UNIT - 4 - Notes

Formal Language And Automata (SRM Institute of Science and Technology)

UNIT - IV

TURING MACHINES

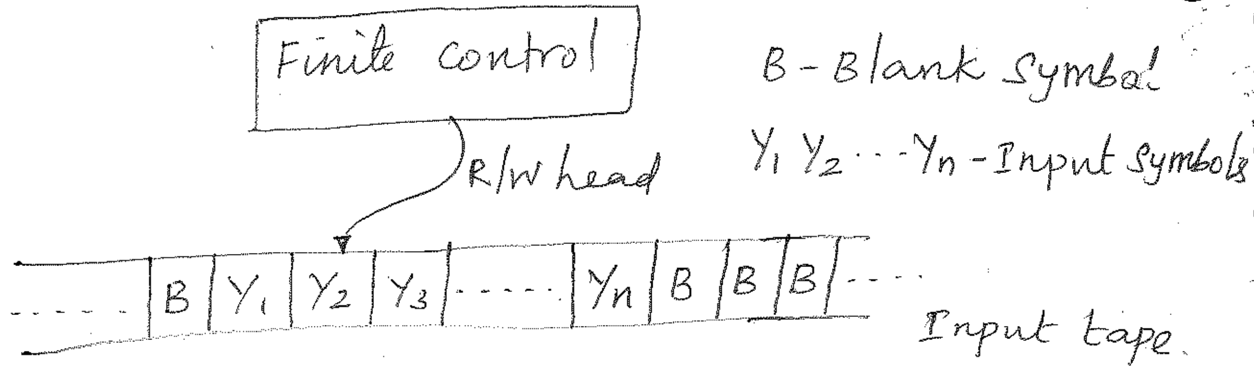
Introduction :

- Alan Turing introduced a new mathematical machine called Turing Machine during the year 1936.
- It is a tool for studying the computability of mathematical functions.
- Finite Automata has finite memory but Turing Machine has infinite tape Memory.
- PDA has infinite memory and access in LIFO order but Turing Machine has infinite memory and there is an head can move either left or right direction to access the input from the tape.

Model of Turing Machine:-

- Turing Machine has a finite control which contains the set of states and transition between the states.
- It has an input tape that is divided into cells and each cell can hold any one of a finite number of symbols over alphabet.
- It has a tape head that scans one cell on the input tape at a time.

* The block diagram of the Turing Machine is given below,



<u>Working of Turing Machine</u> :-

- Input string is initially placed in the input tape. All other tape cells extending infinitely to the left and right of the input tape contains the special symbol called Blank symbol.
- Tape head is positioned at one of the tape cell.
- Initially the tape head points at the left most cell of the input tape.

<u>Formal definition of a Turing Machine</u> :-

Turing Machine M can be defined as,

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

where Q - set of states

Σ - set of input symbols

Γ - set of Tape symbols

δ - Transition function

q_0 - starting state

B - Blank symbol

F - set of final states.

Transition Function of Turing Machine :-

Transition function of the Turing Machine is of the form,

$$\delta(q, X) = (P, Y, D) \text{ where}$$

q - current state in Q

X - Tape Symbol in Γ

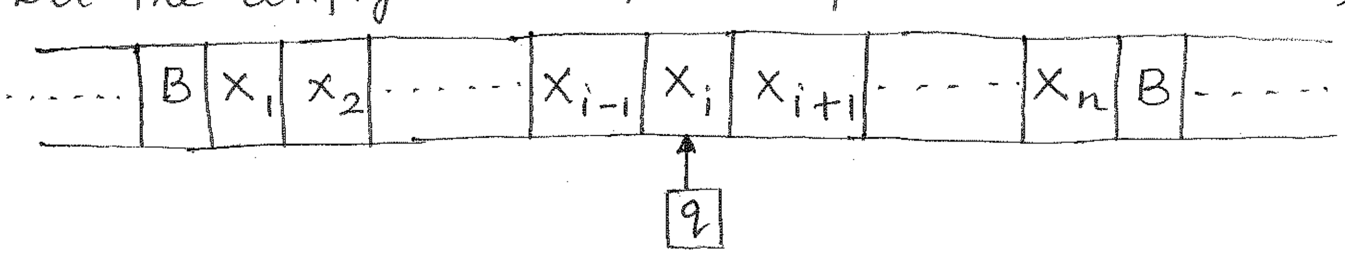
P - changed state in Q

Y - symbol replacing the scanned one. ie X is replaced by $Y \in \Gamma$.

D - Direction of move (Left or Right).

Instantaneous Descriptions of a Turing Machine :-

- Execution sequence of an input string is represented by the instantaneous descriptions of a Turing Machine.
- Each move of the Turing Machine is represented by the instantaneous description.
- Let the configuration of the tape is shown below,



If the transition function of the Turing Machine is $\delta(q, X_i) = (P, Y, L)$. This move can be represented in the ID is,

$$X_1 X_2 \cdots X_{i-1} q X_i X_{i+1} \cdots X_n \vdash X_1 X_2 \cdots P X_{i-1} Y X_{i+1} \cdots X_n$$

Assume the transition as $\delta(q, x_i) = (P, Y, R)$. This move can be represented in the ID is,

$$X_1 X_2 \cdots X_{i-1} q X_i X_{i+1} \cdots X_n \vdash X_1 X_2 \cdots X_{i-1} Y P X_{i+1} \cdots X_n$$

Language of a Turing Machine :-

The Language accepted by the Turing Machine M is defined as L(M) and it is denoted by,

$$L(M) = \{ w \mid w \in \Sigma,\ q_0 w \overset{*}{\underset{M}{\vdash}} \alpha_1 P \alpha_2 \text{ for some } P \text{ in } F \}$$

Transition diagram of a Turing Machine :-

The pictorial representation of the transition functions of the turing machine is called Transition Diagram. It consists of a set of nodes that represents the states, Transition is represented as arc from one state to another state and it is labelled by the form of X|Y, D where X - current Tape symbol, Y - Replacing Tape symbol, D - Direction.

$$q \xrightarrow{X|Y,\rightarrow} P$$

Transition Table :-

Table is constructed in which row is represented as states and column is represented as input symbol which is present in the Tape along with Blank symbol. Entry in the table become the changed state, replacing tape symbol and the direction.

	X
q	(P, Y, →)

Computing Functions by using Turing Machine:-

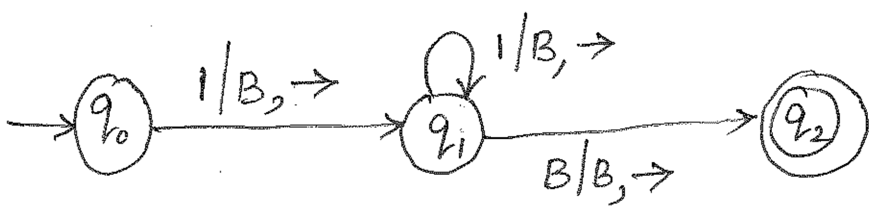
Functions can be computed by using Turing Machine.

Prob1: Design a Turing Machine to process zero function such as $f(x) = 0$ where x is the input.

Sol

Initial configuration of the tape is x number of 1's. our TM Design has to read each 1 and replace it by Blank. For example $x=5$, the tape is initially

1	1	1	1	1	B	B	...



Transition Diagram

Transition Table is,

	1	B
q_0	$(q_1, B, \rightarrow)$	-
q_1	$(q_1, B, \rightarrow)$	$(q_2, B, \rightarrow)$
q_2	-	-

Tracing the string 11111 is,

$q_0 11111B \vdash Bq_1 1111B \vdash BBq_1 111B \vdash BBBq_1 11B$

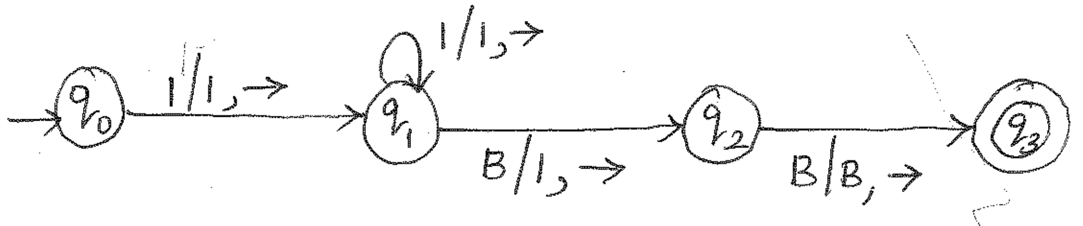
$\vdash BBBBq_1 1B \vdash BBBBBq_1 B \vdash BBBBBBq_2 B$ EF.

The TM $M = (\{q_0, q_1, q_2\}, \{1\}, \{1, B\}, \delta, q_0, B, \{q_2\})$

Prob 2: Design the Turing Machine to implement the function $f(x) = x+1$.

Sol

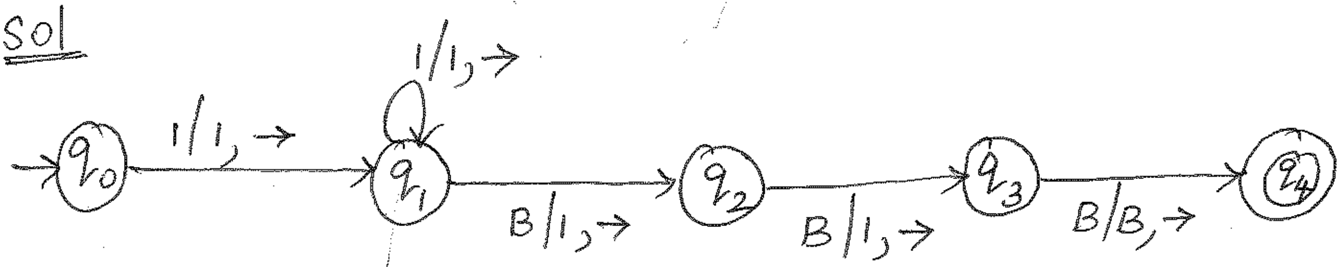
Initially the tape is having x number of 1's and the result is $x+1$ number of 1's in the tape.



TM $M = (\{q_0, q_1, q_2, q_3\}, \{1\}, \{1, B\}, \delta, q_0, B, \{q_3\})$.

Prob 3: Design the Turing Machine to implement the function $f(x) = x+2$.

Sol



TM $M = (\{q_0, q_1, q_2, q_3, q_4\}, \{1\}, \{1, B\}, \delta, q_0, B, \{q_4\})$.

Turing Machine for Integer Functions :-

The following are the some of the Integer functions which can be computed by the Turing Machine,

→ Addition

→ Subtraction

→ Multiplication. [see it in the subroutine heading]

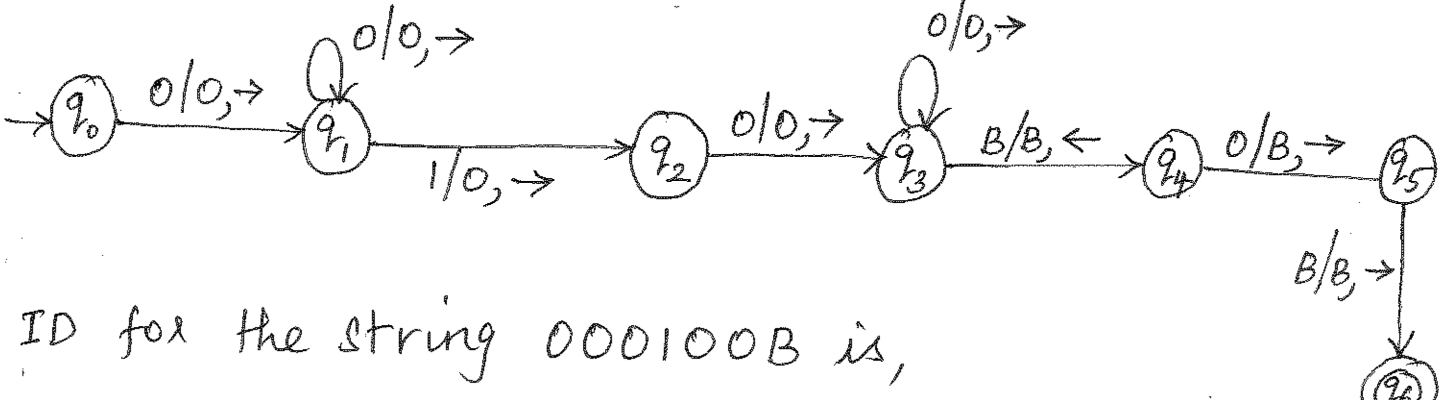
Addition:

This operation requires two arguments.

Problem: Design the turing Machine to perform Addition function operation $f(m,n) = m+n$.

Sol

Initial configuration of the tape is | m | 1 | n | B | ...
Here the 1 is used as a seperator. m and n are the integer value which is represented in the tape as countable 0's. For example $m=3, n=2$ the tape can be | 0 | 0 | 0 | 1 | 0 | 0 | B | .... The Idea of the design is, the 1 is converted to 0 and the last 0 is converted into B symbol.



ID for the string 000100B is,

$q_0 000100B \vdash 0q_1 00100B \vdash 00q_1 0100B \vdash 000q_1 100B$
$\vdash 0000q_2 00B \vdash 00000q_3 0B \vdash 000000q_3 B \vdash$
$00000q_4 0B \vdash 00000Bq_5 B \vdash 00000BBq_6 B \in F.$

TM $M = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}, \{0,1\}, \{0,1,B\}, \delta, q_0, B, \{q_6\})$

Subtraction:

This operation also requires two arguments.

Problem: Design the turing Machine to perform subtraction function operation $f(m,n) = m$

Sol: Based on the input, the subtraction operation can be of two types, one is proper subtraction, the other one is improper subtraction.

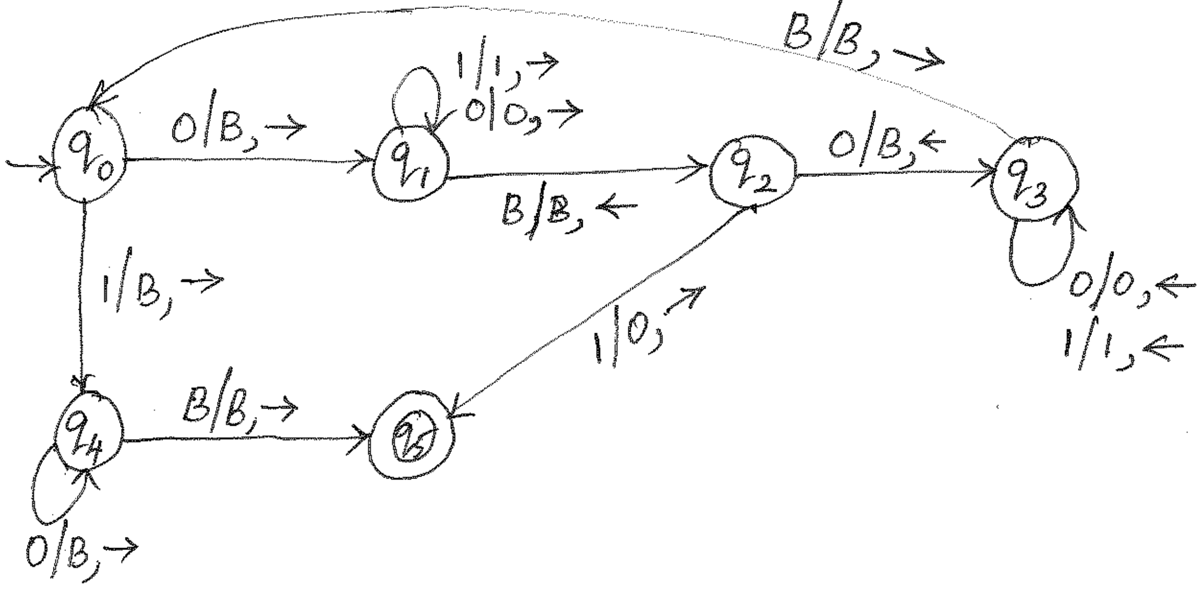
if $m > n$ proper subtraction $\Rightarrow m-n$

if $m \leq n$ improper subtraction $\Rightarrow 0$ (cells are Blank)

Initial configuration of the tape is | m | 1 | n | B | ...

For example $m=3$, $n=2$, the tape is | 0 | 0 | 0 | 1 | 0 | 0 | B | ...

The Idea is, every 0 in the n, the corresponding 0 in the m is cancelled and it is repeatedly called until no more 0's in the n for proper case. For improper case all the excess of 0's in the n is converted into B symbol.



TM $M = (\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{0,1\}, \{0,1,B\}, \delta, q_0, B, \{q_5\})$.

δ can be, $\delta(q_4, 0) = (q_4, B, \rightarrow)$ $\delta(q_4, B) = (q_5, B, \rightarrow)$

$\delta(q_1, B) = (q_2, B, \leftarrow)$ $\delta(q_3, 0) = (q_3, 0, \leftarrow)$

$\delta(q_0, 0) = (q_1, B, \rightarrow)$ $\delta(q_3, 1) = (q_3, 1, \leftarrow)$

$\delta(q_2, 0) = (q_3, B, \leftarrow)$

$\delta(q_1, 0) = (q_1, 0, \rightarrow)$ $\delta(q_3, B) = (q_0, B, \rightarrow)$

$\delta(q_2, 1) = (q_5, 0, \rightarrow)$

$\delta(q_1, 1) = (q_1, 1, \rightarrow)$ $\delta(q_0, 1) = (q_4, B, \rightarrow)$

Downloaded by Harendra Sharma (harendra.pth@gmail.com)

Turing Machine for computing Languages:-

Turing machine is used to validate the string of the Language.

Prob 1: Design a Turing machine for the Language $L = \{1^n \mid n \text{ is even}\}$.

Sol

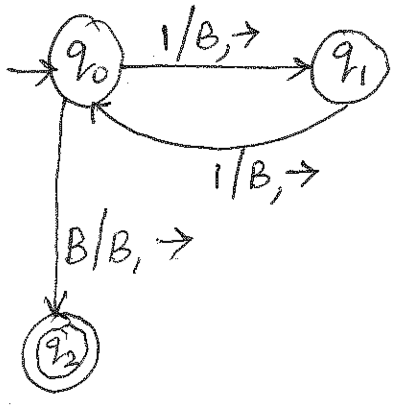


Table:-

	1	B
q_0	$(q_1, B, \rightarrow)$	$(q_2, B, \rightarrow)$
q_1	$(q_0, B, \rightarrow)$	—
q_2	—	—

ID for the string 1111

$$q_0 1111B \vdash Bq_1 111B \vdash BBq_0 11B \vdash BBBq_1 1B \vdash BBBBq_0 B$$

$$\vdash BBBBBq_2 B \in F.$$

TM $M = (\{q_0, q_1, q_2\}, \{1\}, \{1, B\}, \delta, q_0, B, \{q_2\})$.

Prob 2: Design a Turing Machine for the Language $L = \{1^n \mid n \text{ is odd}\}$.

Sol

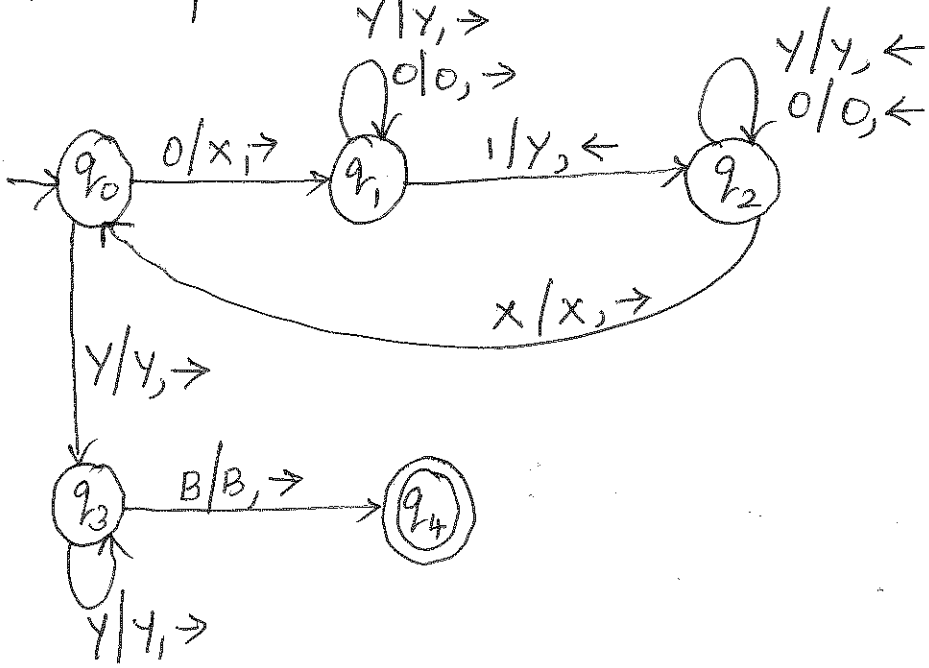
Prob 3: Design a Turing Machine for the Language $L = \{0^n 1^n \mid n \geq 1\}$

Sol

Initial configuration of the tape is

0	0	0	1	1	1	B	...

Every zero in the tape is matched with every 1 in the tape.



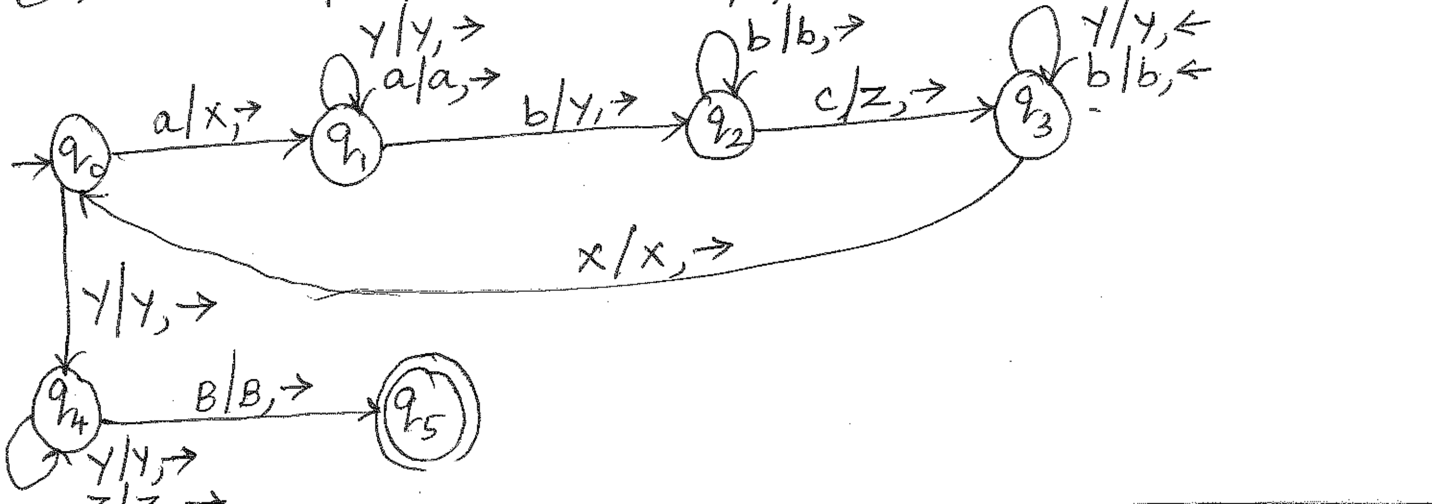
Prob 4: Design a turing machine for the Language $L = \{a^n b^n c^n \mid n \geq 1\}$

Sol

Initial configuration of the tape is

a	a	b	b	c	c	B	...

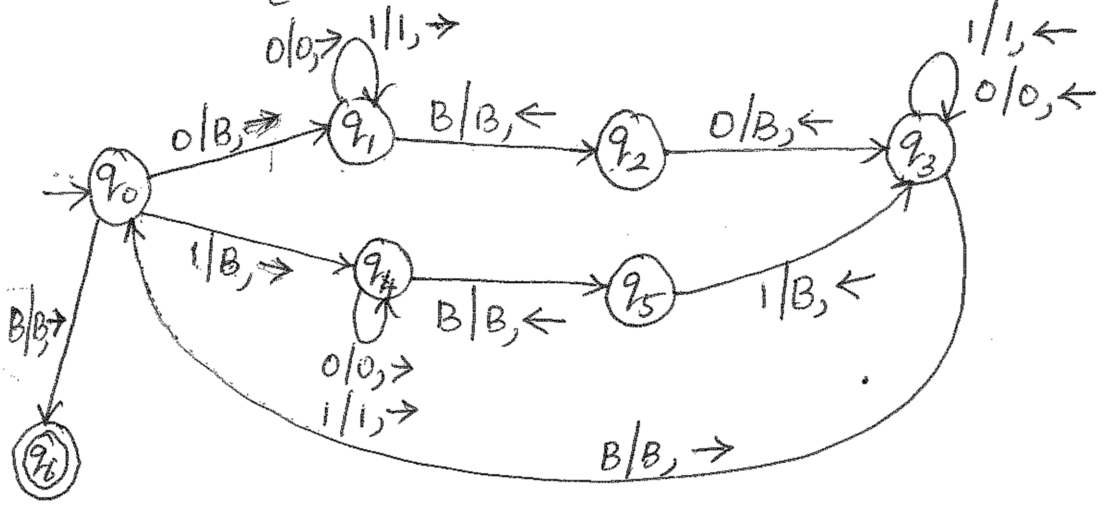
Every a in the tape is matched with every b and every c in the tape.



Prob 5: Design a turing machine $L = \{ww^R \mid w \in (0,1)^*\}$

Sol

$L = \{0110, 1001, \ldots\}$



ID for the string 0110B,

$q_0 0110B \vdash Bq_1 110B \vdash B1q_1 10B \vdash B11q_1 0B \vdash B110q_1 B$

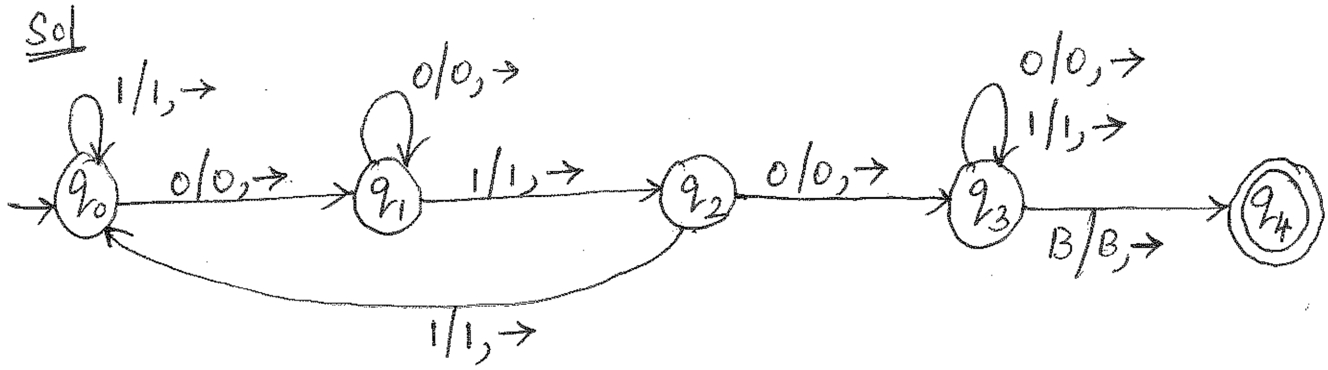
$\vdash B11q_2 0B \vdash B1q_3 1BB \vdash Bq_3 11BB \vdash q_3 B11BB$

$\vdash Bq_0 11BB \vdash BBq_4 1BB \vdash BB1q_4 BB \vdash BBq_5 1BB$

$\vdash Bq_3 BBBB \vdash BBq_0 BBB \vdash BBBq_6 BB \in F.$

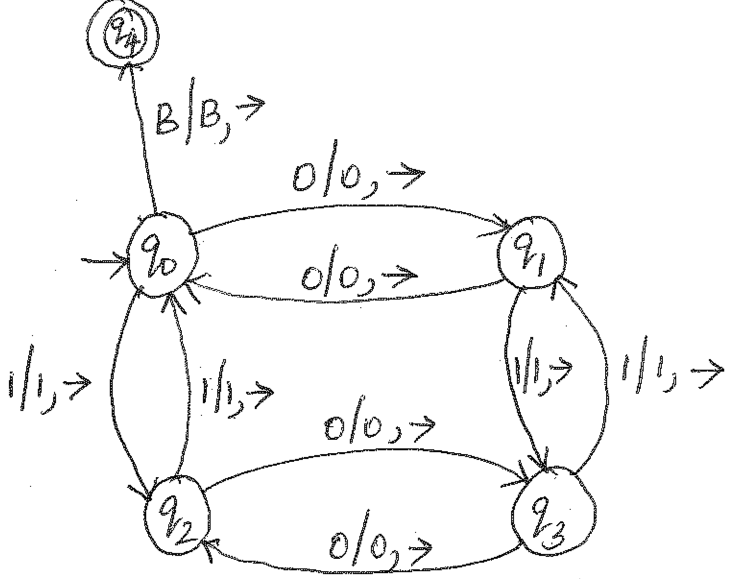
Prob 6: Design a Turing Machine to accept the set of all strings $\{0,1\}$ with 010 as substring.

Sol



Prob 7: Design a Turing machine to accept the set of all strings over alphabet {0,1} with even number of 0's and 1's.

Sol

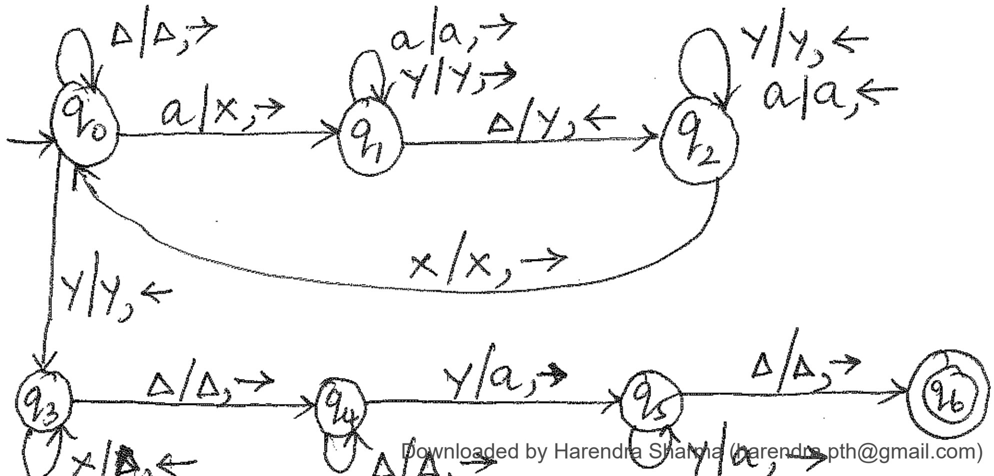


Prob 8: construct a turing machine to move an I/P string over the alphabet A={a} to the right one cell. Assume that the tape head starts somewhere on a blank cell to the left of the input string. All other cells are blank, labeled by Δ. The machine must move the entire string to the right one cell, leaving all remaining cells blank.

Sol

I/P	Δ	Δ	a	a	a	Δ	Δ	Δ ...
O/P	Δ	Δ	Δ	Δ	Δ	a	a	a ...

we can change every a to x and the followed Δ to y, finally change x to Δ, y to a.



Downloaded by Harendra Sharma (harendra.pth@gmail.com)

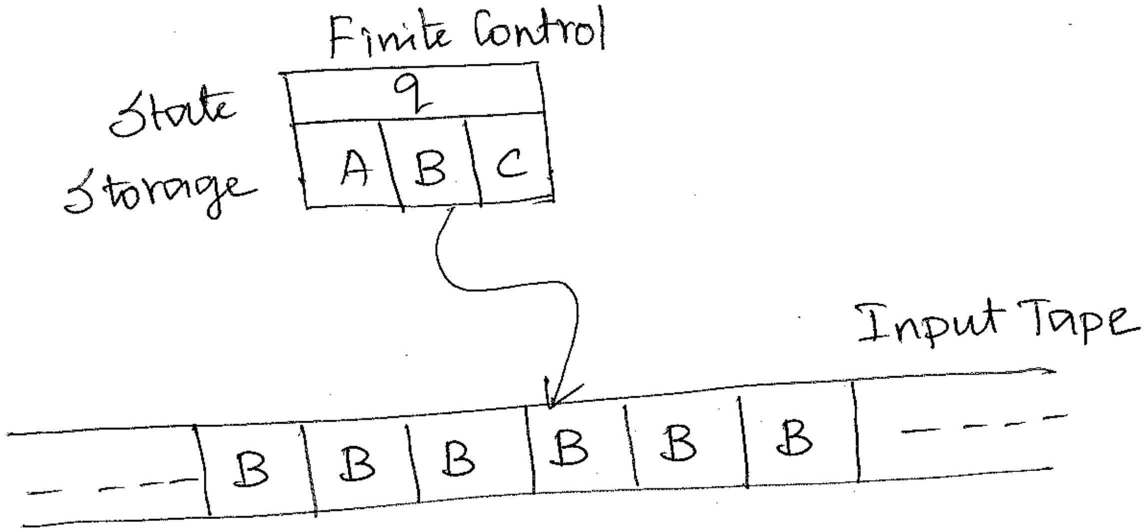
Techniques For Turing Machine Construction

The different techniques that are used to design a Turing machine are as follows,

1) Storage in finite Control
2) Multiple Tracks
3) Subroutines
4) Checking off Symbols.

1) Storage in finite control:

Generally the Turing machine finite control contains State and transition information.
But here in the storage of finite control, we store the data along with the state.
So here we use the finite control to hold finite amount of data and it is shown below,



This type of Turing machine makes the state to remember and to have a memory for the symbol scanned in the input.

- This type of Turing machine can be designed to Store in the state with any data from the input.
- Each state contains the 'B' blank symbol as its storage initially.
- This type of Turing machine is used to store any symbol in the input and to check whether the stored symbol appears in the input.

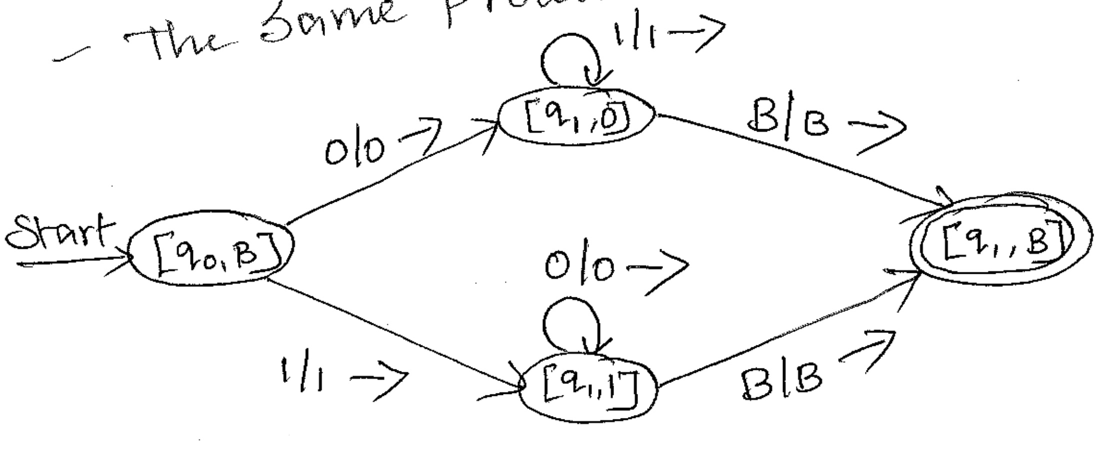
Example:

Design a Turing machine to accept the string

$01^* + 10^*$

Idea of the design is

- Initially the state q_0 is having the storage of B.
- First symbol of the tape is become 0 or 1.
- First symbol of the tape is taken into the storage of q_1
- If it is 0 that is taken into the same state State and the TM will be in the same state if it finds the complement of 0.
- And Reach the final state if it finds B in the tape, and the same is taken in to the q_1 state.
- The same process is consider for 1 also.



Turing Machine definition is,

$M = (\{[q_0,B],[q_1,0],[q_1,1],[q_1,B]\}, \{0,1\}, \{0,1,B\}, \delta, [q_0,B], B, \{[q_1,B]\})$

$[q_0,B] \rightarrow$ Initial State

$[q_1,B] \rightarrow$ Final State

where δ is given as follows,

$\delta([q_0,B],0) = ([q_1,0],0,R)$

$\delta([q_0,B],1) = ([q_1,1],1,R)$

$\delta([q_1,0],1) = ([q_1,0],1,R)$

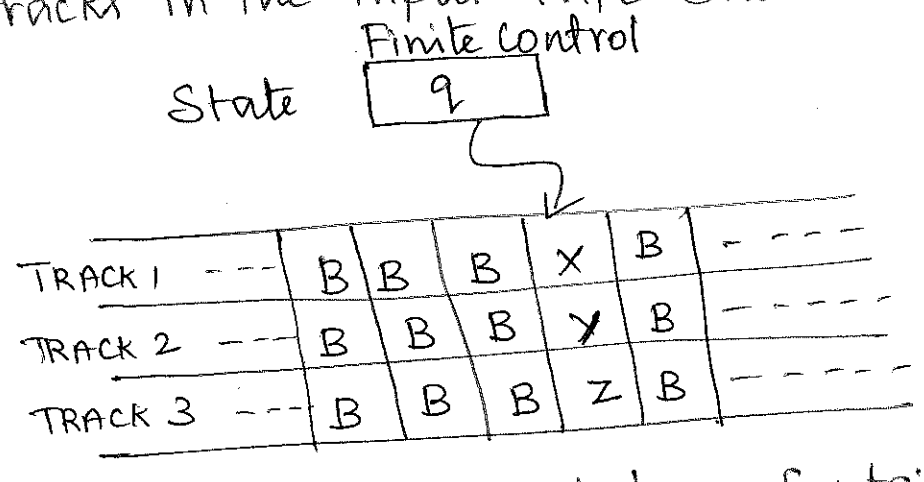
$\delta([q_1,0],B) = ([q_1,B],B,R)$

$\delta([q_1,1],0) = ([q_1,1],0,R)$

$\delta([q_1,1],B) = ([q_1,B],B,R)$

2. Multiple Tracks Turing Machine ~~[illegible]~~ :

we can extend the general TM to include Multiple tracks in the input tape shown below,



- Each track in the input tape contains one symbol.
- Tape Alphabet of the TM consists of tuples with one component in each track and the number of component in the tuples depends on the number of tracks in the input tape.

Downloaded by Harendra Sharma (harendra.pth@gmail.com)

- For example in the above example the cell scanned by the tape head contains the symbol [x, y, z]

Example:

Design a Turing Machine using multiple tracks to check whether the given input number is prime or not.

- The idea of designing the TM is that let us store the input symbol in the first track of input tape.
- Let us store the number 2 is binary in the second track of input tape.
- Let us copy the input in the third track also,
- All the symbols in the three tracks of the TM are in binary form.
- Now subtract the second track from the third track until we get '0' or any remainder.
  - If the remainder is zero, then the number is not prime, since the prime number is one which is divided by 1 and itself.
  - If the remainder is non zero value, then the second track value is incremented by 1 and again the divides procedure is continued.
  - If the value of the second & first track is equal, then the number is prime number

Now process is as follows, for input value 7, Subtracting 2 from 7, we get

7		7		7		7
2	→	2	→	2	→	2
7		5		3		1

The Remainder is 1, so increment the value of second track by 1

7		7		7
3	→	3	→	3
7		4		1

The Remainder is 1, so increment the value of second track by 1

7		7
4	→	4
7		3

The Remainder is 3, so increment the value of second track by 1

7		7
5	→	5
7		2

The Remainder is 2, so increment the value of second track by 1

7		7
6	→	6
7		1

The Remainder is 1, so increment the value of second track by 1

7
7
7

Now the value of first and second track is equal, so the number 7 is a Prime number

Downloaded by Harendra Sharma (harendra.pth@gmail.com)

4) Checking Off Symbols:

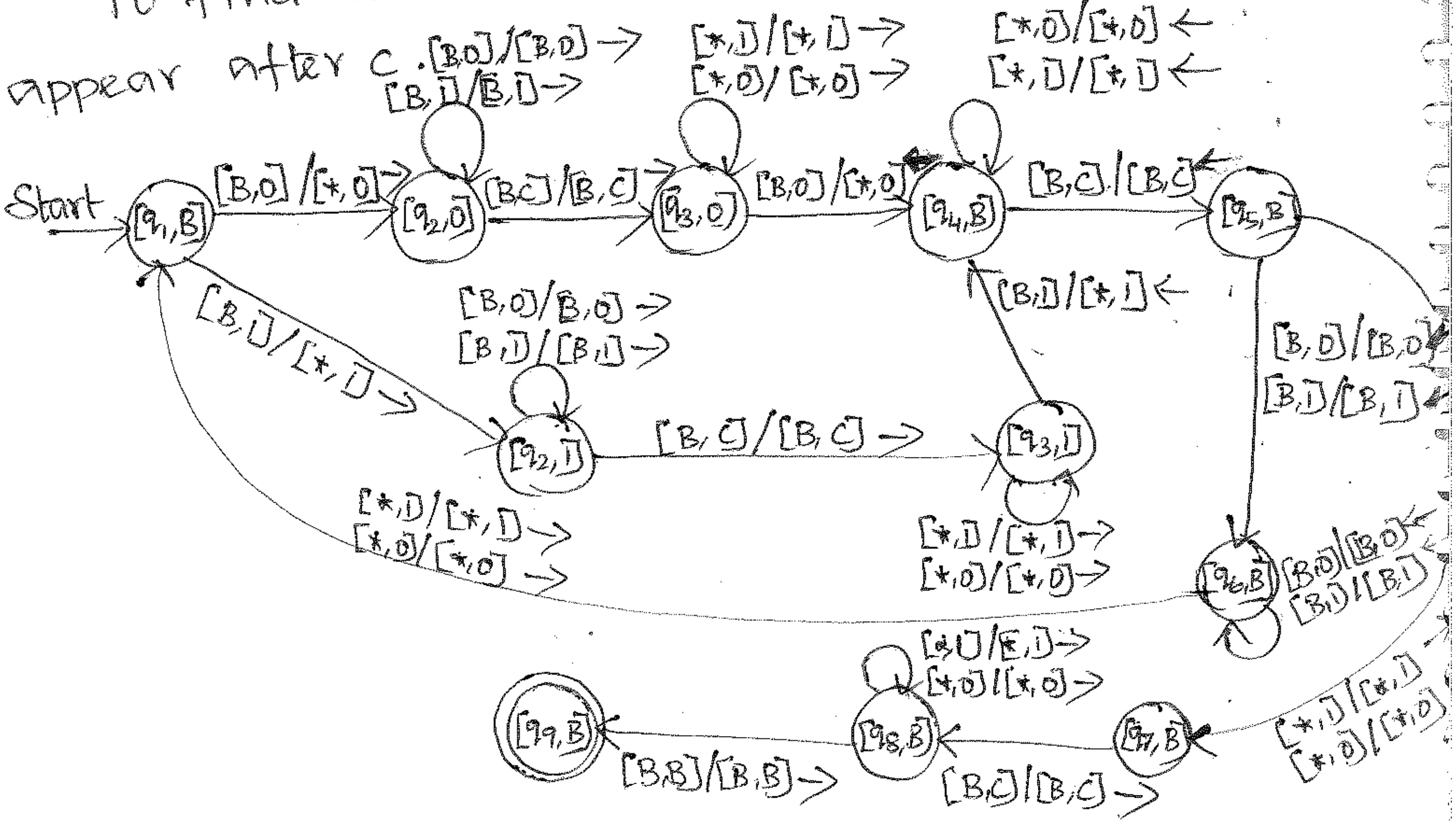
The Turing Machine can be extended by using checking off symbols.
This method is used by the Turing Machine for the language that contains the repeated strings and to compare the length of the two substrings.
The requirements for the TM is having storage in finite control as well as multiple tracks.

Example:

Design a TM to recognize the language
$L = \{ wcw \mid w = \{0,1\}^* \}$

Soln:

The idea here is the input string may be started with 0 or 1 so there are two edges leaving from the start state labeled as 0 and 1.
To find wcw the second substring w should be appear after c.



Subroutines:

There are some problems, in which some tasks need to be performed repeatedly and it can be done by subroutines. Subroutine in the TM is the set of states that specifically performs same tasks.

- The set of states in the subroutine has one start state and another state namely the return state.
- The return state of the subroutine does not have moves and it pass the control to the other set of states of the TM that calls the subroutine.
- The subroutine is called whenever there is a transition to its initial state.

Example:

Design a Turing Machine to perform the multiplication function $f(m,n) = m * n$

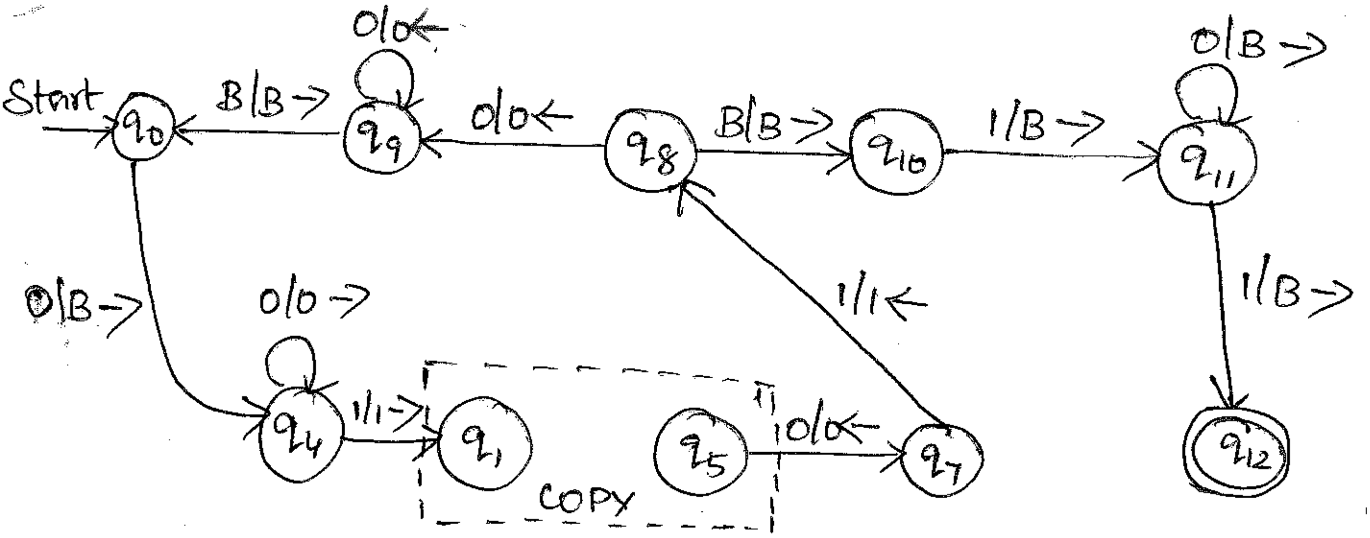
The initial configuration of the tape is $0^m 1 0^n 1$

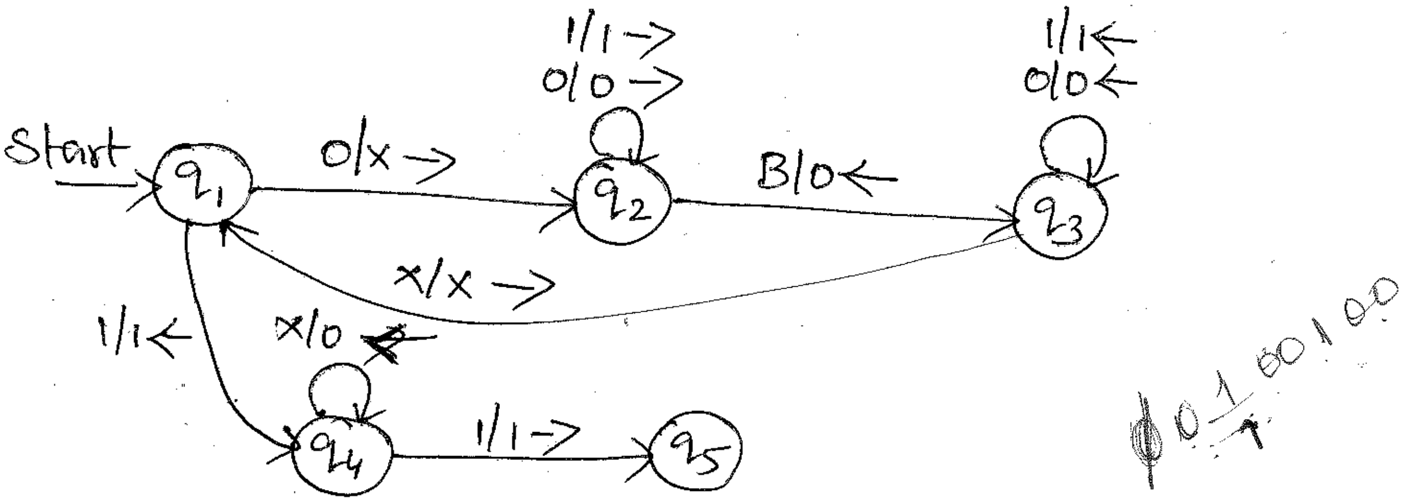
$m = 2 \quad n = 3$

0	0	1	0	0	0	1	B	----

The idea is every 0 in m the no. of 0's in n is copied. This process is repeated for all the 0's in the m.

The copy process is repeatedly called. So we can take that into subroutine.





Non Determininitic Turing Machine:

A Non Determininitic TM is one in which the transition function δ is given as,

$$\delta(q, X) = \{(q_1, Y_1, D_1)\ (q_2, Y_2, D_2)\ \cdots (q_n, Y_n, D_n)\}$$

From the above transition, we can know that the transition for any state 'q' with the tape symbol 'X', we can have many transition.

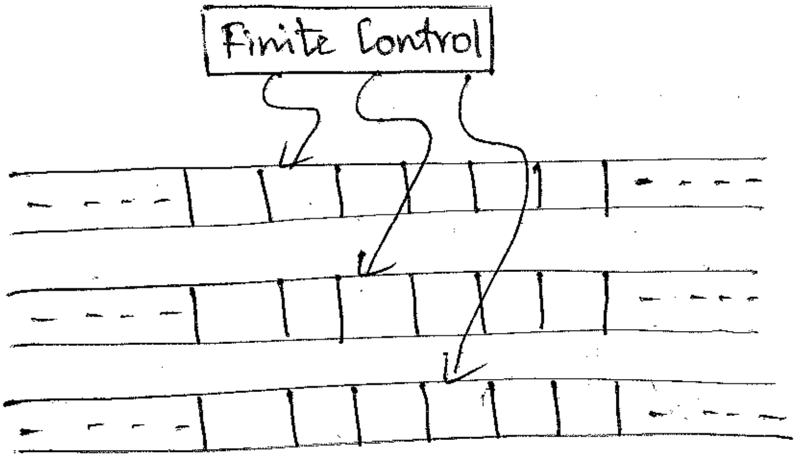
The Non Determininitic TM can choose any one of the transition move at the state q with the tape symbol X.

Multi tape Turing Machines:

(13A)

The Multiple TM has a finite control state and some finite no. of tapes.

Each tape in the multiple (or) multitape TM is divided into cells and each cell can hold any symbol and the multiple TM is shown below,



The multi tape has the following,

1. The input which is the finite sequence of input symbols and is placed on the first tape.
2. All the other cells of all the tapes hold the blank symbols.

A move of the multitape TM depends on the following,

1. State of the finite control
2. Symbol scanned by each tape head

In a single move, the Multiple TM does the following,

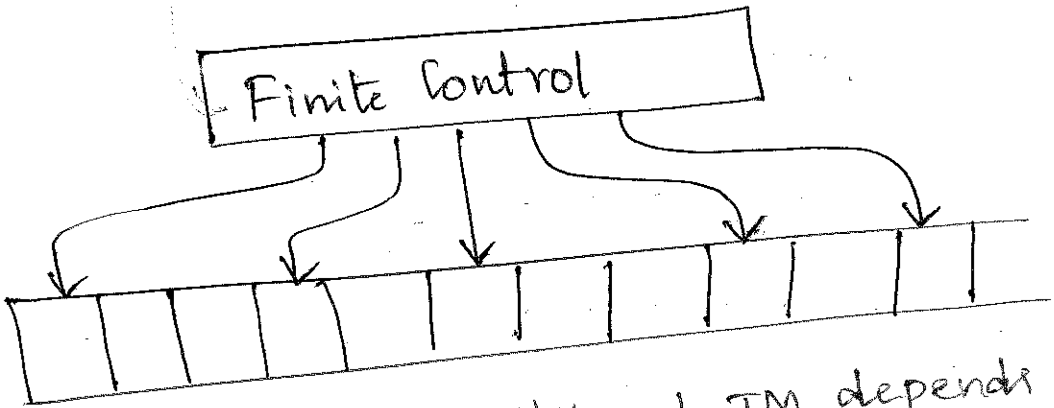
1. The finite control enters a new state
2. ~~One~~ on each tape a new tape symbol is written on the cell scanned.

Downloaded by Harendra Sharma (harendra.pth@gmail.com)

3. Each of the tape head makes a move, which can be

d Turing Machine:

ring Machine is extended to Multi heads achine in which the TM has one control and one input tape with n no.of read write heads.



A move of the multihead TM depends on the following,

1. State of the finite control
2. Symbol Scanned by each tape head

In a single move, the multihead TM does the following,

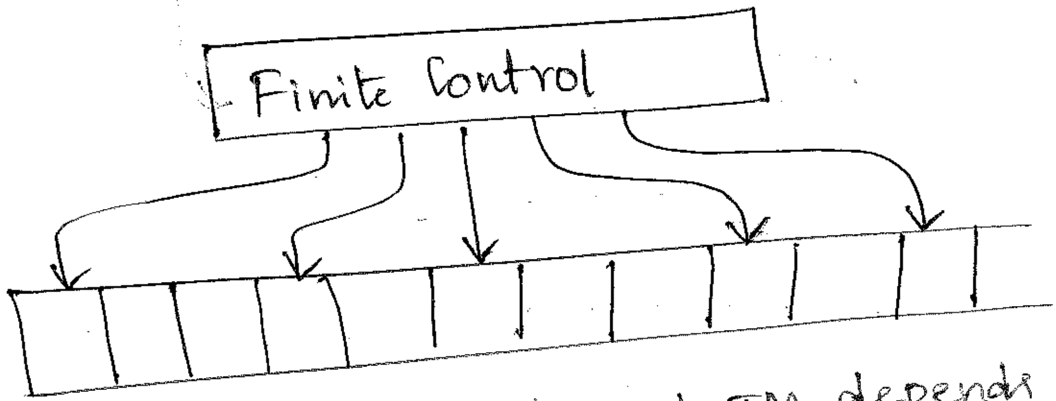
1. The finite control enters a new state
2. On ~~each~~ tape, a new tape symbol is written on the cell scanned
3. Each of the tape head makes a move, which can be either left, right

Two way Infinite tape:

We can make TM is more powerful by allowing two way infinite tape as shown below

Multi head Turing Machine:

The Turing Machine is extended to Multi heads turing machine in which the TM has one finite control and one input tape with n no. of read write heads.



A move of the multihead TM depends on the following,

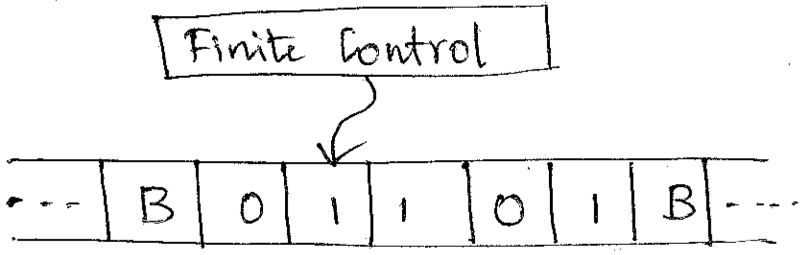
1. State of the finite control
2. Symbol Scanned by each tape head

In a single move, the multihead TM does the following,

1. The finite control enters a new state
2. On each tape, a new tape symbol is written on the cell scanned
3. Each of the tape head makes a move, which can be either left, right

Two way Infinite tape:

we can make TM is more powerful by allowing two way infinite tape as shown below



This TM is different from General TM is, the Blank symbol is placed in both ends of the tape.

Theorem:

L is recognized by a TM with a two way infinite tape if and only if it is recognized by a TM with one way infinite tape.

Proof:

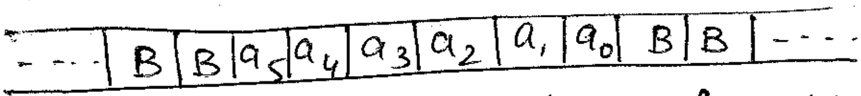
Let M_1 be a TM with one way infinite tape and can be denoted by

$$M_1 = (Q_1, \Sigma_1, \Gamma_1, \delta_1, q_1, B, F_1)$$

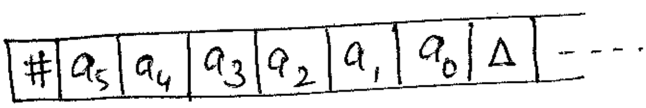
Similarly, M_2 be a TM with two way infinite tape and can be denoted by

$$M_2 = (Q_2, \Sigma_2, \Gamma_2, \delta_2, S_0, B, F_2)$$

The input tapes are as shown by following figure.



Two way infinite tape for M_2 TM



one way infinite tape for M_1 TM

As shown in the above figure, the TM with one way

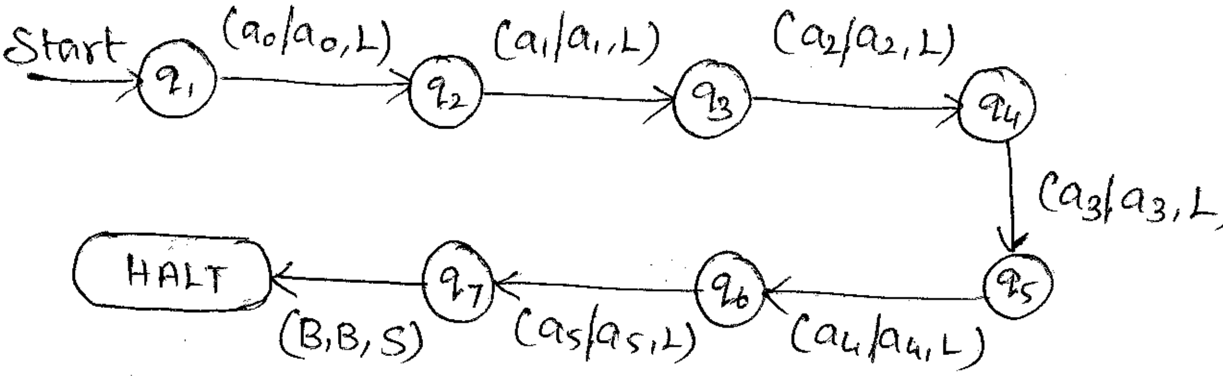
Downloaded by Harendra Sharma (harendra.pth@gmail.com)

... external symbol # placed in the

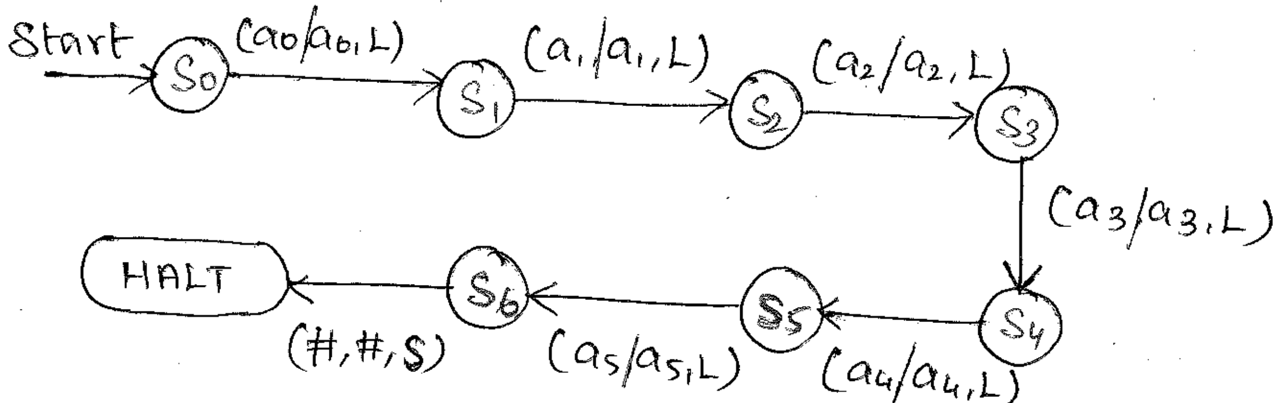
This symbol is used as indicator for the left side termination.

If we want a Language $L = \{a_0, a_1, a_2, a_3, a_4, a_5\}$ sequence then in the two way infinite tape the tape head is fixed at the rightmost symbol.

The TM M_2 can be



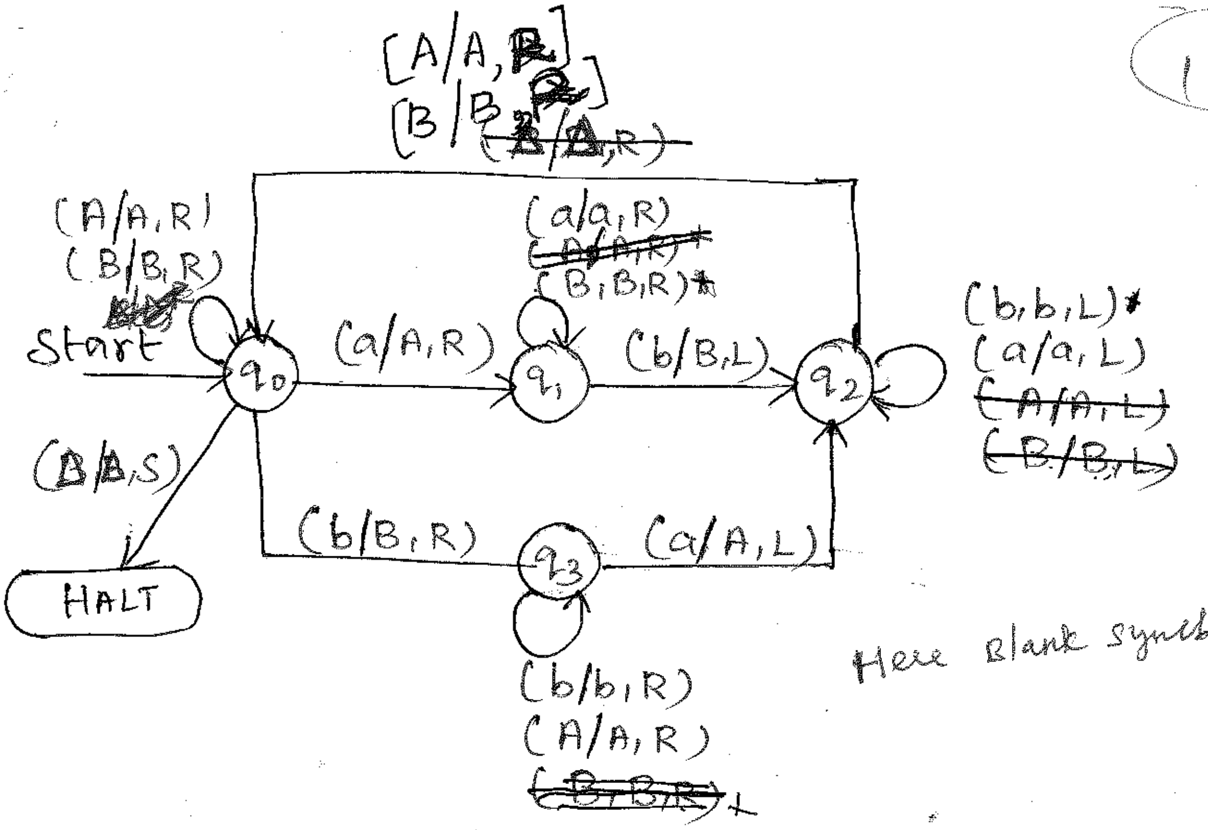
Similarly, with one way infinite tape the machine M_1 will be



From these two string validation we can proved that the Language L is recognized by a TM with a two way infinite tape is also recognized by a TM with one way infinite tape

Example:

Construct a TM for a Language having equal number of a's and b's in it over the input set $\Sigma = \{a, b\}$



Here Blank symbol is 'Δ'.

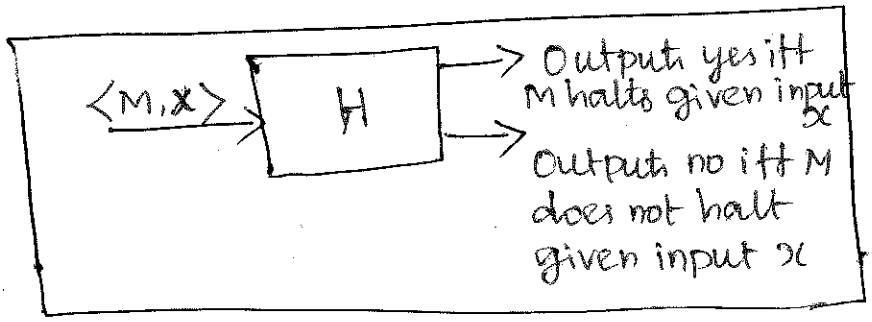
The idea is the input string may be started with a or b. So there are two edges leaving from starting state. Suppose first appear symbol is a change it into A and move towards right to find corresponding b and change it into B and vice versa.

After every finds the head should move left end for repeating the matching process. for that the head needs to know the left end. So it can be easily achieved in two way infinite tape because blank symbol is present in left end also.

Halting Problem:

Turing Machine acceptance is otherwise reffered as Halting.

If the TM Halts and there is no next move whenever the input is accepted.
If the input is not accepted then the TM may or maynot halt.
The TM to halt regardless of whether or not they accept the string, those language are reffered as recursive language.
The Halting problem would be solvable if a TM H that behaves like below can be constructed:



Theorem:

The Halting problem for TM is Unsolvable.

Proof:

The proof is by contradiction. Assume, the halting problem is solvable.
If the halting problem is solvable, then there must be a TM to decide the halting problem, that is the TM H exists.

Then, if H can solve the Halting problem, ~~that is the TM H~~ for input <M, x> it should be able to solve the halting problem for input <M,M>.
Then it should be possible to construct another TM H' that accept as input M but behaves like H:

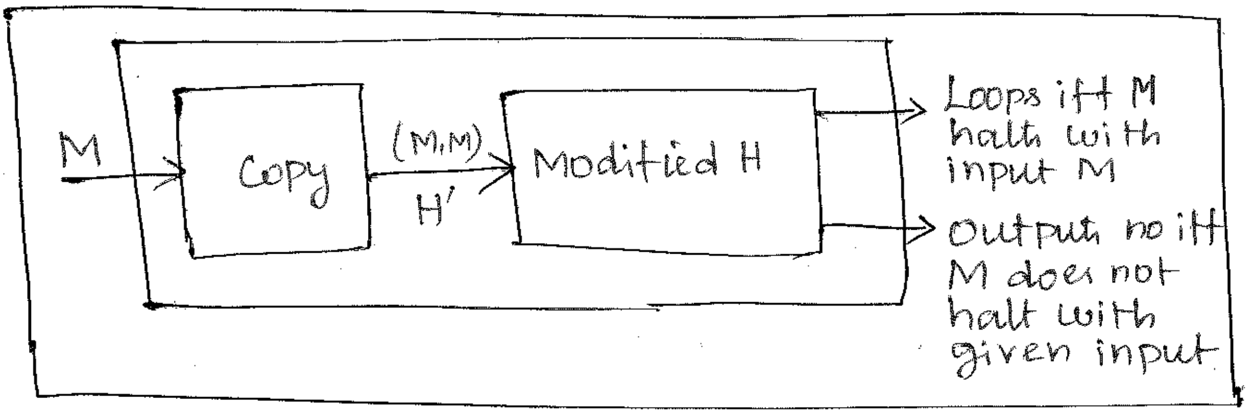
Downloaded by Harendra Sharma (harendra.pth@gmail.com)

H takes the input: M and makes a copy to obtain the input <M, M> (141)

Then H' executes H with the input <M, M> but in the case that H outputs yes, H' loops forever.

This is done by adding a couple of states for a non-terminating right-left loop.

Then the TM H' is illustrated as follows:



Then the TM H' is illustrated as follows:

H' is also a TM, then assume that H' receives <H'> as an input. What is the behaviour of H'?

- If H' halts for <H'>, then H answers yes and H' loops.
- If H' does not halt for e(H'), the H answers no and H' halts.

Both the above represent a contradiction, then H or H' exist, so the Halting problem is effectively unsolvable.

Partial Solvability:

The purpose of an computer application program is to give the output string for every correct input string and this execution can be formulated as a partial function 'F' which computes from one

group of string to another group of strings.
A function is said to be partial if it may be undefined for some arguments.
The TM computing the partial function is partial solvability.
A TM T with input alphabet Σ computes a function whose domain D is a subset of Σ^*.
For every input string ω in the domain of the function T carries out a computation that ends with the output string $f(x)$ on the tape.
TM T computes a partial function with domain D then T accepts D regardless of the output it produces.
A partial function $F: (\Sigma^*)^x \rightarrow \Gamma^*$ is computable by the TM, if there is a TM T that computes F.

Example:

Design a TM that Reverse a String.

$\{a, b\}^* \rightarrow \{b, a\}^*$

Soln:

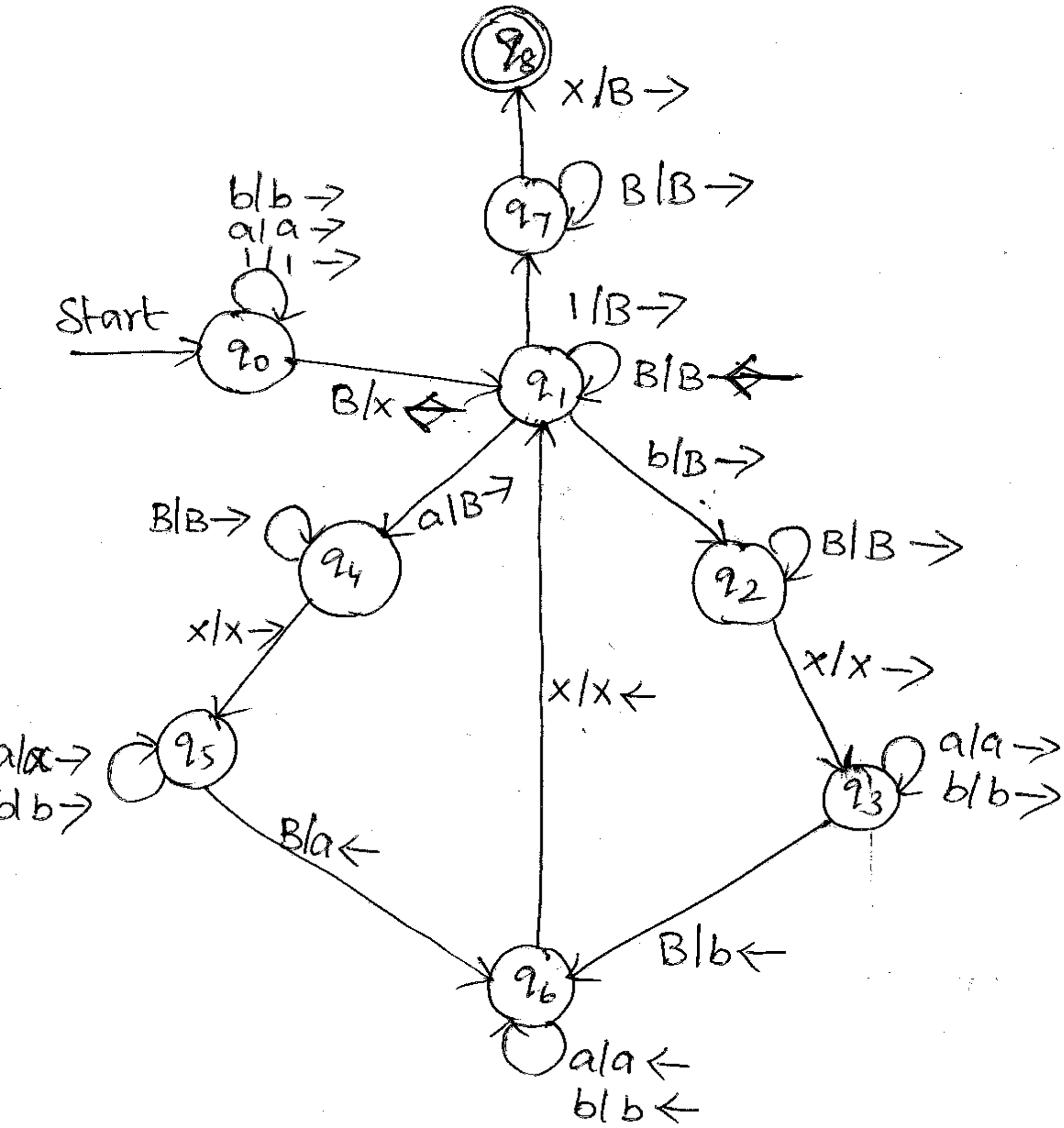
During the process of Reversing the string the head has to looking back the left end of the input tape. So we have to insert a special symbol to mark the left end. So the configuration of the tape is

1	a	b	a	b	B	B	----

Downloaded by Harendra Sharma (harendra.pth@gmail.com)

Reversing the string is started after of the given string. Reversing string is differentiated from the given string in the by making the first Blank after the given in is converted to x.

142



Example 2:

Design a TM to compute the function "n mod 2" where n is an integer and n>2.

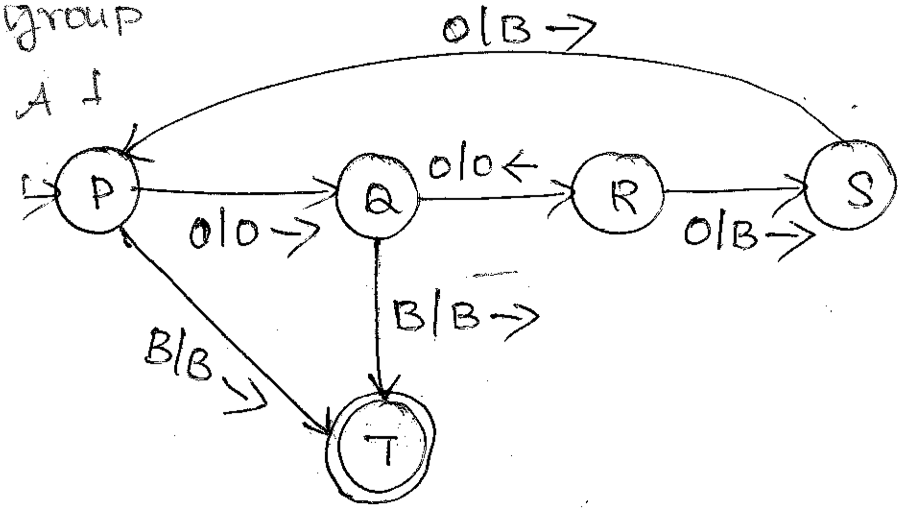
Soln:

The idea is check for 2 consecutive 0 if it is so cancel it, repeat the process for given n.

Downloaded by Harendra Sharma (harendra.pth@gmail.com)

group

A 1



Chomsky Hierarchy of Languages:

Refer Unit-III Types of Grammar.