Turing Machine

A Turing machine M is defined by:

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

where

Q :- is set of states

Σ :- input alphabet
$(\Sigma \subseteq \Gamma - \{\square\})$

Γ :- tape alphabet

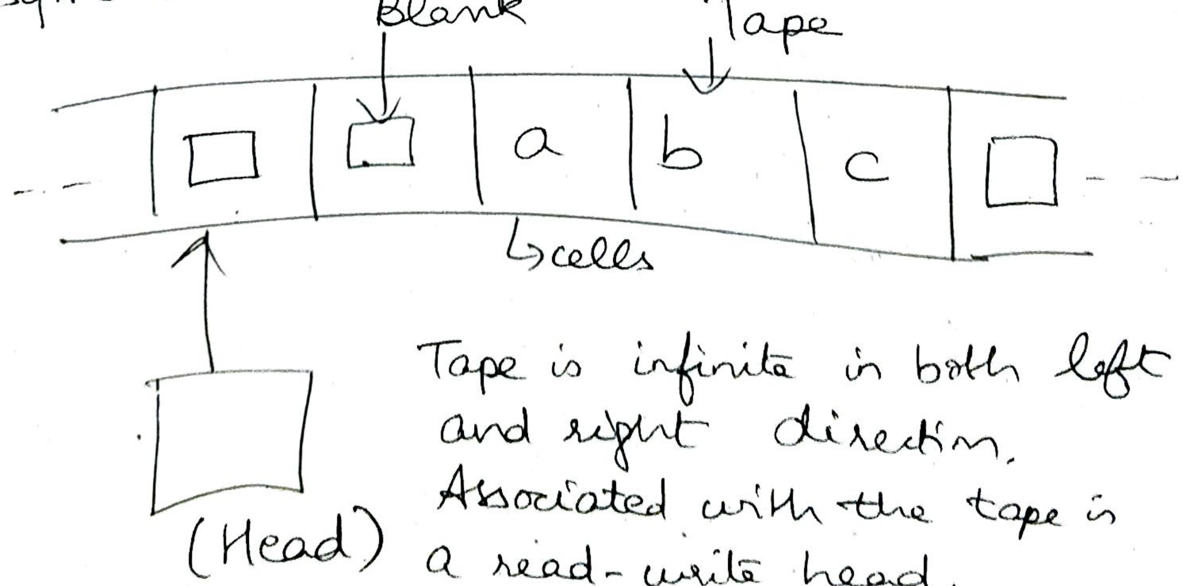
δ : Transition function
$Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$

$\square \in \Gamma$: Blank

$q_0 \in Q$: Initial state

$F \subseteq Q$: set of final states

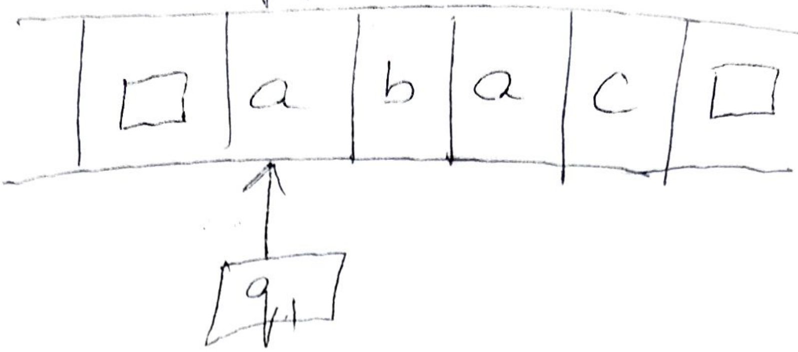
In Turing machine, temporary storage is tape. The tape is divided into cells each of which is capable of holding one symbol.



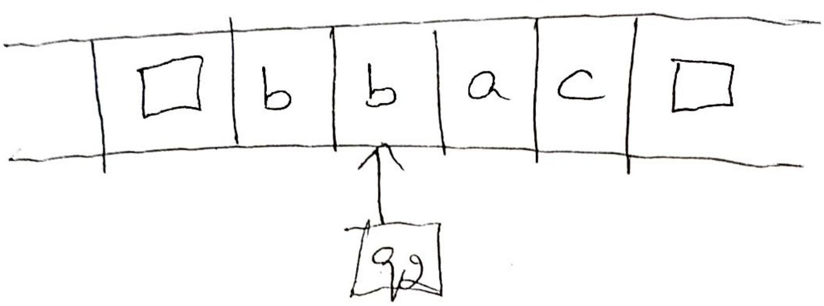
Tape is infinite in both left and right direction.
Associated with the tape is a read-write head.

Initial configuration:

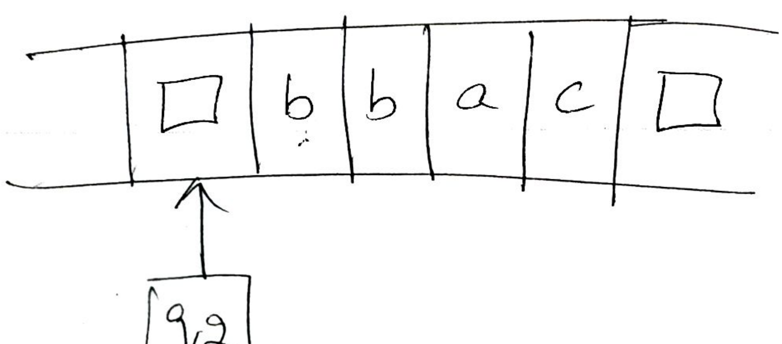
q



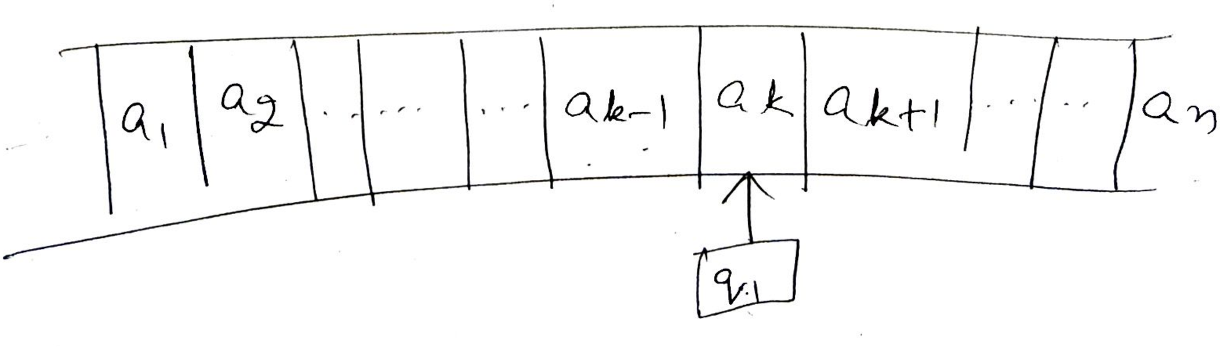
① $\delta(q_1, a) = (q_2, b, R)$



② $\delta(q_1, a) = (q_2, b, L)$



Instantaneous description of a TM:-



D we will denote it by:

$$a_1 a_2 \cdots a_{k-1} q_1 a_k a_{k+1} \cdots a_n$$

Language generated / accepted by TM:-

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ be a TM.
Then the language generated by M is:-

$$L(M) = \{ w \in \Sigma^+ : q_0 w \vdash^* x_1 q_f x_2 \text{ for some } q_f \in F,\ x_1 x_2 \in \Gamma^* \}$$

Q: Consider T.M. defined by:

$Q = \{q_0, q_1\}$

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

$\Gamma = \{a, b, \square\}$

$F = \{q_1\}$

q_0 = initial state

$\square$ = Blank

and

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

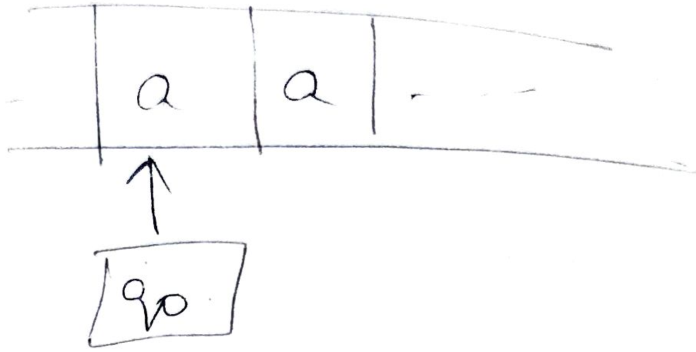
$\delta(q_0, b) = (q_0, b, R)$

$\delta(q_0, \square) = (q_1, \square, L)$

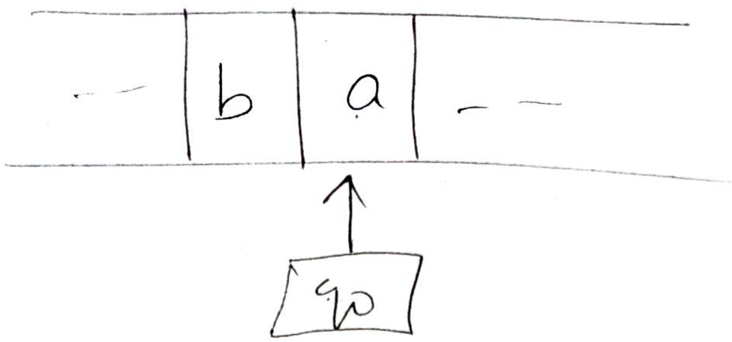
Find the next move if input is 'aa'.

Ans

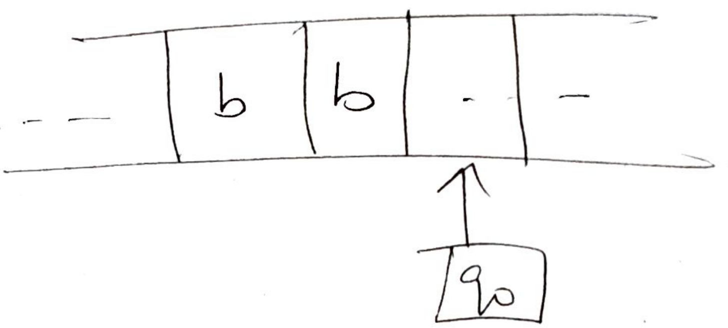
(1)



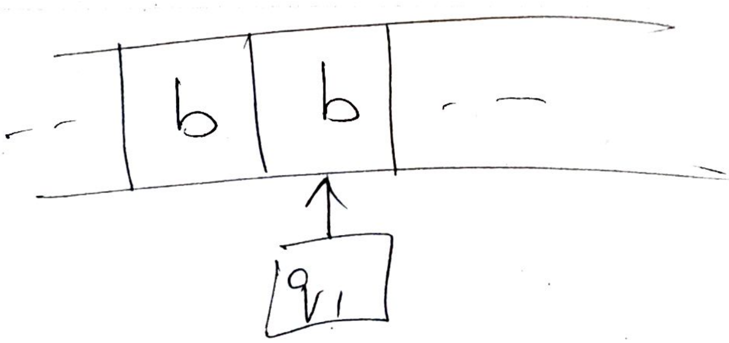
(2)



(3)



(4)



ID is:-

$q_0 aa \vdash b q_0 a \vdash bb q_0 \vdash b q_1 b$

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

$L(M) = L(00^*)$

$L(M) = \{0, 00, 000 \ldots\}$

$L(M) = \{0, 00, 000 \ldots\ldots\}$

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

$Q = \{q_0, q_f\}$

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

$\Gamma = \{0, 1, \square\}$

$\delta(q_0, 0) = (q_0, 0, R)$

$\delta(q_0, \square) = (q_f, \square, R)$

We do not define any move for $(q_0, 1)$ as we dont want '1' to be accepted in our language.

Q: Design a TM that accepts the language $L = \{a^n b^n : n \geq 1\}$

$L = \{ab, aabb, aaabbb \ldots\}$

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

$Q = \{q_0, q_1, q_2, q_3, q_f\}$

$\Sigma = \{a, b\}$ $\cancel{x, y, \square}$

$\Gamma = \{a, b, x, y, \square\}$

$F = \{q_f\}$

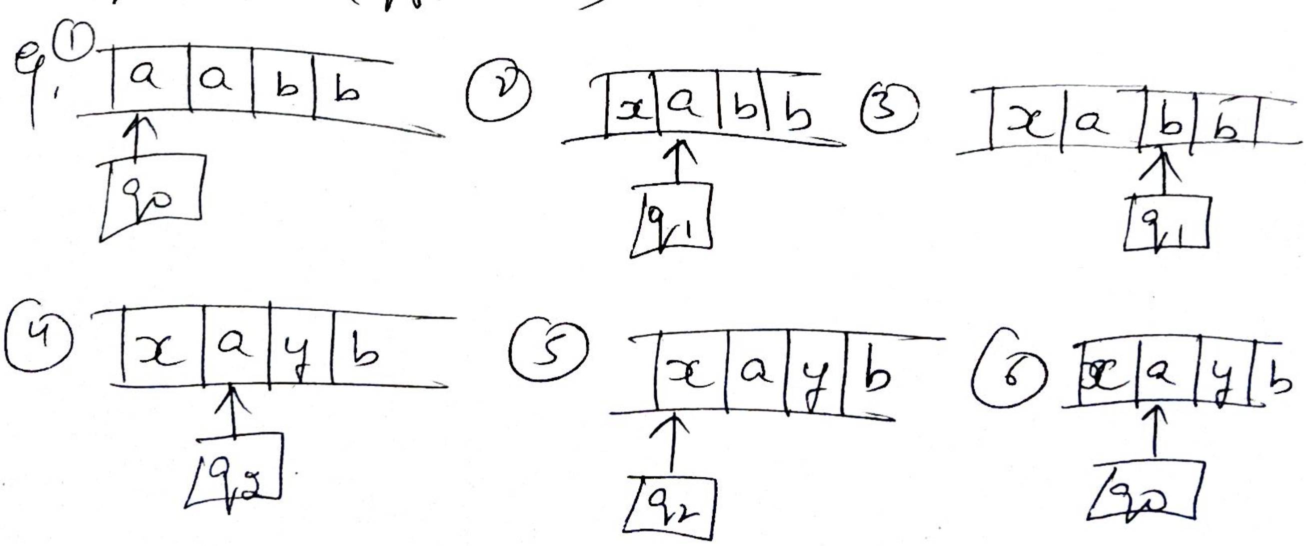
① aabb   ② xabb   ③ xayb

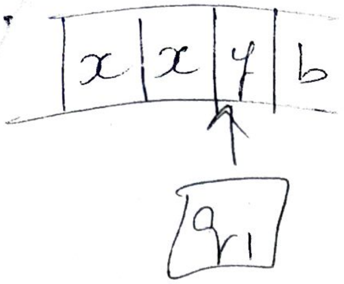
x   y   x

④ xxyb   ⑤ xxyy

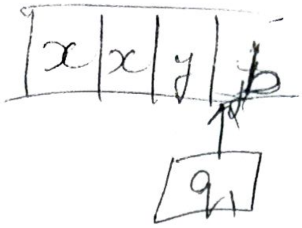
y

$$\delta(q_0, a) = (q_1, x, R)$$
$$\delta(q_1, a) = (q_1, a, R)$$
$$\delta(q_1, b) = (q_2, y, L)$$
$$\delta(q_2, a) = (q_2, a, L)$$
$$\delta(q_2, x) = (q_0, x, R)$$
$$\delta(q_1, y) = (q_1, y, R)$$
$$\delta(q_2, y) = (q_2, y, L)$$
$$\delta(q_0, y) = (q_3, y, R)$$
$$\delta(q_3, y) = (q_3, y, R)$$
$$\delta(q_3, \square) = (q_f, \square, R)$$

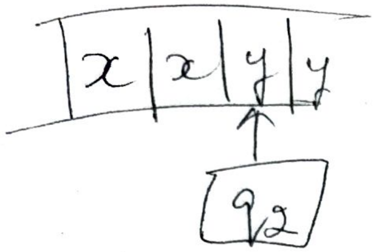




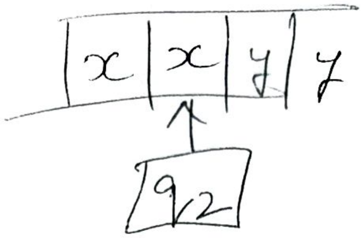
8



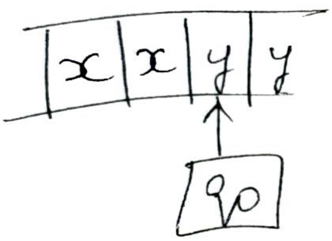
9



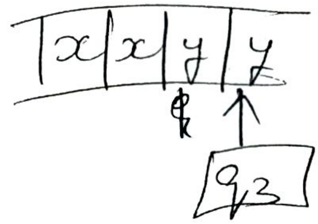
10



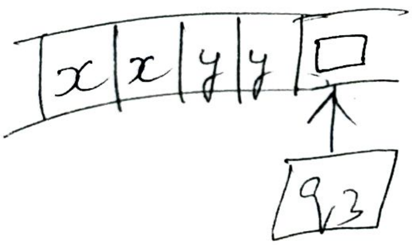
11



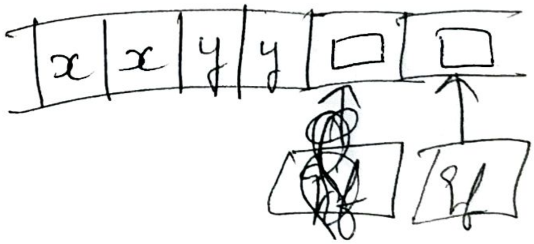
12



13



14



Q: Design a T.M. that accepts

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

$L = \{abc, aabbcc, aaabbbccc \text{---}\}$

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

Here we replace 'a' by 'x', 'b' by 'y' and c by 'z'.

$\delta(q_0, a) = (q_1, x, R)$

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

$\delta(q_1, y) = (q_1, y, R)$

$\delta(q_1, b) = (q_2, y, R)$

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

$\delta(q_2, Z) = (q_2, Z, R)$

$\delta(q_2, c) = (q_3, Z, L)$

$\delta(q_3, Z) = (q_3, Z, L)$

$\delta(q_3, b) = (q_3, b, L)$

$\delta(q_3, y) = (q_3, y, L)$

$\delta(q_3, a) = (q_3, a, L)$

$\delta(q_3, x) = (q_0, x, R)$

$\delta(q_0, y) = (q_4, y, R)$

$\delta(q_4, y) = (q_4, y, R)$

$\delta(q_4, Z) = (q_4, Z, R)$

$\delta(q_4, \square) = (q_f, \square, R)$

Design TM for

(5)

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

$L = \{abb, aabbbb - - \}$

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

δ is:

$\delta(q_0, a) = (q_1, x, R)$

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

$\delta(q_1, b) = (q_2, y, R)$

$\delta(q_2, b) = (q_3, y, L)$

$\delta(q_3, y) = (q_3, y, L)$

Here we replace one 'a' by 'x' and 2 'bs' by 'y'

$\delta(q_3, a) = (q_3, a, L)$

$\delta(q_3, x) = (q_0, x, R)$

$\delta(q_1, y) = (q_1, y, R)$

$\delta(q_4, y) = (q_4, y, R)$ (A)

$\delta(q_0, y) = (q_4, y, R)$

$\delta(q_4, \square) = (q_f, \square, R)$

$q_0 aabbbb \vdash x q_1 abbbb \vdash x a q_1 bbbb \vdash$
$x a y q_2 bbb \vdash x a q_3 yy bb \vdash x q_3 a yy bb$
$\vdash q_3 x a yy bb \vdash x q_0 a yy bb \vdash x x q_1 yy bb$
$\vdash x x y q_1 y bb \vdash x x yy q_1 bb \vdash x x yyy q_2 b$
$\vdash x x yy q_3 yy \vdash x x y q_3 yyy \vdash x x q_3 yyyy$

$\vdash x q_3 x y y y y \vdash x x q_0 y y y y$

$\vdash x x y q_4 y y y \vdash x x y y q_4 y y \vdash x x y$

$\vdash x x y y y y q_4 \square \vdash x x y y y y q_f$

Q: Find the T.M that accepts ~~the~~ language for even no' of '1's

$\Sigma = \{1\}$

$\Gamma = \{b, \square\}$

$M = (Q, \Sigma, \Gamma, \delta, q_1, \square, F)$

$\delta(q_1, 1) = (q_2, b, R)$
$\delta(q_2, 1) = (q_1, b, R)$
$\delta(q_1, \square) = (q_f, \square, R)$

Here we replace '1' by 'b' and after every second go to initial state q_1

eq. $w = 1111$

$q_1 1111 \vdash b q_2 111 \vdash b b q_1 11 \vdash b b b q_2 1$

$\vdash b b b b q_1 \square \vdash b b b b q_f$

eq. $w = 111$

$q_1 111 \vdash b q_2 11 \vdash b b q_1 1 \vdash b b b q_2 \square$

So machine halts, not accepted

Design language that accepts:

$L = \{01^* + 10^*\}$
or

$\delta(q_0, 0) = (q_1, b, R)$
$\delta(q_1, 1) = (q_1, 1, R)$
$\delta(q_1, \square) = (q_f, \square, R)$

$\Big\}\ 01^*$

or

$\delta(q_0, 1) = (q_2, b, R)$
$\delta(q_2, 0) = (q_2, 0, R)$
$\delta(q_2, \square) = (q_f, \square, R)$

$\Big\}\ 10^*$

eg. $w = 011$

$q_0 011 \vdash b q_1 11 \vdash b1 q_1 1 \vdash b11 q_1 \square$
$\vdash b11 q_f$

eg. $w = 100$

$q_0 100 \vdash b q_2 00 \vdash b0 q_2 0 \vdash b00 q_2 \square$
$\vdash b00 q_f$

Variations of Turing Machine

Multitape turing machine

A multitape turing machine is a turing machine with several tapes each of which has an independent read-write head.

So, we define a n-tape machine by

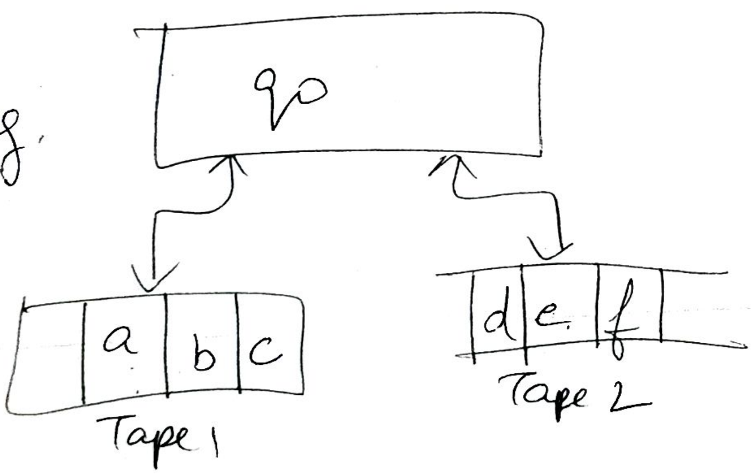
$$M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$$

where

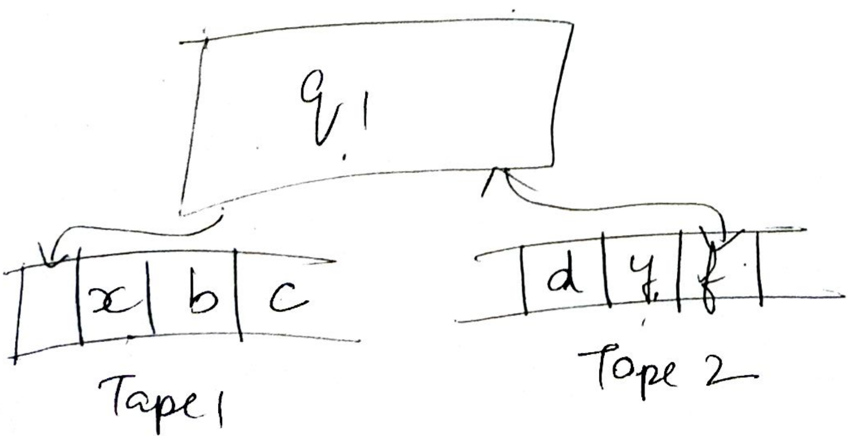
$$\delta : Q \times \Gamma^n \to Q \times \Gamma^n \times \{L, R\}^n$$

e.g. if $n=2$.

eg.



$$\delta(q_0, a, e) = (q_1, x, y, L, R)$$



To show equivalence between multitape and S.T.M we prove that.

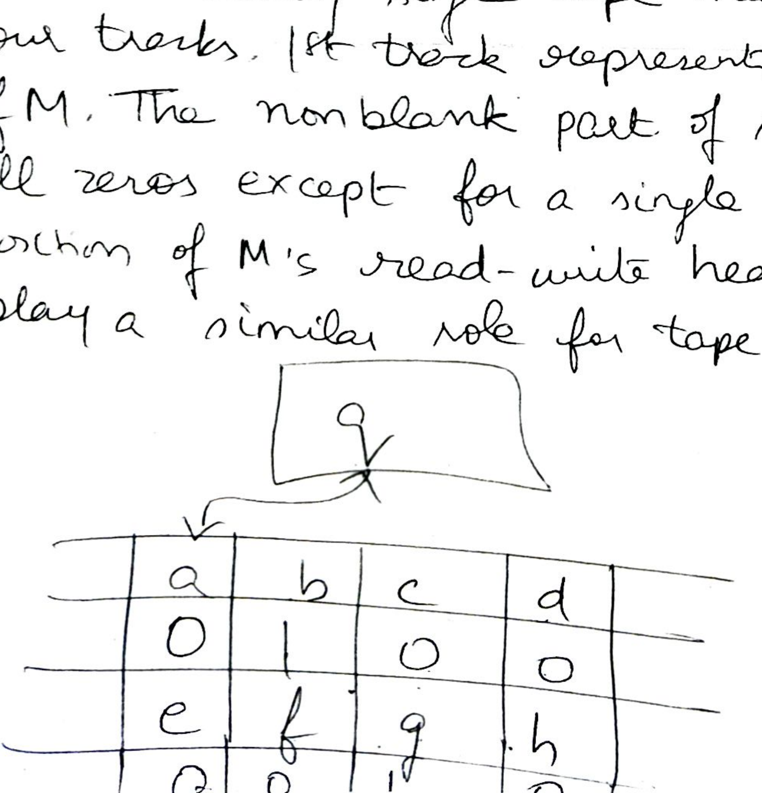
(1.) Any multitape T.M. M can be simulated by S.T.M $\hat{M}$ and

(2) Any S.T.M. can be simulated by a multitape one.

To prove (2), we can always elect to run a multitape machine with only one of its tapes doing useful work.

To prove (1.), the representation of multitape machine by a single-tape machine is similar to that used in simulation of an offline machine.

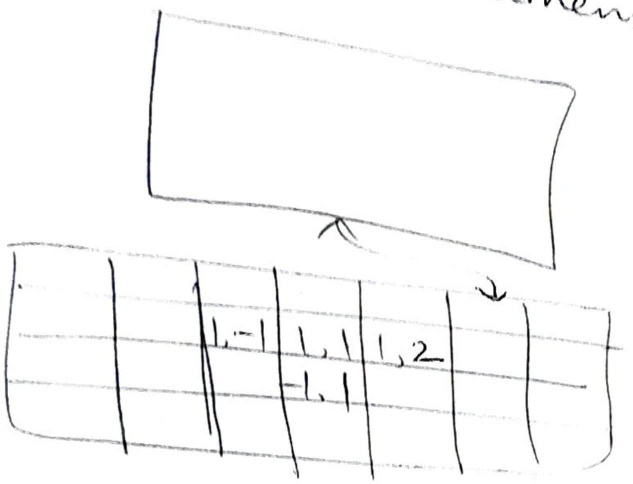
Consider, for example, 2-tape machine in the configuration depicted in the example. The simulating single tape machine will have four tracks. 1st track represents contents of tape 1 of M. The nonblank part of second track has all zeros except for a single 1 marking the position of M's read-write head. Tracks 3 and 4 play a similar role for tape 2 of M.



'Desk'

Multidimensional Turing Machine (2)

A multidimensional turing machine is one in which the tape can be viewed as extending infinitely in more than one dimension.

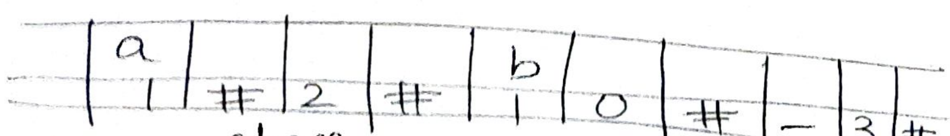


Formal definition of a 2-D T.M. involves a δ of the form:-

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R, U, D\}$$

where U and D specify movement of read-write head up and down respectively.

To simulate this machine on a S.T.M., we can use 2-track model. First, we associate an ordering or address with the cells of 2-D tape. This can be done in no. of ways eg. in 2-D fashion indicated in above figure, the 2-track tape of simulating machine will use 1 track to store cell contents and the other one to keep the associated address.



In the above configuration, in which cell (1,2) contains a and cell (10,-3) contains b.

The cell address can involve arbitrarily large integers, so address track cannot use a fixed size field to store addresses. Instead we use a variable field size arrangement, using some special symbols to delimit the fields.

Regular — Regular expression

CFG — stack Push & pop

stack ← not possible — CSL/G

Turing M|c with a stay option

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$$

Theorem: The class of T.M. with stay option is equivalent to the class of S.T.M's

i.e. we have to prove that

1) a S.T.M. can be simulated with stay option.
2) A stay option T.M. can be simulated with a S.T.M.

① ∵ T.M with stay option is an extension of S.T.M, ∴ any S.T.M, ∴ any S.T.M can be simulated one with a stay option

② To show that a T.M. with stay option can be simulated by S.T.M.

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ be a T.M. with stay-option.

Let $\hat{M} = (\hat{Q}, \hat{\Sigma}, \Gamma, \hat{\delta}, \hat{q_0}, \square, \hat{F})$ be a S.T.M.

* for each transition

$$\delta(q_i, a) = (q_j, b, R)$$

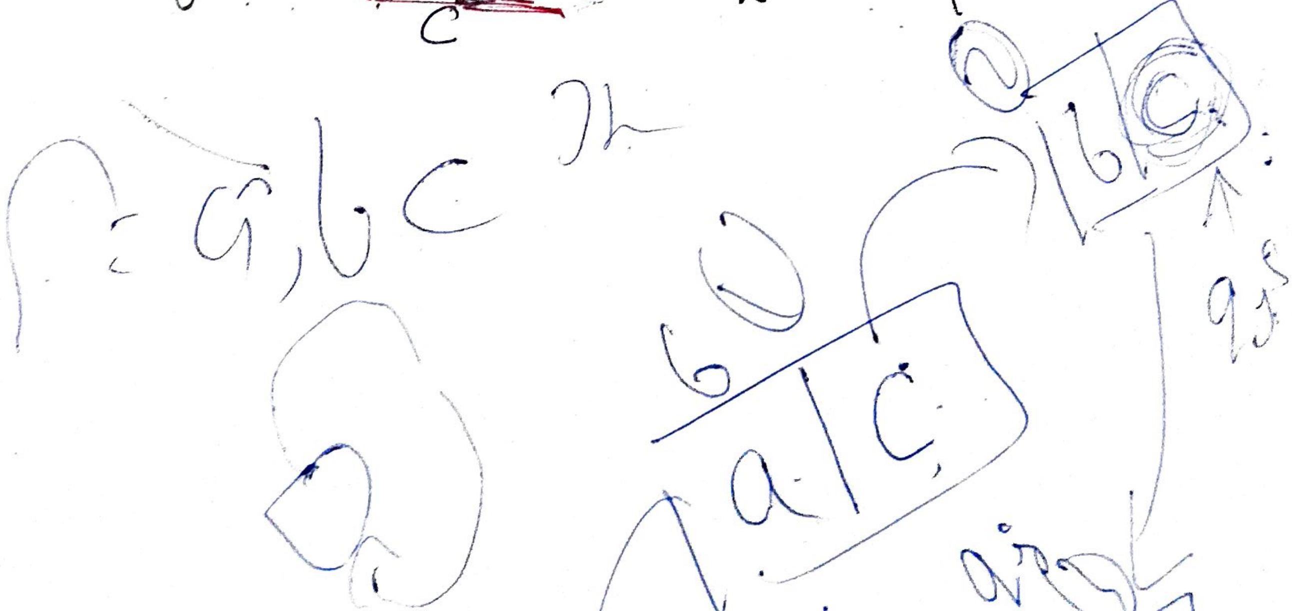
we add

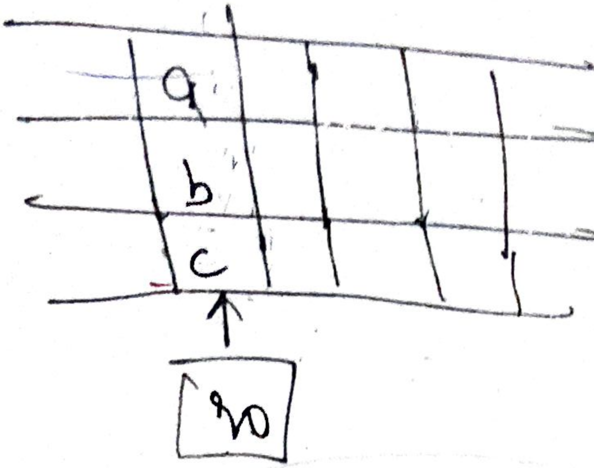
$\hat{\delta}(\hat{q_i}, a) = (\hat{q_j}, b, R)$ to $\hat{M}$

* for each transition
$\delta(q_i, a) = (q_j, b, L)$
we add

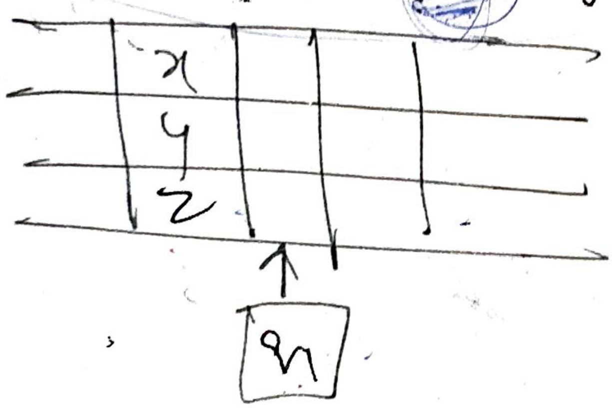
$\hat{\delta}(\hat{q_i}, a) = (\hat{q_j}, b, L)$ to $\hat{M}$

* for each transition
$\delta(q_i, a) = (q_j, b, S)$
we add
$\hat{\delta}(\hat{q_i}, a) = (\hat{q_{is}}, b, R)$
$\hat{\delta}(\hat{q_{is}}, c) = (\hat{q_j}, c, L)$
for all $c \in \Gamma$, $\hat{q_{is}} \in \hat{Q}$





$\delta(q_0, abc) = (q_1, xyz, R)$ ①



$\delta(q_0, abc) = (q_1, x, b, c, R)$

Theorem:- The class of multitrack T.M's is equivalent to class of S.T.M.

We have to prove that:-

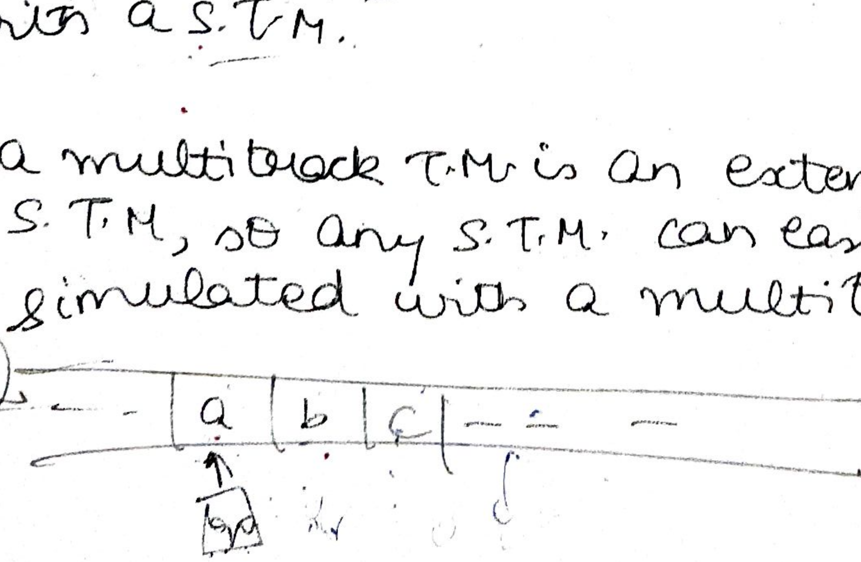
① A S.T. can be simulated with a multitrack T.M.

② A multitrack T.M. can be simulated with a S.T.M.

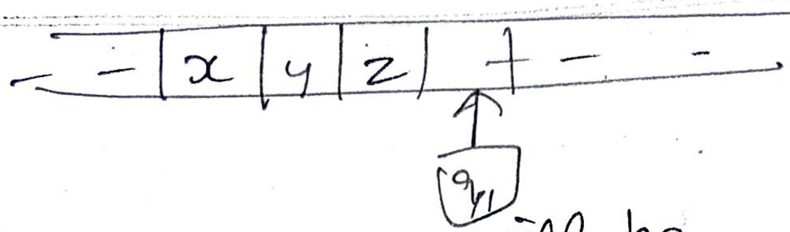
①

∴ a multitrack T.M. is an extension of S.T.M, so any S.T.M. can easily be simulated with a multitrack T.M.

②



(q_0, x, R)

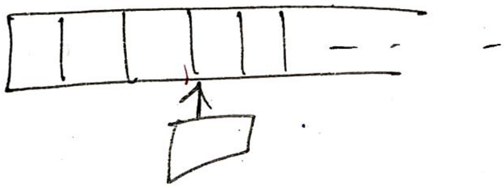


So 3 rules ~~that~~ ~~are~~ will be added, & they are:

$\delta(q_0, a) = (q_0, x, R)$
$\delta(q_0, b) = (q_0, y, R)$
$\delta(q_0, c) = (q_1, Z, R)$

T.M. with semi-infinite tape

In this T.M, the tape is unbounded only in one direction.

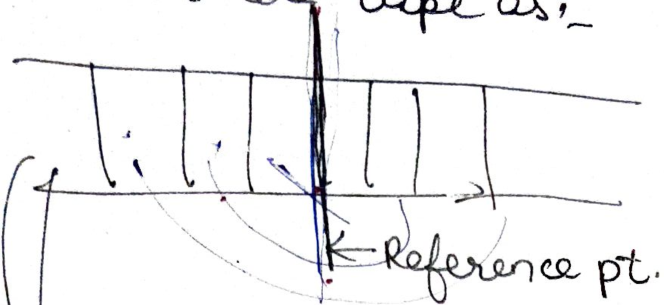


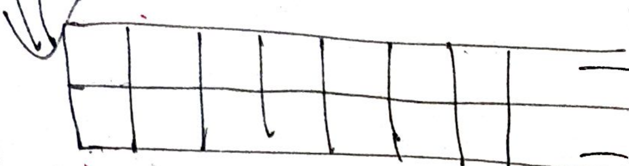
Theorem: The class of T.M. with semi-infinite tape is equivalent to the class of S.T.M. ie we have to ~~prove~~ ~~that~~ simulate:

1. A T.M. with semi-infinite tape with S.T.M.
2. A S.T.M. with T.M. with semi-infinite tape

(1) ∵ a S.T.M. is an extension of T.M. with semi-infinite tape ∴ a T.M. with semi-infinite tape can easily be simulated by a S.T.M.

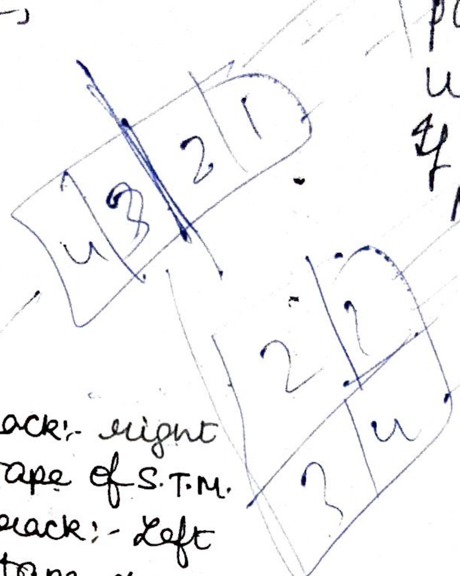
To simulate a S.T.M. by a machine with a semi infinite tape, we divide tape as:-



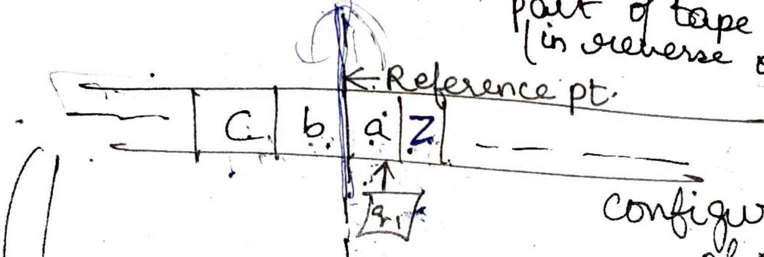


Upper track:- right part of tape of S.T.M.

Lower track:- Left part of tape of S.T.M. (in reverse order)

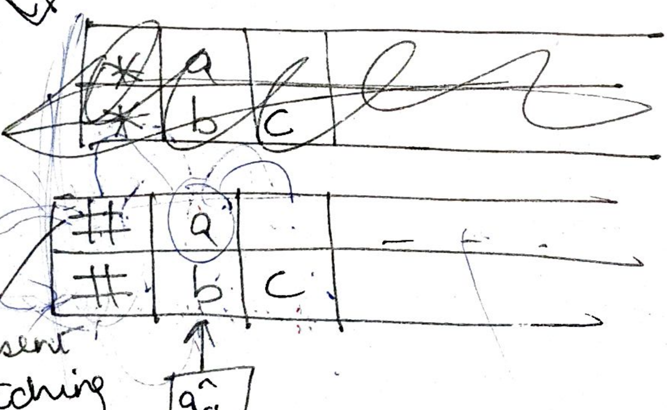


eg.



Configuration of M

S.T.M



Represent switching from 1 track to other

Configuration of $\bar{M}$

Semi Infinite Tape

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

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

If M... part... upper... If M... $\bar{M}$

T...

hine

If M's Read write head is at right part, $\hat{M}$ uses information on the upper Track.

If M's Read write Head is at left part $\hat{M}$ uses information on lower Track

To simulate

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

we add to $\hat{M}$

$$\hat{\delta}(\hat{q_1}, (a, b)) = (\hat{q_2}, (c, b), L)$$

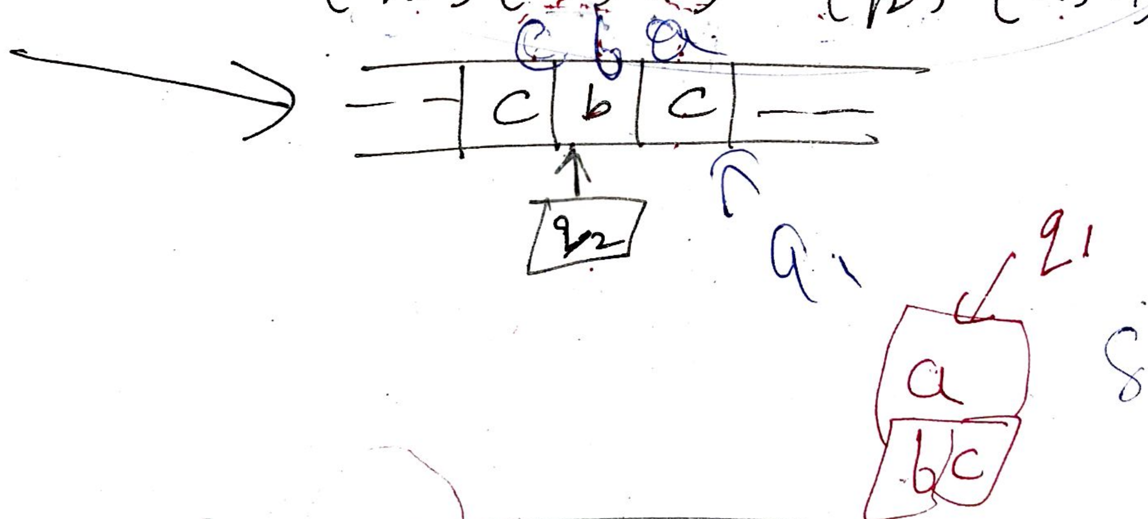
$$\hat{\delta}(\hat{q_2}, (\#, \#)) = (\hat{q_2}, (\#, \#), R)$$

right of S.T.M.

:- Left of S-T.M. order)

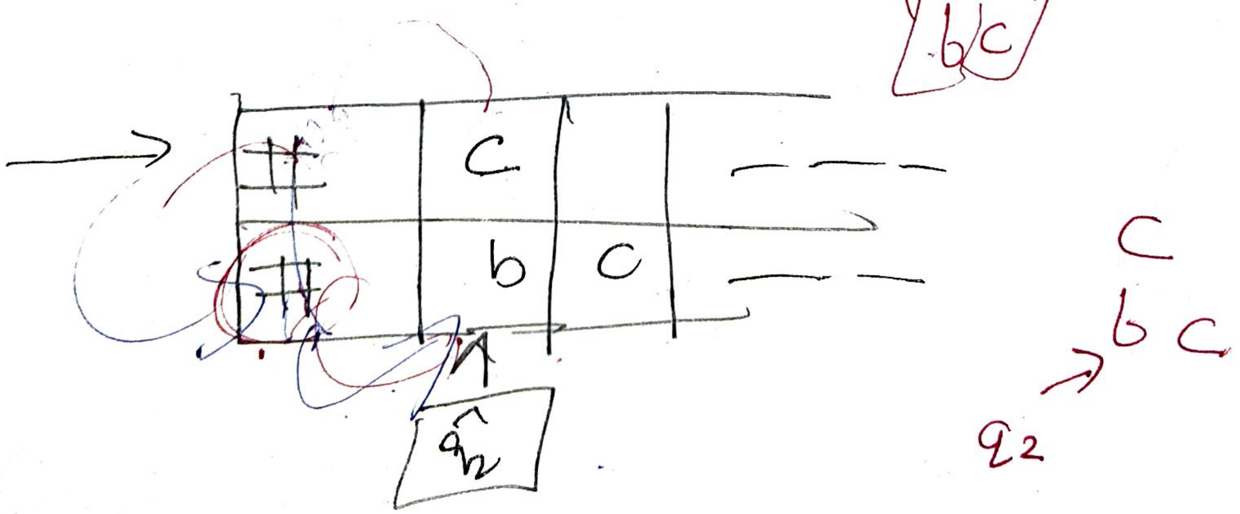
ation

S.T.M



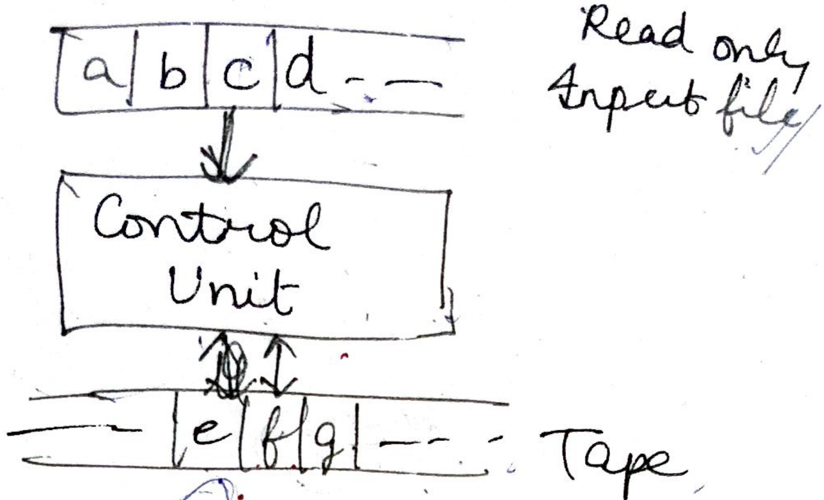
uration

finite Tape



Offline T.M.

T.M. with I/P file along with Tape.



$\delta : Q \times \Sigma \times \Gamma \to Q \times \Gamma \times \{L, R\}$

Σ is not necessarily subset of $\Gamma - \{\square\}$

$Q \times \Gamma = Q \times F(L,R)$

<u>Theorem</u>:- The class of offline T.M. is equivalent to the class of S.T.M.

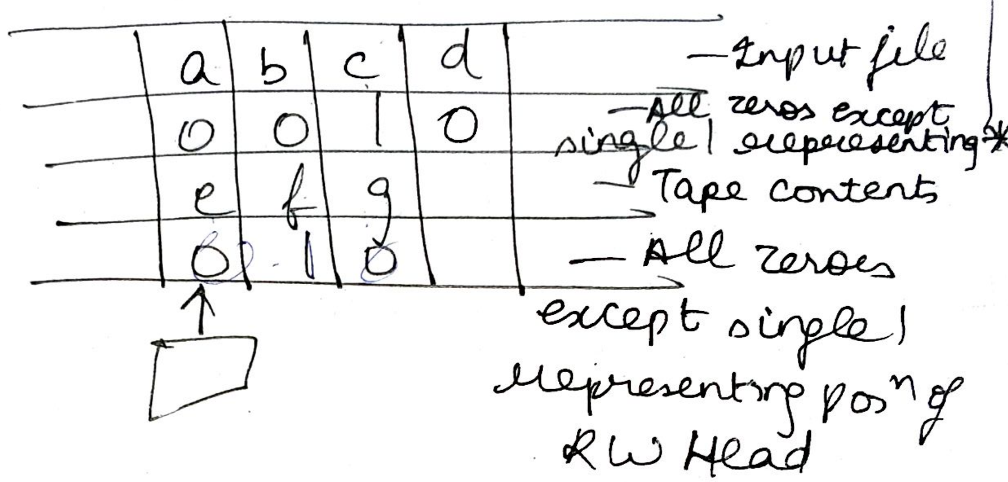
We have to prove that, we can simulate:-

1. a S.T.M. with an offline T.M.
2. An offline T.M. with a S.T.M.

(1) Copy input from tape of S.T.M. to the input file of offline T.M. Then offline T.M. can proceed in the same

way as the S.T.M.

② S.T.M. can simulate computation of O.T.M. by using a four track arrangement



Simulating of each Move of M requires a no. of moves of $\hat{M}$. Starting from some standard position, say left end, & with the relevant information marked by special end markers.

① $\hat{M}$ searches track 2 to locate the position at which the input file of M is read. The symbol found in corresponding cell on track 1 is remembered by putting the control unit of $\hat{M}$.

② Track 4 is searched for posn of RW of M using δ of M, remembered i/p & symbol on track 3, $\hat{M}$ move to another configuration & all four tracks of $\hat{M}$'s tape are modified to reflect the move of M.

Finally, the R/W Head of M returns to standard pos^n for the simulation of next move.

Multitape T.M. ⇒ many tapes, each tape has own head, input on only 1 tape.

~~Multidimensional T.M.~~

~~Example - 10.1~~

~~Linear - bounded automata (defn)~~

~~do from book.~~

Non deterministic turing machine :- For a given configuration, more than one move are possible.

ie. $\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$

eg. if a T.M. has transition specified by
$\delta(q_0, a) = \{(q_1, b, R), (q_2, c, L)\}$
It is nondeterministic
∵ both moves $q_0 aaa \vdash b q_1 aa$
$q_0 aaa \vdash q_2 \square caa$
caa

T.M. is said to accept w if there is any possible sequence of moves s.t. $q_0 w \overset{*}{\vdash} x_1 q_f x_2$ with $q_f \in F$.

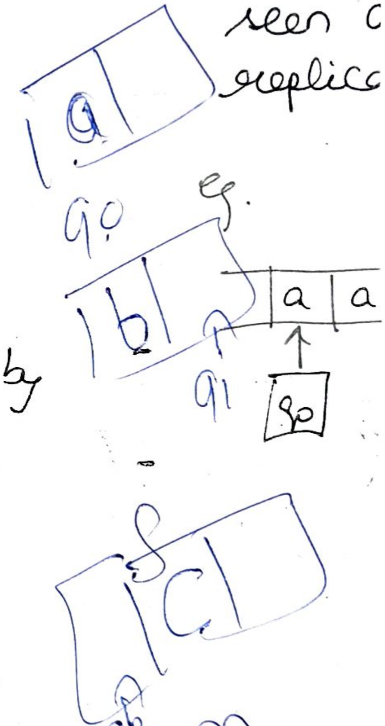
Theorem and are Proof, can

1.) Nonc (dete

2.) S.T

3.) Ever T.M.

4.) A n seen c replica



TM ⇒ D

$Q \times \Gamma \Rightarrow Q \times \Gamma$

TM → deter

nonde

$Q \times \Sigma \rightarrow Q$

NFA $Q \times \Sigma \rightarrow 2^Q$

Pda → de

Theorem:- The class of deterministic T.M. and the class of non deterministic T.M. are equivalent.

Proof, we have to prove that we can simulate:-

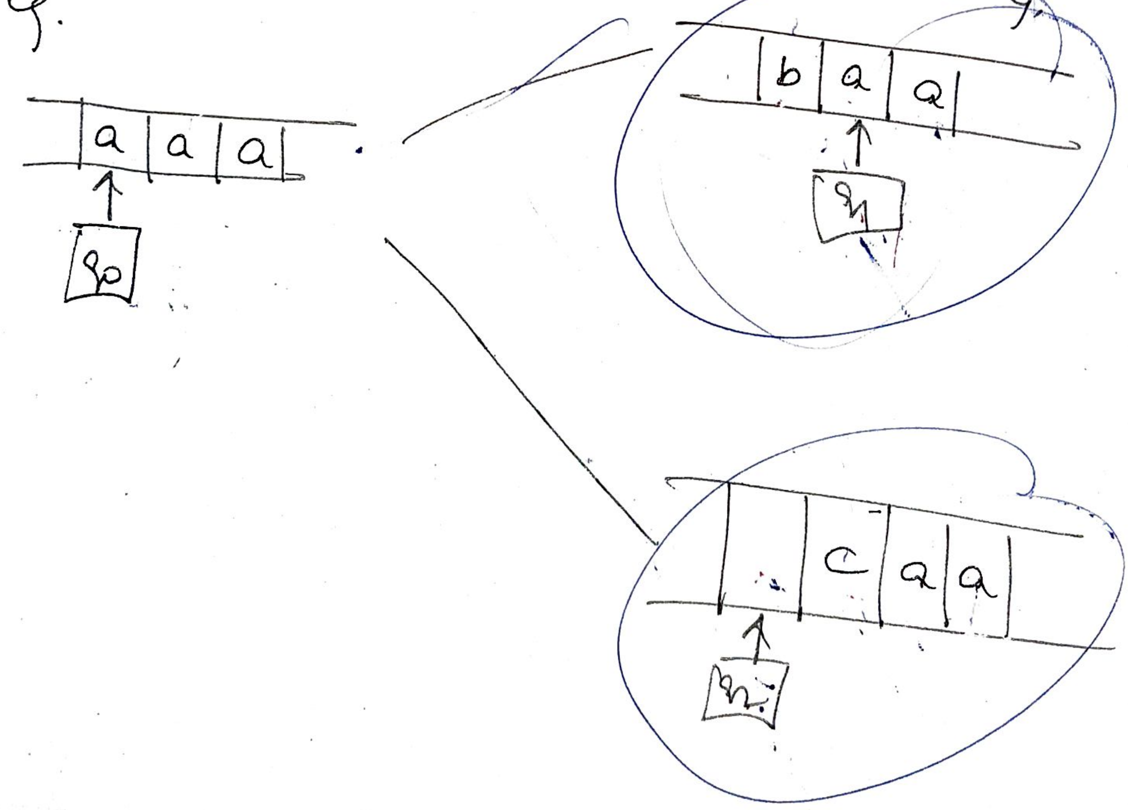
1.) Nondeterministic T.M. with S.T.M (deterministic)

2.) S.T.M. with Nondeterministic T.M.

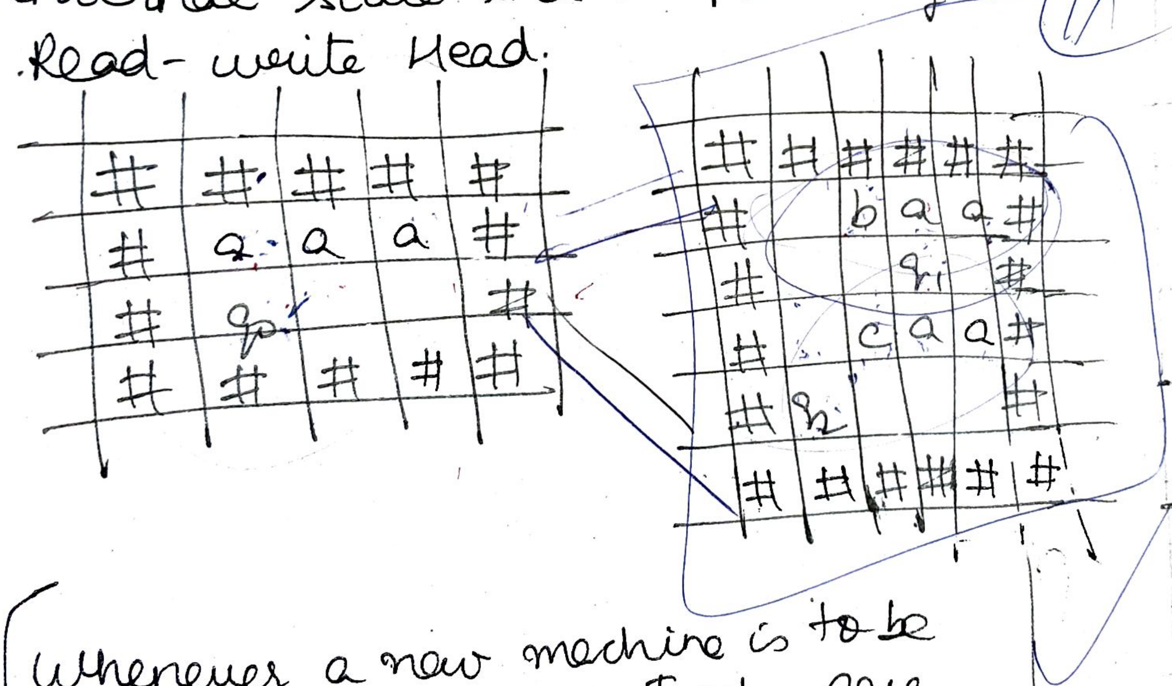
2.) Every S.T.M. is also Non deterministic T.M.

1.) A non deterministic machine can be seen as one which has ability to replicate itself whenever necessary.

eg.



whenever more than one move is possible, the machine produces as many replicas as needed & gives each replica the task of carrying out one of the alternatives.
To represent these replicas, we use two dimensional T.M. Each pair of horizontal track represents one machine, top track containing the machine's tape, the bottom one for indicating its internal state and the position of Read-write Head.



Whenever a new machine is to be created, two new Tracks are started with the appropriate information and active tracks are bracketed with special markers

Universal T.M.

- A T.M. is a special purpose computer. Once δ is defined, the machine is restricted to carrying out one particular type of computation.
- Digital computers, are general purpose machines that can be programmed to do different jobs at different times.
- Universal T.M. is general purpose reprogrammable machine.

A universal T.M. M_U is an automaton, that given as input the description of any T.M. M and a string w, can simulate the computation of M on w.

We will use following standard way of describing T.M's.

For any T.M., →

$Q = \{q_1, q_2, \ldots, q_n\}$ with q_1, the initial state, q_2 the single final state, & $\Gamma = \{a_1, a_2, \ldots, a_m\}$ where a_1 represents the blank.

We will represent q_1 by 1

q_2 by 11

q_3 by 111

We will represent a_1 by 1

a_2 by 11

a_3 by 111

L-1 R-11

8, 9, 10, 11, 12, 13

The symbol 0 will be used as a separator. We will represent

L by 1

& R by 11

The transition function δ is encoded according to this scheme with the arguments & result in some prescribed format.

eg. $\delta(q_1, a_2) = (q_2, a_3, L)$

we will denote it by

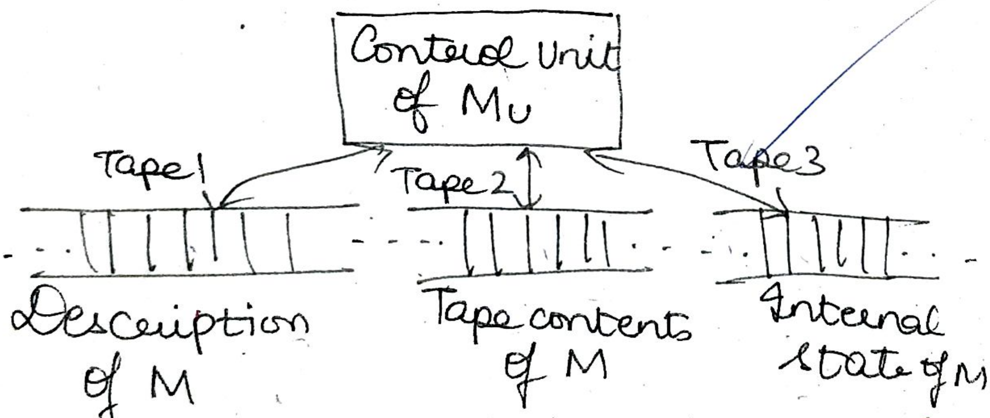
1 0 11 0 11 0 111 0 1

$q_1, a_2 \quad q_2, a_3, L$

2 3

11 0 111

Block diagram of Universal T.M.



For any input M + w, tape 1 will keep an encoded defination of M. Tape 2 will contain the tape contents of M + tape 3 the internal state of M.

M_U looks ~~the~~ first at the contents of tapes 2 + 3 to determine the configura-tion of M. It then consults tape 1 to see what M would do in this configuration. Finally tapes 2 + 3 will be modified to ~~reflect~~ reflect the result of the move.

Every T.M. can be represented by a strip of 0's + 1's.

Check   eg. set of real no's

Set — Finite set — countable

Set — Infinite set — countable / uncountable

A set is infinite countable if there is 1–1 correspondance between its elements + set of natural numbers. Else the set is infinite uncountable.

eg. of infinite countable – set of natural no's, set of integers, set of even no. set of rational numbers.

Integers: $\ldots -3, -2, -1, 0, 1, 2, 3 \ldots$

$$\begin{array}{ccccccccc} 0, & 1, & -1, & 2, & -2, & 3, & -3, & \ldots \\ | & | & | & | & | & | & | & \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & \end{array}$$

infinite