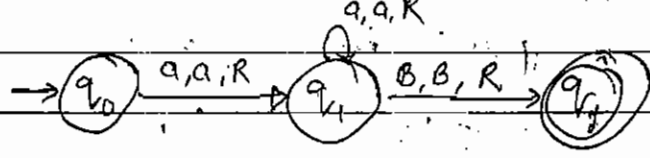
Γ : Tape alphabet



Q. CFL
R.L. Reg.

→ Construct TM for lang.

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

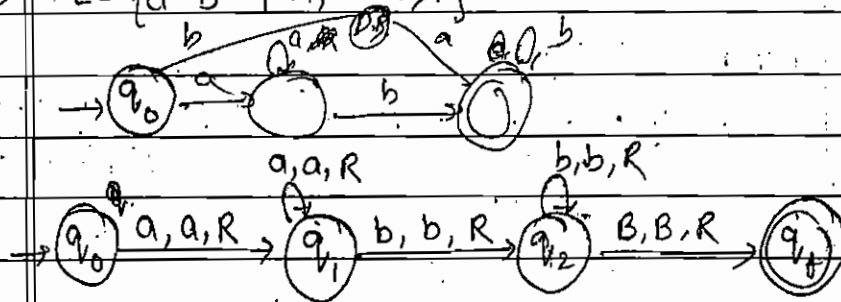


R. Enum

	a	b
q_0	(q_1, a, R)	

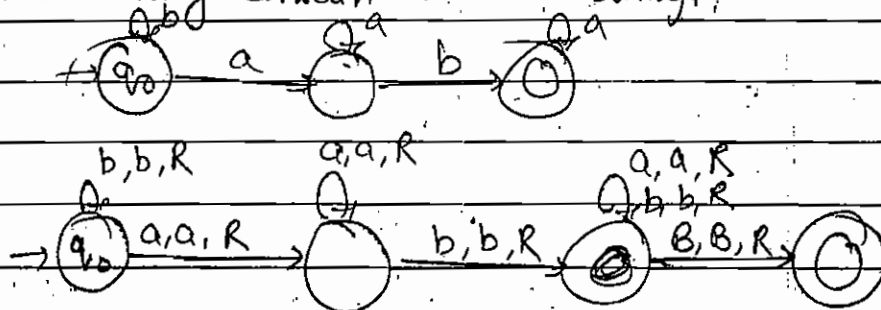
q_1	(q_1, a, R) (q_1, b, R)
q_f	HALT

→ $L = \{a^n b^m \mid n, m \geq 1\}$



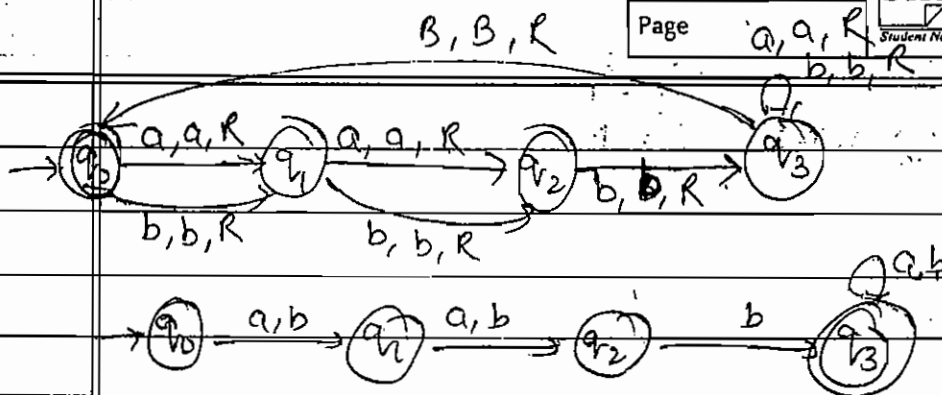
Note: If any string goes to halt state, not acceptable by TM otherwise acceptable.

→ TM that accepts all strings of a's & b's where each string contains ab as substring.

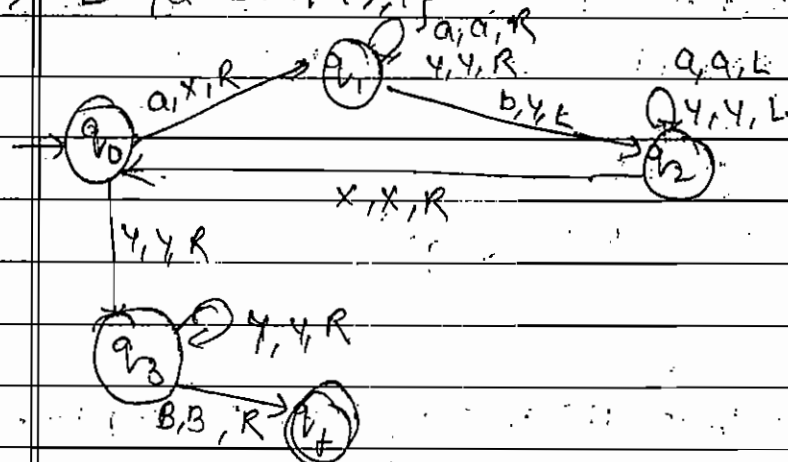


→ Identify lang. determined by following TM -

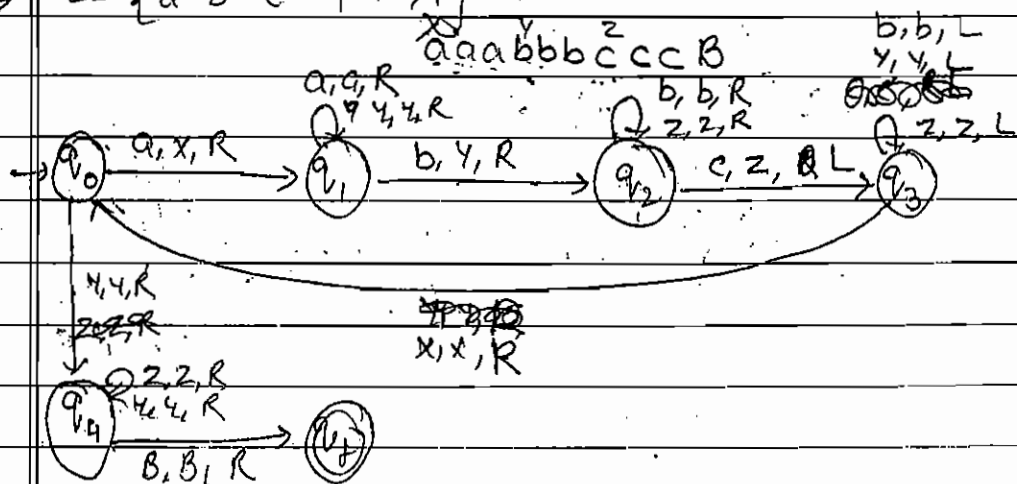
	a	b	B	B
q_0	(q_1, a, R)	(q_1, b, R)		HALT
q_1	(q_2, a, R)	(q_2, b, R)		HALT
q_2	HALT	(q_3, b, R)		HALT
q_3	(q_3, a, R)	(q_3, b, R)	(q_1, B, R)	
q_f				HALT



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



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

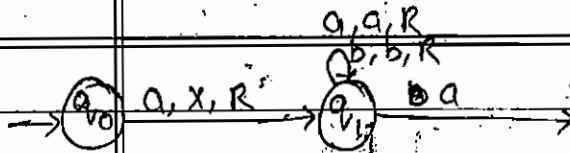


TM that accepts $a^n b^n$ where each string is even length palindrom.

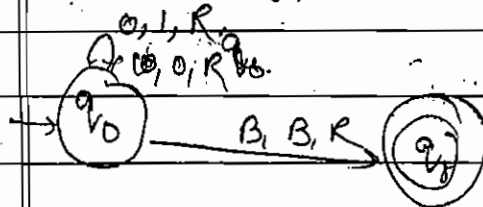
$L = \{ww^R\}$

ab aab er
xx BB

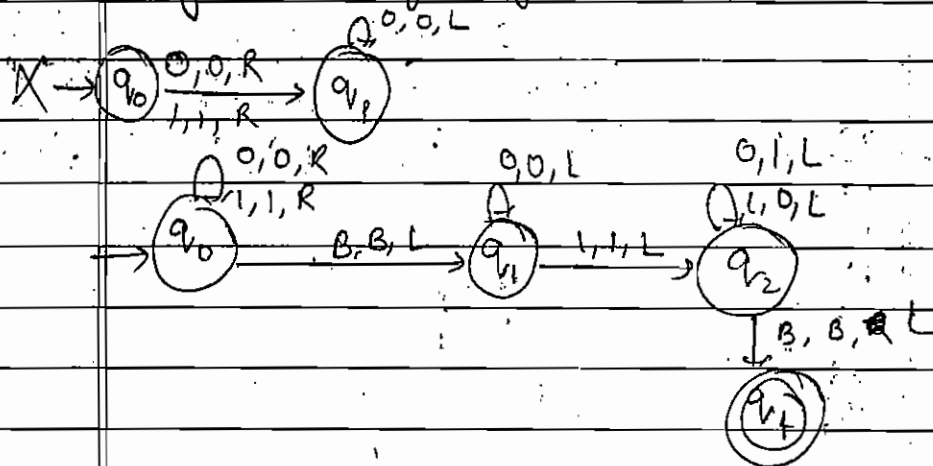
abaaba
RR BB



→ Construct TM that produces 1's complement of given binary string as O/P.



→ Construct TM that produces 2's complement of binary string as O/P (Assume that we are using two ways infinite TM.)



Lang. Generated Category of TM:

String of lang. generated by TM are effectively enumerated in either alphabetical order or canonical order, then that lang. is known as Recursive lang.

Rec. Lang.

R.E.L.

R.E.L.

If string of lang. is enumerated in either alphabetical or canonical or in any order, then that lang. is R.E.L.

impliment

All Recursive lang. are R.E.L but all R.E.L. need not be Recursive.

2003
→

If string of lang. produced by T.M. are effectively enumerated in lexico graphic order then that lang. is known as - Recursive lang.

neat of

are

- ① Re.E.L. but not Recursive
- ② Not Recursive & not R.E.L.
- ✓ ③ Recursive & R.E.L.
- ④ NOT

Lexico Graphic → Alphabetical order

Modifications of T.M. :-

/ a a b b B

t

- ① 2-way infinite tape TM
- ② Multitape
- ③ Two Tape TM
- ④ NTM
- ⑤ Of Time TM

1 more

actical

not lang

After applying modification to TM, expressive power of it remains same.

Note: For every NTM of polynomial time complexity, we can construct an equivalent DM of exponential time complexity.

Date _____
Page _____

bill 10/10
Student Notebooks

PLAUBI

Recursive & R.E.L:

→ By reading every string S in lang., TM always Halts, the lang. accepted by such type of TM are known as recursive lang.

→ By reading string S in lang., TM may Halt or may enter into ∞ loop, lang. recognized by such TM are known as REL.

→ All Recursive lang. are REL but all REL need not be recursive.

REL

Recursive lang.

CSL

CFL

DCFL

R.L.

Operation

- ① Union
- ② Concatenation
- ③ Kleen Closure
- ④ Positive Closure
- ⑤ Intersection
- ⑥ Complement
- ⑦ Difference
- ⑧ Reversal
- ⑨ Intersection with R.L.
- ⑩ Diff. with R.L.

Problem

REL

Recursive

CSL

CFL

DCFL

R.L.

Q. 13

4, we

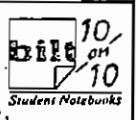
similarly if A, B & C are disjoint

$$P(A \cup B \cup C) = P(A) + P(B) + P(C)$$

Date / /

Page

Undecidability



ways

are

may

organized

REL

REL

Recursive

CSL

CFL

DEFL

R.E.

Problem

DIS WEL?

Is $L = \emptyset$?

Is $L = \Sigma^*$?

(Complement closed, emptiness)

Is $L_1 \neq L_2$?

$L_1 \subseteq L_2$?

$(L_1 \cup L_2 \in \text{mb})$

$L_1 \cap L_2 = \emptyset$?

If L finite or not?

Complement of L is L ?

same type?

Yes

Yes

Yes

Yes

Yes

Yes

Yes

Yes

Yes

Collectively Exhaustive Events:

A set of Events of S is said to be exhaustive if it includes all possible outcomes of R.E.

Date / /

Page



→ Consider

$A = \{1, 3, 5\}$, $B = \{2, 4, 6\}$ of $S = \{1, 2, 3, 4, 5, 6\}$ in throwing single dice, then we can observe that two events A & B mutually Exclusive, Collective Exha & equally likely events.

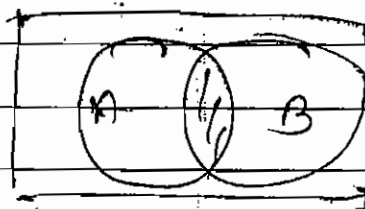
Addition theorem of Probability :-

Let A & B any two events of S of R.E. Then probability of occ. of atleast one of two event A, B is given by

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

Proof:- From fig.

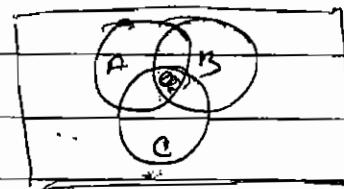
$$\frac{n(A \cup B)}{n(S)} = \frac{n(A) + n(B) - n(A \cap B)}{n(S)}$$



Therefore $P(A \cup B)$ gives probability of occurrence of event A or B or Both A & B .

Note: Similarly $P(A \cup B \cup C) = P(A) + P(B) + P(C) - P(A \cap B) - P(A \cap C) - P(B \cap C) + P(A \cap B \cap C)$

therefore $P(A \cup B \cup C)$ gives probability of occ. of A or B or C or any two of them or all three of them. i.e. occ. of atleast 1 of them.



Note: Suppose A & B are disjoint then

$$P(A \cup B) = P(A) + P(B)$$

Complement of Event:-
let $E \subseteq S$, be any event
of sample space S of random
Exp., then E' or $\bar{E}$ or E^c

Date / /
Page

$$P(\bar{E}) = \frac{n(\bar{E})}{n(S)} = \frac{n(S) - n(E)}{n(S)}$$

$$P(E) = 1 - P(\bar{E})$$

gives probability of non occurrence of event E .
Therefore $P(E) + P(\bar{E}) = 1$

Note!

$\frac{n(E)}{n(S)} \rightarrow$ odds in Favorable of 'E'.

$\frac{n(E')}{n(S)} \rightarrow$ odds against to 'E'.

Note!

If it is given either of them, we can
calculate $P(E)$ & $P(E')$.

$$P(E) = \frac{n(E)}{n(S)} = \frac{n(E)}{n(E) + n(E')}$$

$$P(E') = \frac{n(E')}{n(S)} = \frac{n(E')}{n(E) + n(E')}$$

Equally Likely Events:

Two or more events of a sample space
are said to be Equally Likely Events, if there
is no reason to believe that one particular event
occurs more than others.

Mutually Exclusive Events:

Two or more events of S are said to be
Mutually Exclusive or Disjoint events if one of
them occurs then the others will not occur or
no two of them can occur together.

Probability of Occurrence of an Event :-

Let E subset of S , be any event of sample space S of $R.F.$, the probability of occurrence of E

$$P(E) = \frac{n(E)}{n(S)} = \frac{\text{no. of favorable cases of } E}{\text{total no. of cases in } S}$$

→ Consider $S = \{1, 2, 3, 4, 5, 6\}$, then

$A = \{1, 3, 5\}$, $B = \{2, 3, 5\}$, $C = \{5, 6\}$ etc.

are events then

$$P(A) = \frac{n(A)}{n(S)} = \frac{3}{6} = \frac{1}{2} = 50\%$$

$$P(B) = \frac{n(B)}{n(S)} = \frac{3}{6} = \frac{1}{2} = 50\%$$

$$P(C) = \frac{n(C)}{n(S)} = \frac{2}{6} = \frac{1}{3} = 33\%$$

Note:

We know that empty set is subset to any set, i.e.

$\emptyset \subset S$, then

$$P(\emptyset) = \frac{n(\emptyset)}{n(S)} = \frac{0}{n(S)} = 0 = 0\%$$

hence $\emptyset$ is called impossible event.

Note:

We know that every set is subset to itself i.e.

$$S \subseteq S$$

$$P(S) = \frac{n(S)}{n(S)} = \frac{1}{1} = 1 = 100\%$$

hence, the sample space S itself is called certain or sure event.

Note:

We know that $\emptyset \subseteq E \subseteq S$

$$n(\emptyset) < n(E) < n(S)$$

$$0 < P(E) < 1$$

Hence, probability of occurrence of any event lies b/w 0 & 1 (0% - 100%).

Cases of E

Ins.

r no. of objects are of 3rd category & remain objects are independently diff. Then total no. of arrangements

$n!$

etc.

$p! q! r! \dots$

$$G) \binom{7}{4} \times \binom{5}{3} \times 7$$

↓ ↓
selection arrangement.

Probability:

Exp. set, i.e.

Experiment: It is an operation, in which the result or outcome can be predicted with certainty.

Random Exp.: It is an experiment, in which outcome cannot be predicted with certainty.

* If a stone is thrown upwards, definitely in 11 sec come down (Experimental).

Self i.e.

* If a dice thrown once, we may get 1, 2, ..., 6. (Random Experiment).

* Drawing a card from pack of cards (Random).

Def

Sample Space:-

Set of all possible outcomes of random experiment. It is denoted by S.

Event (E):-

Any subset of sample space of random experiment.

Ex. b/w

$${}^3C_1 \times {}^2C_1 \times {}^1C_1$$

4 people.

(i) $(7-1)! = 6!$

(ii) $2! \times (6+1)! = 2! \times 5!$
A, B, C, D, E, F, G

(iii) If 3 persons sit together
A, B, C, D, E, F, G
 $3!$

$3! \times (5-1)! = 3! \times 4!$

3 persons not sitting together $= 6! - 3! \times 4!$
 $= 720 - 144 = 576$

(F) 5R, 2G, 3B

$$\frac{10!}{5! 2! 3!}$$

Notes → We know that n diff. objects can be arranged in a row in $n!$ ways.

$$n! \rightarrow \boxed{{}^nC_1} \times \boxed{{}^{n-1}C_1} \times \dots \times \boxed{{}^2C_1} \times \boxed{{}^1C_1}$$

1 2 n-1 n

Notes But among n objects suppose p no. of objects are of one kind, q no. of objects are of 2nd kind,

D → Out of 5 mathematicians & 7 physicists a committee consisting of 3 mathematicians & 2 physicists is to be formed in how many ways this can be done if:

- ① Any M or any P can be included in committee.
- ② 1 particular M must be in committee always.
- ③ 1 particular P can not be in committee.

E → In how many ways can 7 people be seated at a round table if:

- ① They sit anywhere.
- ② 2 particular people are sitting together always.
- ③ 3 particular people must not sit together.

F → 5 red marbles, 2 green & 3 blue marbles are arranged in a row, if all the marbles of same color are not distinguishable from each other then how many diff. arrang. are possible.

G → From 7 consonants & 5 vowels how many words can be formed, consisting of 4 diff. consonants and 3 diff. vowels

$$A) ① {}^{10}C_4 = \frac{10!}{4!6!} = \frac{10 \times 9 \times 8 \times 7}{4 \times 3 \times 2 \times 1} = 210$$

$$K) ② {}^{10}P_4 = \frac{10!}{6!} =$$

$$③ \frac{7!}{3!} = {}^7P_3$$

$$① \begin{array}{|c|c|c|c|} \hline \downarrow & \downarrow & \downarrow & \downarrow \\ \hline {}^9C_1 \times {}^8C_1 \times {}^7C_1 \times {}^6C_1 \\ \hline = 9 \times 8 \times 7 \times 6 = 3024 \end{array}$$

$$② \begin{array}{|c|c|c|c|} \hline \downarrow & \downarrow & \downarrow & \downarrow \\ \hline {}^9C_1 \times {}^{10}C_1 \times {}^{10}C_1 \times {}^{10}C_1 = 9000 \end{array}$$

$$③ \begin{array}{|c|c|c|c|} \hline \downarrow & \downarrow & \downarrow & \downarrow \\ \hline {}^{10}C_1 \times {}^{10}C_1 \times {}^{10}C_1 \times {}^{10}C_1 = 10000 \end{array}$$

Committee formed

⇒ Let x, y, z 3 diff. items, then we can select any two items in ${}^3C_2 = \frac{3!}{2!1!} = 3$
 $(x, y) (x, z) (y, z)$

Committee

always

⇒ Similarly, after selecting 2 items from 3, if we arrange them in an order, then total no. of arrangements ${}^3P_2 = \frac{3!}{1!} = 6$

if

$(x, y) (y, x) (x, z) (z, x) (y, z) (z, y)$

always

A → How many 4 digit nos. can be formed with 10 digits from 0, 1, 2, ..., 9, if

- 1) Repetitions are not allowed
- 2) Repetitions are allowed
- 3) Last digit must be 7 & repetitions are not allowed
- 4) 4

of some other

B → 4 diff. mathematics books, 5 diff. Physics & 6 diff. Chemis books are to be arranged on a shelf, then how many diff. arrangements are possible, if

ing words

- ① Books in each particular subject must all stand together.
- ② Only physics books must stand together.
- ③ It is required

Q

C → It is required to sit 5 men & 4 women in a row so that, five men occupy odd places, then how many such arrangements are possible?

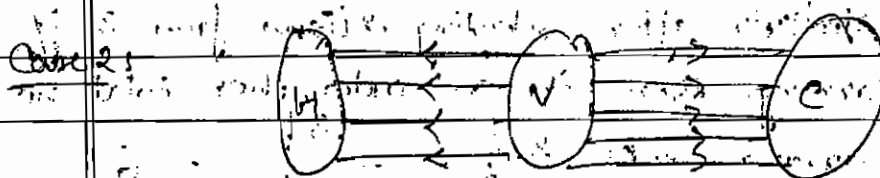
Q

536

7

448

→ Suppose A person at H railway station reaching C, via V i.e. he is doing both activities A, & A₂ then the total no. of ways of reaching C is
 $4 \times 5 = 20$



→ If the person is at V then he can do either A, or A₂ but not both i.e. the total no. of ways of leaving the station V by person
 $4 + 5 = 9$

Combination → Selection

Permutation → Arrangement

Combination:

Let there be 'n' diff. objects available.

Among n objects, a r no. of objects being selected the total no. of such selection given by

$${}^n C_r = \frac{n!}{r!(n-r)!}$$

Permutation:

Suppose n diff. objects & r no. of objects being selected & also arranged in proper order, then total no. of such arrangement

$${}^n P_r = \frac{n!}{(n-r)!}$$

Case 1:

PERMUTATION:

PROBABILITY:

Fundamental Concepts in Counting:-

Product Rule:-

Let there be n activities $A_1, A_2, \dots, A_n$.

Suppose A_1 can be done in m_1 ways, A_2 can be done in m_2 ways & same as A_n can be done in m_n ways, if all the activities are completed to get the job completed.

Total no. of ways of getting the job completed is given by $m_1 \times m_2 \times \dots \times m_n$, This is called product rule.

Sum Rule:-

Suppose among n activities as mentioned above, any 1 activity is over to get the job done i.e.

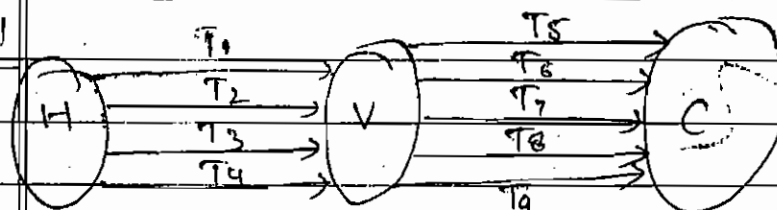
A_1 is completed or A_2 is completed or $\dots$ or A_n is completed then total no. of ways of getting the job done is

$$m_1 + m_2 + m_3 + \dots + m_n.$$

and : $\times$, $\cap$, $\cap$, Cond.

or : $+$, $\cup$, $\cup$, Total

Case : 1



A_1 and A_2
 A_1 : Reaching from H to V
 A_2 : Reaching from V to C.

→ 05:30 - 09:00 : wakeup / Ready / find / hrs study

09:00 - 07:30 : office / Rest

07:30 - 09:30 : Cooking / Dinner

→ 09:30 - 10:00 : Study

11:00 - 10:30 : Sleeping

• The
cer
pec

• Pre
was

• In
rep
wat

• The
Yo
Till

• The
lew
Dur

• The
Per

• The
Cal

• Wh
cor

• The
dur
-19

• The
wa

• The
ins

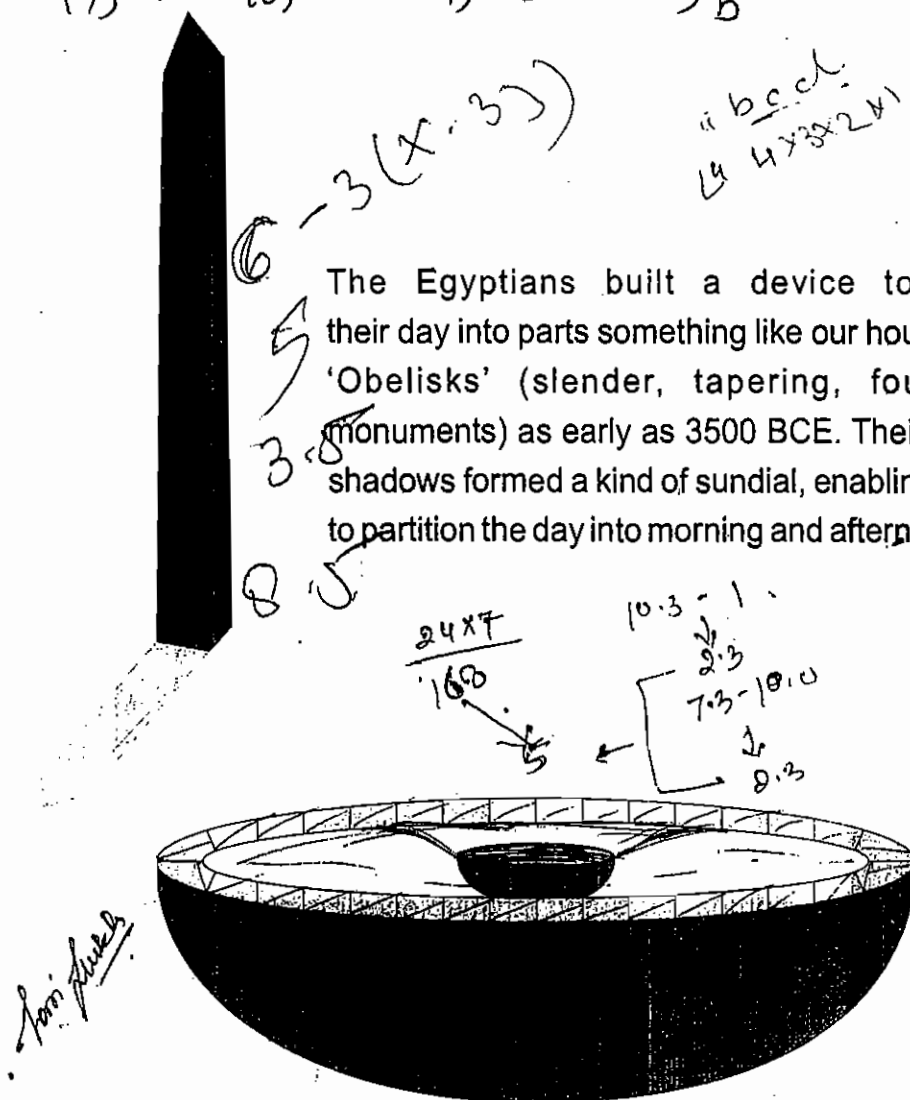
• The
ins

Facts about Watches

- The first pocket watches were invented in Tudor, England, in the 16th century. These pocket watches were incredibly big, which is why people wore them around their neck.
- Breguet was the most preferred watch of the Russian Tsars. Breguet was also worn by Napoleon at Waterloo.
- In 1795, Breguet was the one to invent the tourbillon, which till date represents a great achievement in making time pieces. It compensates watch errors caused by gravity. Even today tourbillon is one of the most sophisticated mechanisms to manufacture.
- The first wristwatch was invented by Patek Philippe. Until the First World War the wristwatch was considered to be a woman's accessory. Till then men wore pocket watches.
- The first wristwatch for men was invented by Cartier. The famous jeweler created a watch for his friend, Brazilian pilot Alberto Santos-Dumont.
- The first mass production of wristwatches was started by Girard Perregaux. The company created its timepieces mainly for military use.
- The Tag Heuer Carrera Chronograph was made in memory of famous Carrera Panamericana automobile races that were held in 1950s.
- When Sir Edmund Hilary climbed Mt. Everest (the first human to conquer the mountain) in 1953 he wore a Rolex Oyster.
- The Russian and American astronauts wore Omega Speedmaster during the world's first craft meeting of Apollo-Soyuz that took place in 1975.
- The world's first anti magnetic watch was created by the famous Swiss watchmaker Tissot.
- The Jantar Mantar at Delhi consists of 13 architectural astronomy instruments, built by Maharaja Jai Singh II of Jaipur, from 1724.

- 1) c 2) c 3) c 4) a 5) a 6) d 8) b
 9) b 10) c 11) d 12) a 13) b 14) c 15) b 16) b
 17) a 18) a 19) a 20) b

Facts about Watches



Water clocks were among the earliest timekeepers that didn't depend on the observation of celestial bodies. They were stone containers designed to slowly fill with water coming in at a constant rate. Markings on the inside surfaces measured the passage of "hours" as the water level reached them. Another version consisted of a metal bowl with a hole in the bottom; when placed in a container of water the bowl would fill and sink in a certain time. These were still in use in North Africa in the 20th century.

100