Name :- Anand Gharu
Desig :- Assistant Prof.
Subject :- TOC
Class :- TE Computer.

COMPUTER ENGINEERING

TOC EXAM NOV/DEC-22 MODEL ANSWER-2022

Q 1.

a) Convert the full grammar to CNF — 09-Marks.

→ Step 0 :- $S \rightarrow a \mid aA \mid B$
$A \rightarrow aBB \mid \epsilon$
$B \rightarrow Aa \mid b$

→ Step 1 :- Removing Null production
Nullable set = $\{S, A\}$
The set of production becomes
$S \rightarrow aA \mid B \mid a$
$A \rightarrow aBB$.
$B \rightarrow Aa \mid b. \mid a$

Step 2 :- Removing Unit Production
$S \rightarrow aA \mid Aa \mid a \mid b$
$A \rightarrow aBB$
$B \rightarrow Aa \mid b \mid a$

Step 3 :- Substitute $C_a \rightarrow a$ and $C_b \rightarrow b$.
$S \rightarrow C_aA \mid AC_a \mid a \mid b$
$A \rightarrow C_aBB$
$B \rightarrow AC_a \mid b. \mid a$

Step 4 :-

Sr No	Rewriting Prod in step-3	Production Equivalent CNF Production
1	$S \rightarrow C_aA \mid AC_a \mid a \mid b$	$S \rightarrow C_aA \mid AC_a \mid a \mid b$
2.	$A \rightarrow C_aBB$	$A \rightarrow C_aC_1$ $C_1 \rightarrow BB$
3.	$B \rightarrow AC_a \mid b$	$B \rightarrow AC_a \mid b$

Step 5 :- Final Productions are
$S \rightarrow C_aA \mid AC_a \mid a \mid b$
$A \rightarrow C_aC_1$
$C_1 \rightarrow BB$
$B \rightarrow AC_a \mid b. \mid a$
$C_a \rightarrow a$
$C_b \rightarrow b.$

Mr. A. N. Gharu

Q.1 b) Convert the following grammar to GNF — 09 Marks

$S \rightarrow XB \mid AA$
$A \rightarrow a \mid SA$
$B \rightarrow b$
$X \rightarrow a$

→ **Step 1 :** As given grammar G is already in CNF and there is no left recursion so we can skip step 1 & 2 & directly goto step-3

The Prod Rule $A \rightarrow SA$ is not in GNF, so we substitute $S \rightarrow XB \mid AA$ in prod rule $A \rightarrow SA$ as

$S \rightarrow XB \mid AA$
$A \rightarrow a \mid XBA \mid AAA$
$B \rightarrow b$
$X \rightarrow a$

The Prod rule $S \rightarrow XB$ and $B \rightarrow XBA$ is not in GNF, so we substitute $X \rightarrow a$ in prod rule $S \rightarrow XB$ and $B \rightarrow XBA$ as.

$S \rightarrow aB \mid AA$
$A \rightarrow a \mid aBA \mid AAA$
$B \rightarrow b$
$X \rightarrow a$

Now We will remove Left Recursion ($A \rightarrow AAA$) we get,

$S \rightarrow aB \mid AA$
$A \rightarrow aC \mid aBAC \mid aBA \mid a$
$C \rightarrow AAC \mid AA$
$B \rightarrow b$
$X \rightarrow a$

We have formula

$A \rightarrow \beta B \mid \beta$
$B \rightarrow \alpha B \mid \alpha.$

But In this example, B is already given so Re-write formula

Beta

$A \rightarrow \beta C \mid \beta$
$C \rightarrow \alpha C \mid \alpha$ → Modified formula.

C → AA is not in GNF.

The Prod S → AA is not in GNF, So we substitute A → aBAC | aC | aBA | a in prod S → AA as.
C → AA as

S → aB | aBACA | aCA | aBAA | aA
A → aBAC | aC | aBA | a
C → AAC
C → aBACA | aCA | aBAA | aA ← C → AA Substitute A prod.
B → b
X → a

The Prod C → AAC is not in GNF, so we substitute A → aBAC | aC | aBA | a in prod C → AAC as.

S → aB | aBACA | aCA | aBAA | aA
A → aBAC | aC | aBA | a
C → aBACAC | aCAC | aBAAC | aAC. C → AAC Substitute A prod
C → aBACA | aCA | aBAA | aA
B → b
X → a.

Re-Write final GNF grammer, we get.

S → aB | aBACA | aCA | aBAA | aA
A → aBAC | aC | aBA | a
C → aA | aAC | aCA | aBAA | aBACA
aBAAC | aCAC | aBACAC.
B → b
X → a.

Hence, this is in GNF form for the grammar G.

Note : To remove left recursion, ε rule is given on Internet But I have not referred that formula

I refered = | A → BC | B | To Remove
C → αC | α | Left Recursion

By Mr A.N. Gharu

OR

Q.2 a) Show that the foll. grammar is ambiguous

$S \rightarrow iCtS$

$S \rightarrow iCtSeS$

$S \rightarrow a$

$C \rightarrow b$

06-Marks

Ans → Using the grammar, it is possible to derive string ibtibtaea using two derivation as shown in fig. a and fig b

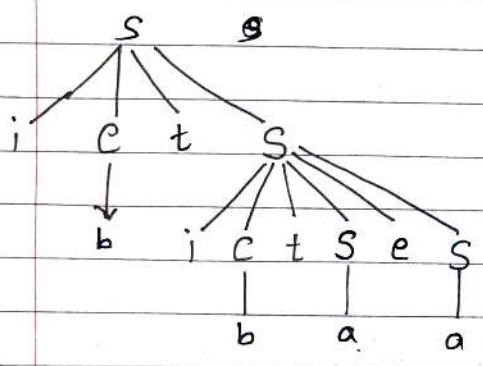


Fig (a)

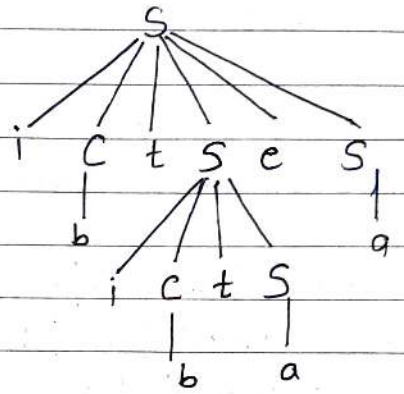


Fig (b)

Hence, grammar is ambiguous.

b) Convert the foll. grammar to CNF

$G = (\{S\}, \{a, b\}, P, S\}$

$P = \{S \rightarrow aSa \mid bSb \mid a \mid b \mid aa \mid bb\}$

→ Step 1 ÷ $S \rightarrow aSa \mid bSb \mid a \mid b \mid aa \mid bb$. 6-Marks

Remove Simplification

There is no simplification

Step 2 ÷ Substitute $A \rightarrow a$ & $B \rightarrow b$.

$S \rightarrow ASA \mid AA \mid a$

$S \rightarrow BSB \mid BB \mid b$.

Step 3 ÷ Convert to CNF.

$S \rightarrow ASA \mid BSB$

$C_1 \rightarrow SA$

$C_2 \rightarrow SB$

Final production are

$S \rightarrow AC_1 \mid BC_2 \mid a \mid b$.

$C_1 \rightarrow SA$

$C_2 \rightarrow SB$

$S \rightarrow AA \mid BB \mid a \mid b$.

c] Consider the foll. grammar. Derive string id-id*id
$E \rightarrow E+E \mid E-E \mid id$ 1) Leftmost and Rightmost Derivation?

→ Student may derive half parse tree with E-E And Can declare that given string is not possible to derive using given grammar. 6-Mark

OR

Some students may rectify grammar assuming that E+E should be E-E and can derive given string by Complete parse tree.

Note: Assume E+E as E*E

Rewrite grammar $E \rightarrow E+E$, $E \rightarrow E*E$, $E \rightarrow id$ string id-id*id

→ Leftmost derivation

$E \rightarrow E*E$	1st prod
$\rightarrow E-E*E$	$E \rightarrow E-E$
$\rightarrow id-E*E$	$E \rightarrow id$
$\rightarrow id-id*E$	$E \rightarrow id$
$\rightarrow id-id*id$	$E \rightarrow id$

Rightmost derivation.

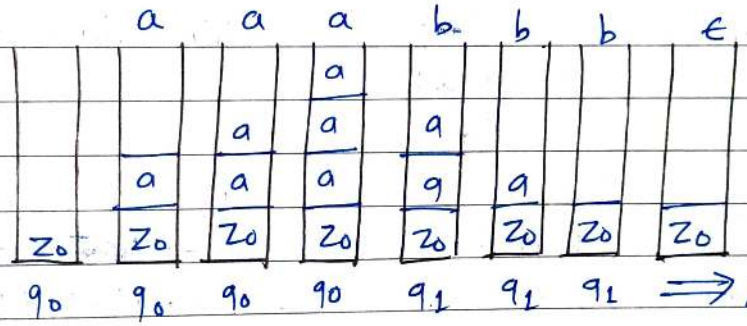
$E \rightarrow E-E$	
$E \rightarrow E-E*E$	$E \rightarrow E*E$
$E \rightarrow E-E*id$	$E \rightarrow id$
$E \rightarrow E-id*id$	$E \rightarrow id$
$E \rightarrow id-id*id$	$E \rightarrow id$

Q.3 a) Find the transition rule of PDA for accepting a lang. $L = \{w \in \{a,b\}^* \mid w \text{ is of the } a^n b^n \text{ with } n \geq 1\}$ through both empty and final state and demonstrate the stack oper for string aaabbb.

→ -9 Mark

- Transition Rules :

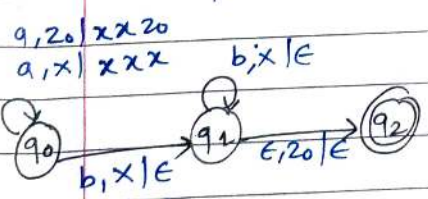
1) $\delta(q_0, a, Z_0) = (q_0, aZ_0)$
2) $\delta(q_0, a, a) = (q_0, aa)$
3) $\delta(q_0, b, a) = (q_1, \epsilon)$
4) $\delta(q_1, b, a) = (q_1, \epsilon)$
5) $\delta(q_1, \epsilon, Z_0) = (q_1, \epsilon)$ Through Empty stack
6) $\delta(q_1, \epsilon, Z_0) = (q_2, Z_0)$ Through final state.



b) Design a PDA for accepting Lang. $\{a^n b^{2n} \mid n >= 1\}$ Simulate this PDA for the input string aaabbbbbb. 9 Marks

→ 1. For every a in $a^n b^{2n}$, xx are pushed on the stack
2. For every b in $a^n b^{2n}$, one x is popped from stacks.

The required PDA is



1) $\delta(q_0, a, Z_0) = (q_0, xxZ_0)$
2) $\delta(q_0, a, x) = (q_0, xxx)$
3) $\delta(q_0, b, x) = (q_1, \epsilon)$
4) $\delta(q_1, b, x) = (q_1, \epsilon)$
5) $\delta(q_1, \epsilon, Z_0) = (q_2, \epsilon)$

Simulation of aaabbbbbb

Tracing $a^3 b^6$ PDA

$\delta(q_0, aaabbbbbb, Z_0) \overset{R_1}{\Rightarrow} (q_0, aabbbbbb, xxZ_0)$
$\Rightarrow (q_0, abbbbbb, xxxZ_0)$
$\Rightarrow (q_0, bbbbbb, xxxxxZ_0)$
$\Rightarrow (q_0, bbbbb, xxxxxZ_0)$
$\Rightarrow (q_1, bbbb, xxxxZ_0)$
$\Rightarrow (q_1, bbb, xxxZ_0)$
$\Rightarrow (q_1, bb, xxZ_0)$
$\Rightarrow (q_1, b, xZ_0)$
$\Rightarrow (q_1, \epsilon, Z_0)$
$\Rightarrow (q_2, \epsilon, \epsilon)$

9-Marks.

Q.4 a) Design a PDA for accepting lang $\{0^n 1^m 0^n \mid m,n \geq 1\}$
Simulate this PDA for the input string "0011100"

→ In this PDA, n number of 0's are followed by any number of 1's followed by any numbers of 0's.
Hence, the logic design of such PDA will be as follow,

Push all 0's on to stack on encountering 1st 0's.
Then, if we read 1 just do nothing Then read 0 and on each read of 0's pop one 0' from the stack.

The PDA are. →

1) $\delta(q_0, 0, Z_0) = (q_0, 0Z_0)$
2) $\delta(q_0, 0, 0) = (q_0, 00)$
3) $\delta(q_0, 1, 0) = (q_1, 0)$
4) $\delta(q_1, 1, 0) = (q_1, 0)$
5) $\delta(q_1, 0, 0) = (q_2, \epsilon)$
6) $\delta(q_2, 0, 0) = (q_2, \epsilon)$
7) $\delta(q_2, \epsilon, Z_0) = (q_2, \epsilon)$

Simulation of string 0011100.

$\delta(q_0, 0011100, Z_0)$
$\xrightarrow{R_1}$ $(q_0, 011100, 0Z_0)$
$\xrightarrow{R_2}$ $(q_0, 11100, 00Z_0)$
$\xrightarrow{R_3}$ $(q_0, 1100, 00Z_0)$
$\xrightarrow{R_4}$ $(q_1, 100, 00Z_0)$
$\xrightarrow{R_4}$ $(q_1, 00, 00Z_0)$
$\xrightarrow{R_5}$ $(q_2, 0, 0Z_0)$
$\xrightarrow{R_6}$ (q_2, ϵ, Z_0)
$\xrightarrow{R_7}$ $(q_2, \epsilon, \epsilon)$ //
Accepted

Q.4. b) Construct PDA for $L = \{0^n 1^m 2^m 3^n \mid m, n >= 0\}$ 6-Marks

→ There can be four cases

Case - 1 = $n = 0$

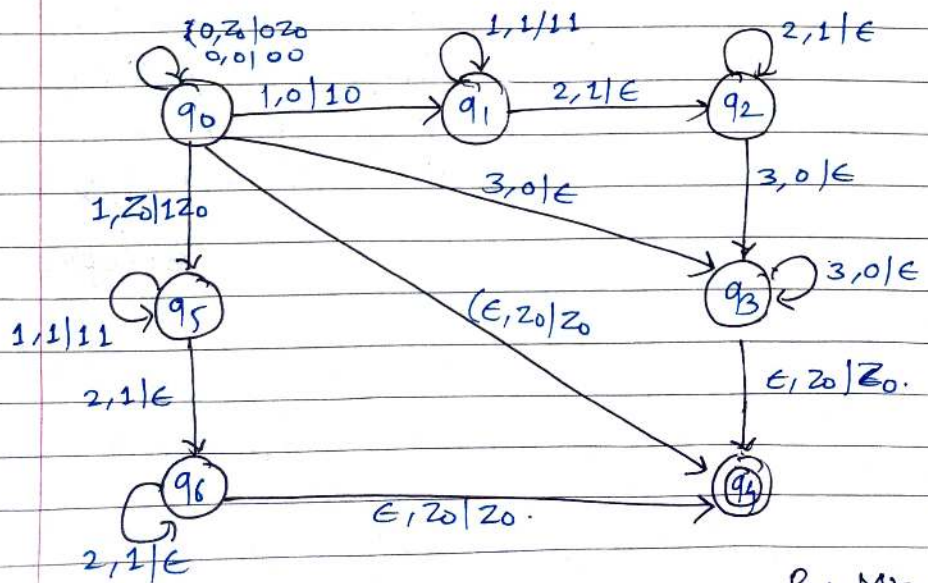
Case - 2 = $m = 0$

Case - 3 = $m, n > 0$

Case - 4 = $m = 0, n = 0$.

Final Pushdown Automata can be :

1) $\delta(q_0, 0, Z_0) = (q_0, 0Z_0)$ } push number of 0's
2) $\delta(q_0, 0, 0) = (q_0, 00)$ }
3) $\delta(q_0, 1, 0) = (q_1, 10)$ } push number of 1's
4) $\delta(q_1, 1, 1) = (q_1, 11)$ }
5) $\delta(q_1, 2, 1) = (q_2, \epsilon)$ } pop number of 2's
6) $\delta(q_2, 2, 1) = (q_2, \epsilon)$ }
7) $\delta(q_2, 3, 0) = (q_3, \epsilon)$ } pop number of 3's
8) $\delta(q_3, 3, 0) = (q_3, \epsilon)$ }
9) $\delta(q_3, \epsilon, Z_0) = (q_4, Z_0)$ } For final state
10) $\delta(q_0, 1, Z_0) = (q_5, 1Z_0)$ } if $n = 0$. push 1's
11) $\delta(q_5, 1, 1) = (q_5, 11)$ }
12) $\delta(q_5, 2, 1) = (q_6, \epsilon)$ } & pop 2's
13) $\delta(q_6, 2, 1) = (q_6, \epsilon)$ }
14) $\delta(q_6, \epsilon, Z_0) = (q_4, Z_0)$ // For final state if $n = 0$
15) $\delta(q_0, 3, 0) = (q_3, \epsilon)$ // if $m = 0$. pop 3's
16) $\delta(q_0, \epsilon, Z_0) = (q_4, Z_0)$ // if $m = 0$ and $n = 0$.



By Mr. A. N. Ghogre

Q.4 c) Difference between FA and PDA. 3-Marks.

→

Sr. No	FA	Sr. No	PDA
1.	FA stands for Finite Automata	1.	PDA stands for Pushdown Automata.
2.	FA has no memory	2.	PDA has memory.
3.	It accept Regular lang.	3.	It accept regular, Non-regular and Context free Language.
4.	It is simple to represent	4.	It is must powerful than FA.
5.	It makes use of states & transition to model any lang.	5.	It makes use of Push & Pop oper to model any language.

Q.5 a) Write a short note on Halting Problem of TM 4-Marks.

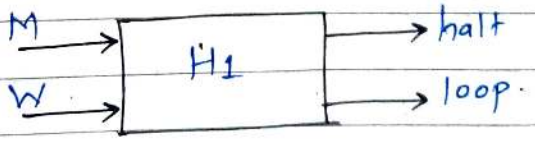
→ (1) Let's assume that the halting Problem is Solvable.

→ There exists machine H2 (says) H1 takes two input.

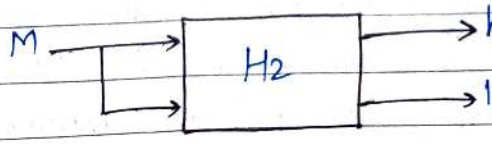
1. A string describing M.
2. An input w for Machine M.

H1 generates Output "halt" if H1 determines that M. stops on input w otherwise H Output "loop"

Working of Machine H1 shown below :-



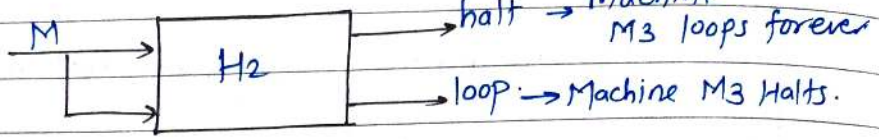
(2) Let's revise the Machine H1 as H2 to take M as both Input. H2 should be able to determine if M will halt on M as it's input. Note that, Machine can be described as a string Over 0 and 1.



By A. N. Ghare

③ Let's construct new TM H_3 that takes output of H_2 as input & does the following:

1) if the output of H_2 is "loop" then H_3 Halts
2) if the output of H_2 is "Halt then H_3 will loop forever



H_3 will do the Opposite of H_2.

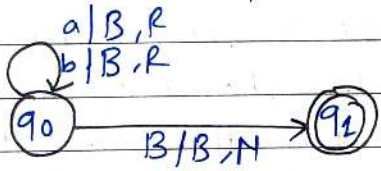
b) Design TM for the foll. lang. by Considering transition table and diagram.

i) TM that erases all Non-blank symbol on the tape where the sequence of non-blank symbol does not contain any blank symbol B in between

ii) TM that find 2's Complement of a Binary number

— 09-Marks.

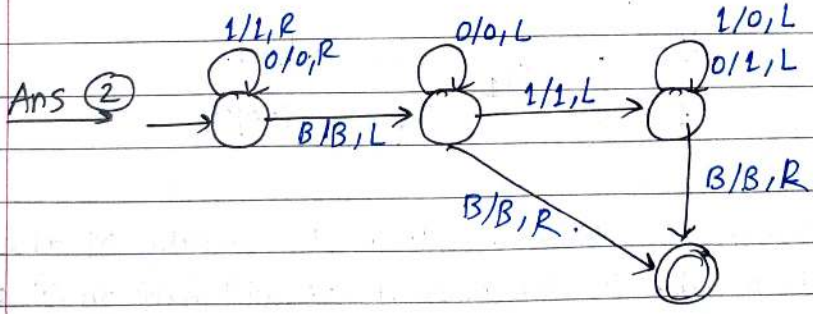
Ans (1)



Transition fn is defined as: →

$$\delta(q_0, a) = (q_0, B, R)$$
$$\delta(q_0, b) = (q_0, B, R)$$
$$\delta(q, B) = (q_1, B, N)$$

Ans (2)



Logic :- 2's Complement of Binary number can be found by not changing bits from right end till the first 1's and the complementing remaining bits of Binary number.

e.g. 010110 1000 ⟹ 1010011000

(Complement every bits) (no change)

Q.5 c] Design TM that reads a string representing a binary number and erases all leading 0's in the string However. if the string comprises of only 0's, it keeps one 0. 5-Mark

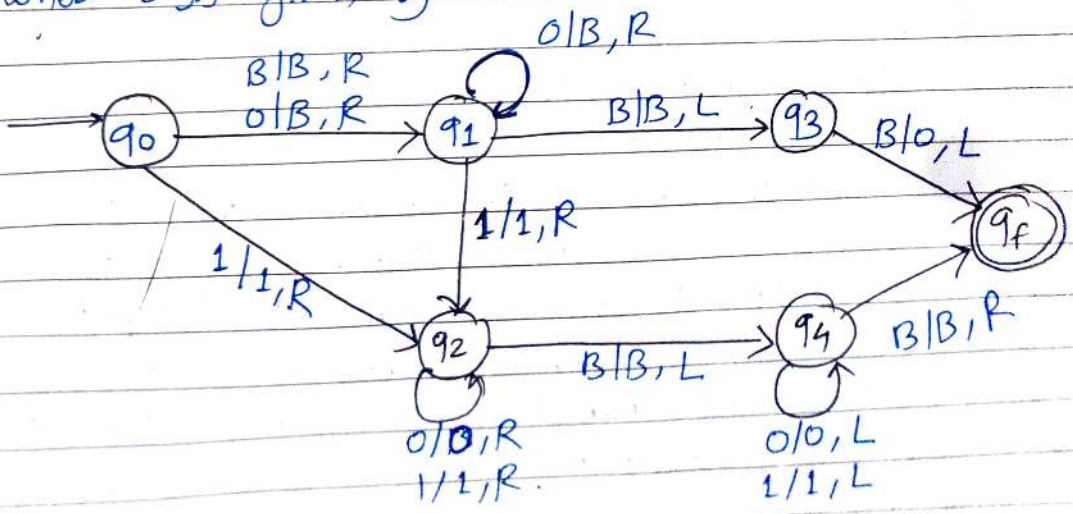
→ Let's assume that the input string is terminated by a Blank Symbol B. at each end of string.

The TM M can be Constructed by the foll moves -

- Let q_0 be the inital state
- if M is in q_0, On reading 0, it moves right, enters the state q_1 and erases 0. On reading 1, it enters the state q_2 and moves right.
- If M is in q_1, on reading 0, it moves right and erases 0. i.e. it replaces 0's by B's. On reaching the leftmost 1, It enters q_2 and moves right. if it reaches B, i.e. the string comprises of only 0's, it moves left and enters the state q_3.
- If M is q_2 on reading either 0 or 1, it moves right. On reaching B, it moves left and enters the state q_4. This validate that the string Comprises only 0's and 1's.
- If M is in q_3, it replaces ~~by~~ B by 0, moves left and reaches the final state q_f.
- If M is in q_4, on reading either 0 or 1, it moves left. On reaching the beginning of string i.e. when it reads B., it reaches the final state q_f.

Hence, $M = \{(q_0, q_1, q_2, q_3, q_4, q_f), (0, 1, B), (1, B), \delta, q_0, B, \{q_f\}\}$

where δ is given by :



OR

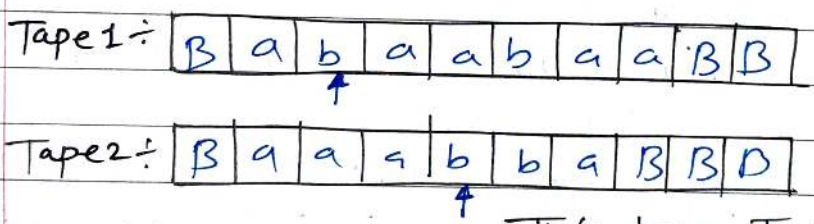
Q.6 a) Write short notes on — 04-Mark
1) Reducibility
2) Multi-tape TM.

Ans→ 1) Reducibility :-

→ - There are Some problem solvable and Unsolvable Problem
- If we can establish that one decision Problem, P_1 can be reduced to another P_2 or that having a general solution to P_2 would guranee a general solution to P_1 then it is reasonable to say informally that P_1 is no harder than P_2. It should then follow that if P_2 is solvable, P_1 is solvable.
(or equivalently, if P_1 is unsolvable, then P_2 is Unsolvable)
- It is possible to reduce one decision problem to another or one lang to another.

ii) Multi-tape TM :-

A Multi-tape TM has Multiple tuple with each tape having its own independent head. Let's consider the case of two tape TM. It is shown in fig



Two-tape Turing Machine

The transition behaviar of two tape TM can be defined as

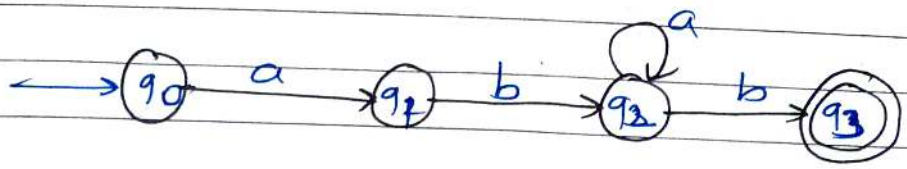
$$\delta(q_1, a_1, a_2) = (q_2 (S_1, M_1), (S_2, M_2))$$

where,

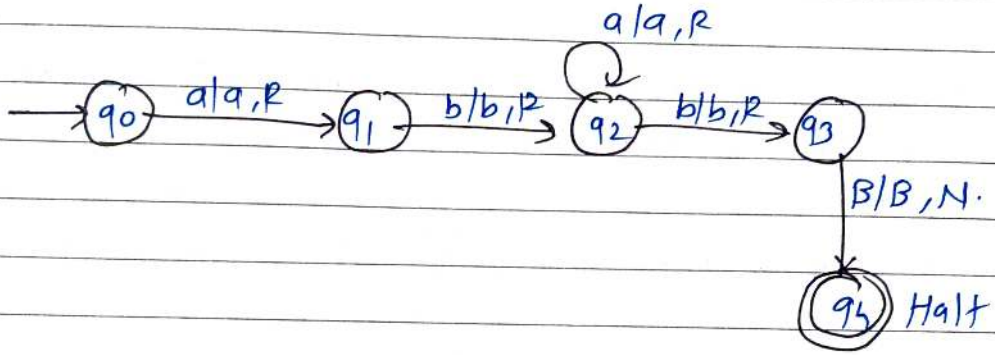
q_1 is current state
q_2 is next state
a_1 is the symbol under the head on tape 1
a_2 is the symbol under the head on tape 2
S_1 is symbol written in current cell on tape 1
S_2 is symbol written in current cell on tape 2
M_1 is the movement (L, R, N) of head on tape-1
M_2 is the movement (L, R, N) of head on tape-2

Q.6 b) Construct a TM for $R = aba^*b$ — 6-Marks

→ Step 1 : DFA for aba^*b.



Step 2 : DFA to TM.



The TM for the above lang. is given by

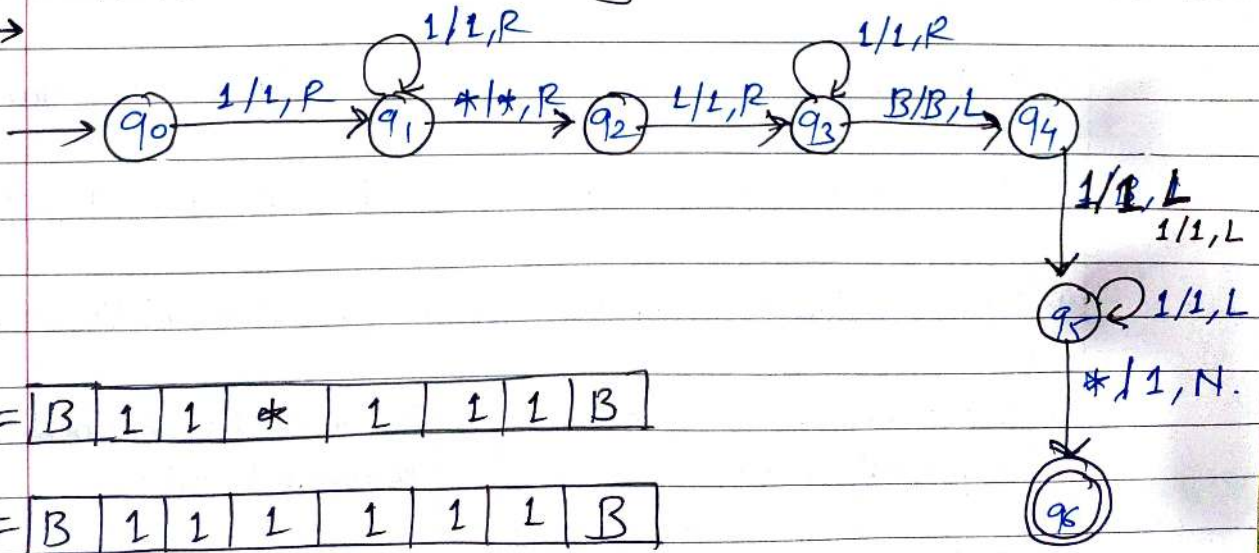
$M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$

where,

$Q = \{q_0, q_1, q_2, q_3, q_4\}$
$\Sigma = \{a, b\}$
$\Gamma = \{a, b, B\}$

δ = tran^n fu^n
q_0 = Initial state
B = Blank Symbol
F = q_4 Halt

c) Design TM that multiplies two Onary number over {1}. Write simulation for string 11*111 — 8 Marks.



Input = | B | 1 | 1 | * | 1 | 1 | 1 | B |

Output = | B | 1 | 1 | 1 | 1 | 1 | 1 | B |

By . A. N. Ghosh

Q.7 a) Justify "Halting Problem of TM is Undecidable"? 8-Marks

→ A general algo. running on a TM that solves the halting problem for all possible-program-input pair necessarily can't exist.

Hence, the Halting problem is Un-decidable for TM.

Note: Refer Q.5 a) Halting problem

And example (solve) $L=\{a^n b^n \mid n \geq 1\}$

b) Define and Compare class P and NP Problem with Suitable example 8-Marks

→ P class: The P class stands for Polynomial Time It is the collection of decision problem (Problem with yes or no answer) That can be solved by deterministic machine of polynomial time.

NP Class: NP in NP class stands for Non-deterministic Polynomial time. It is the collection of decision problem that can be solved by Non-deterministic machine in polynomial time.

Sr No	P Class	Sr No	NP Class
1.	It stands Polynomial time	1.	It stands for Non-deterministic Polynomial time
2.	It is easy to understand	2.	It is difficult to understand
3	It is subset of NP Problem	3.	It is superset of P Problem
4.	It is deterministic in nature	4.	It is non-deterministic in nature.
5	It can be obtained in Polynomial time.	5.	It can't be obtained in Polynomial time
6.	Examples: Linear Search, Selection Sort etc	6.	Examples: Knapsack problem Travelling salesman Problem etc

OR

Q-8 a] Explain in brief term "Recursively enumerable?" 06-Mark

→ 1) A language is called Recursively enumerable if and only if the language is accepted by some Turing Machine M.

2) In other word, L is said to be L = L(M) for some TM. There are certain lang. is not recursively enumerable

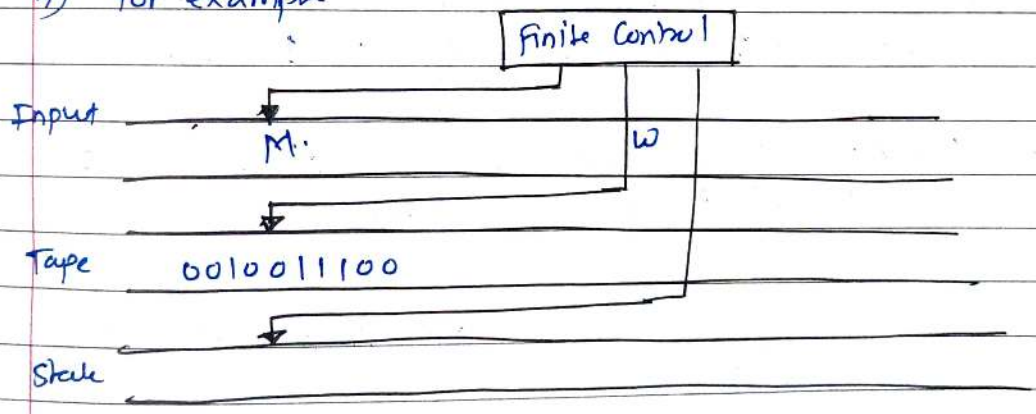
3) Conside lang consisting of pair (M, w)
Where, 1) M, a TM with input set (0, 1)
2) w, a binary nos consisting of 0's & 1's
3) M accept w

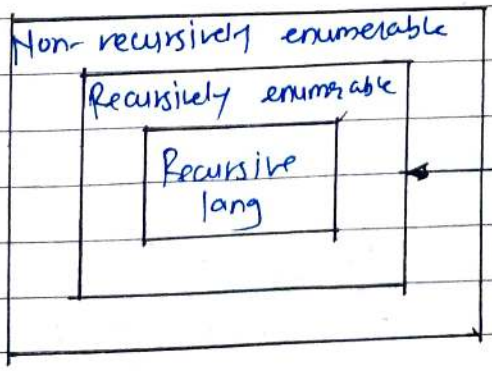
4) The foll. statements are equivalent
- L is turing Acceptable.
- L is recursively enumerable.
- L is recursive
- L is Turing Decidable.

5) Every Turing decidable lang is turing acceptable

6) Every Turing Acceptable lang need not be turing decidable

7) For example





This shows that every recursively enumerable set has recursive subset.

* Recursive = Repeatation of Same language
* Enumerable = Repeata of list of element.

Q.8 b) Explain Example of Problem in NP. — 06 Mark

→ Example of NP-Complete : SAT

"The satisfiability problem is :

"Given a Boolean expression, is it satisfiable?

→ A Boolean expression is said to be satisfiable if at least one truth assignment makes the Boolean expre true"

The Boolean expression are created using.

1) The variable whose value can be 0 or 1.
2) The operator that can be used in expression can be $\vee$, $\wedge$. The $\vee$ means OR and $\wedge$ means And operation.
3) Unary operator $\sim$ stands for negation.
4) The parenthesis are used to group the operand and operator in the expression.

The highest precedence is to $\sim$ then $\wedge$ then $\vee$.

For Example :

① The Boolean expre $((x_1 \wedge x_2) \vee \sim x_3)$ is true for $x_1 = 1$, $x_2 = 0$ and $x_3 = 0$.

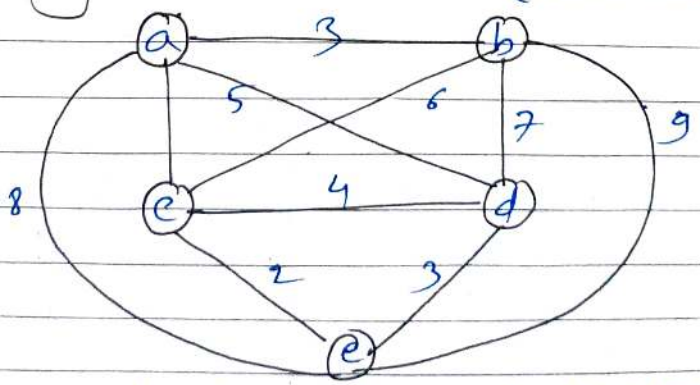
therefore $(\cdot(x_1 \wedge x_2) \vee \sim x_3)$ is satisfiable

$\leftrightarrow$ $(1 \wedge 0) \vee \sim 0) = 1$ (true)

② $(x \wedge y \wedge z)$ is satisfiable when x, y & $z = 1$.

But $x \wedge (\sim y)$ is not satisfiable

2) Travelling Salesman Problem : (Write theory of TSP)



The tour a-b-d-e-c-a and total cost will be 16.

c] Compare P class and NP class. — 04 Mark.

→ Note : Refer Q.7 b.

By Mr. A.N. Ghare.