Madhusudan.G.
Asst. prof
CS&E.

Turing Machine – Unit-5.

A Turing Machine can be defined as a set of 7 tuples

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

$Q \rightarrow$ Non empty set of states.

$\Sigma \rightarrow$ Non empty set of Symbols.

$\Gamma \rightarrow$ Non empty set of Tape Symbols

$\delta \rightarrow$ Transition function defined as

$$Q \times \Sigma \rightarrow \Gamma \times (R/L) \times Q$$

$q_0 \rightarrow$ Initial State.

$b \rightarrow$ Blank Symbol.

$F \rightarrow$ Set of Final states (Accept & Reject State)

Thus, the production rule of Turing machine will be written as $\delta(q_0, a) \rightarrow (q_1, Y, R)$

Turing's Thesis :- Turing's Thesis states that any computation that can be carriedout by mechanical means can be performed by some turing machine

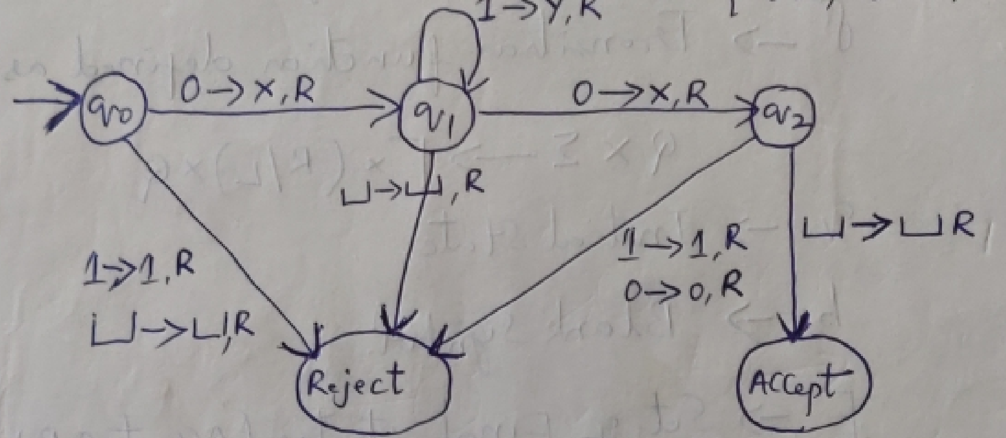
Few arguments for accepting this thesis are:

i) Anything that can be done an existing digital computer can also be done by turing machine.

ii) No one yet been able to suggest a problem solv[illegible] by what we consider an algorithm, for which a turing machine program cannot be written.

1) Design a Turing Machine which recognizes the language $L = 01^*0$

$L = \{010, 0110, ----\}$



x	y	y	x	⊔	⊔

Recursively Enumerable Language :- A language L and Σ is said to be Recursively Enumerable, if there exists a turing machine that accepts it.

2) Design a turing machine which recognizes the language $L = \{0^N 1^N \text{ for } N \geq 1\}$

Algorithm: i) Change 0 to x

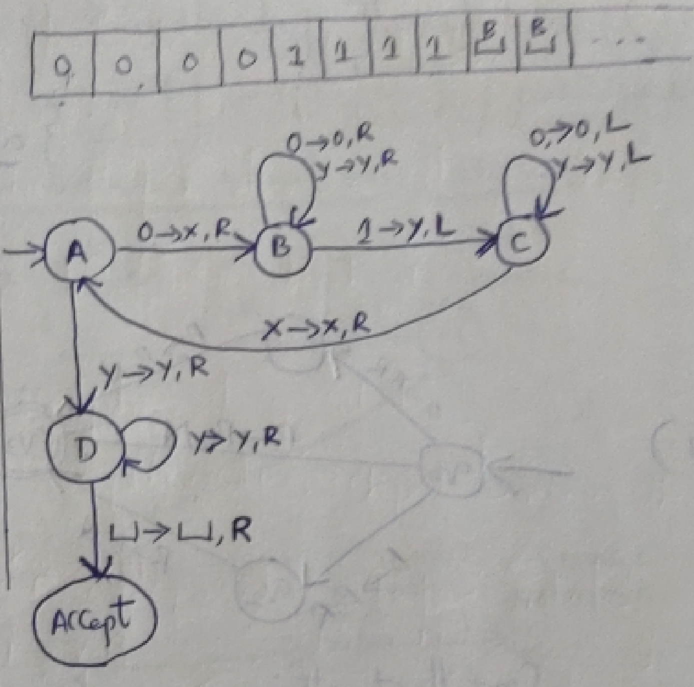
ii) Move Right to first '1'
If None "Reject"

iii) Change '1' to 'y'

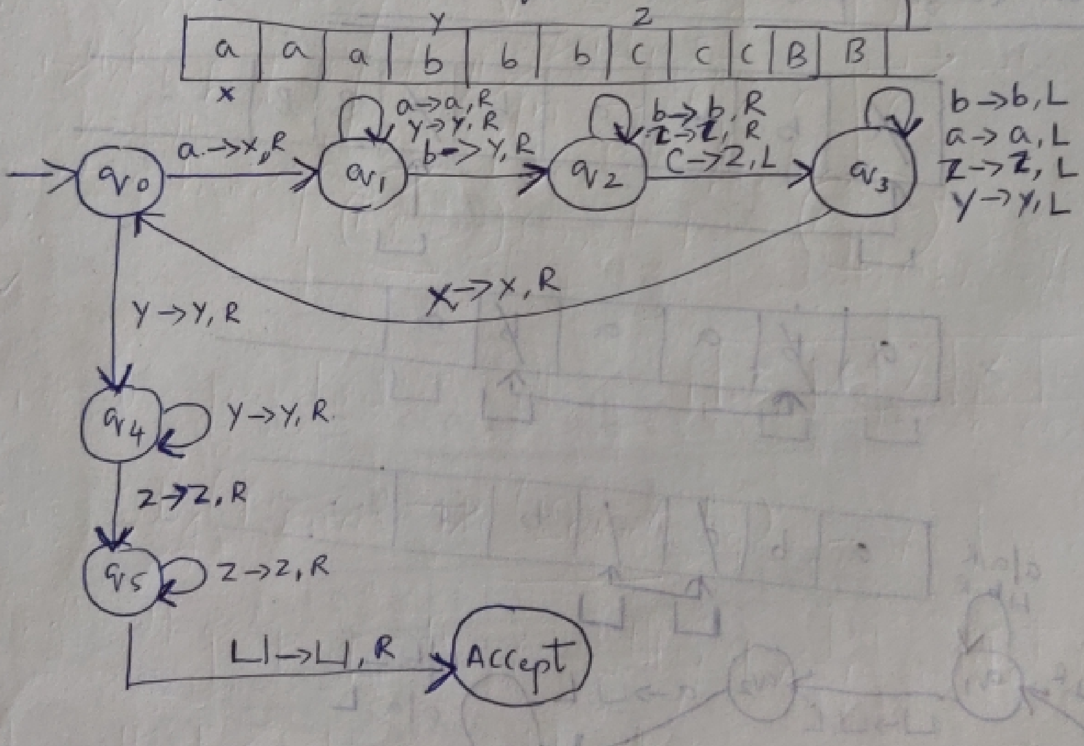
iv) Move Left to leftmost '0'

v) Repeat the above steps until no more '0's

vi) Make Sure no more '1's remain



2) Design a turing machine $L = \{a^n b^n c^n \mid n \geq 1\}$



3) Design a turing machine to accept the language

$L_1 = \{ WW^R \mid w \in (a,b) \}$

Even length palindrome $WW^R = \{aabbaa\}$

Smallest string : ϵ

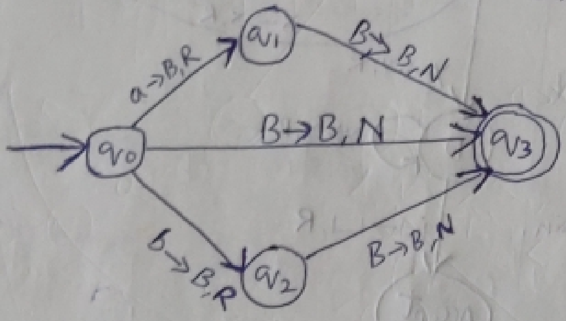
ii) odd length palindrome $= \{ waw^R$ OR $wbw^R \}$

$= \{ \underbrace{abb}_{w} \underset{\downarrow}{a} \underbrace{bba}_{w^R} \quad \underbrace{abb}_{w} \underset{\downarrow}{b} \underbrace{bba}_{w^R} \}$

Seperator Seperator

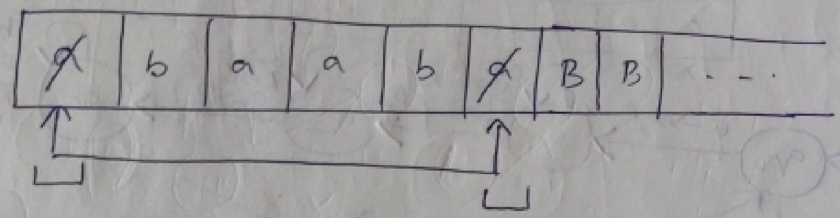
Smallest string: a Smallest string: b

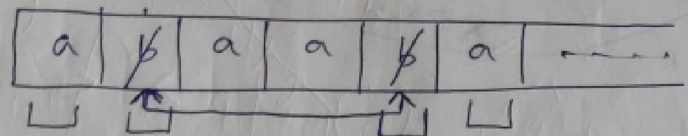
i)

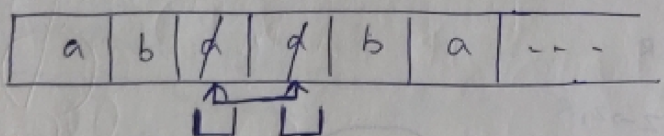


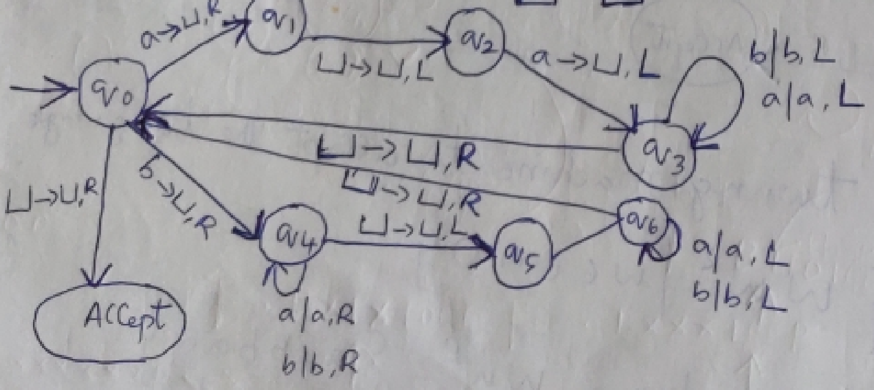
Smallest string of ϵ, a & b

Even length palindrome : abaaba



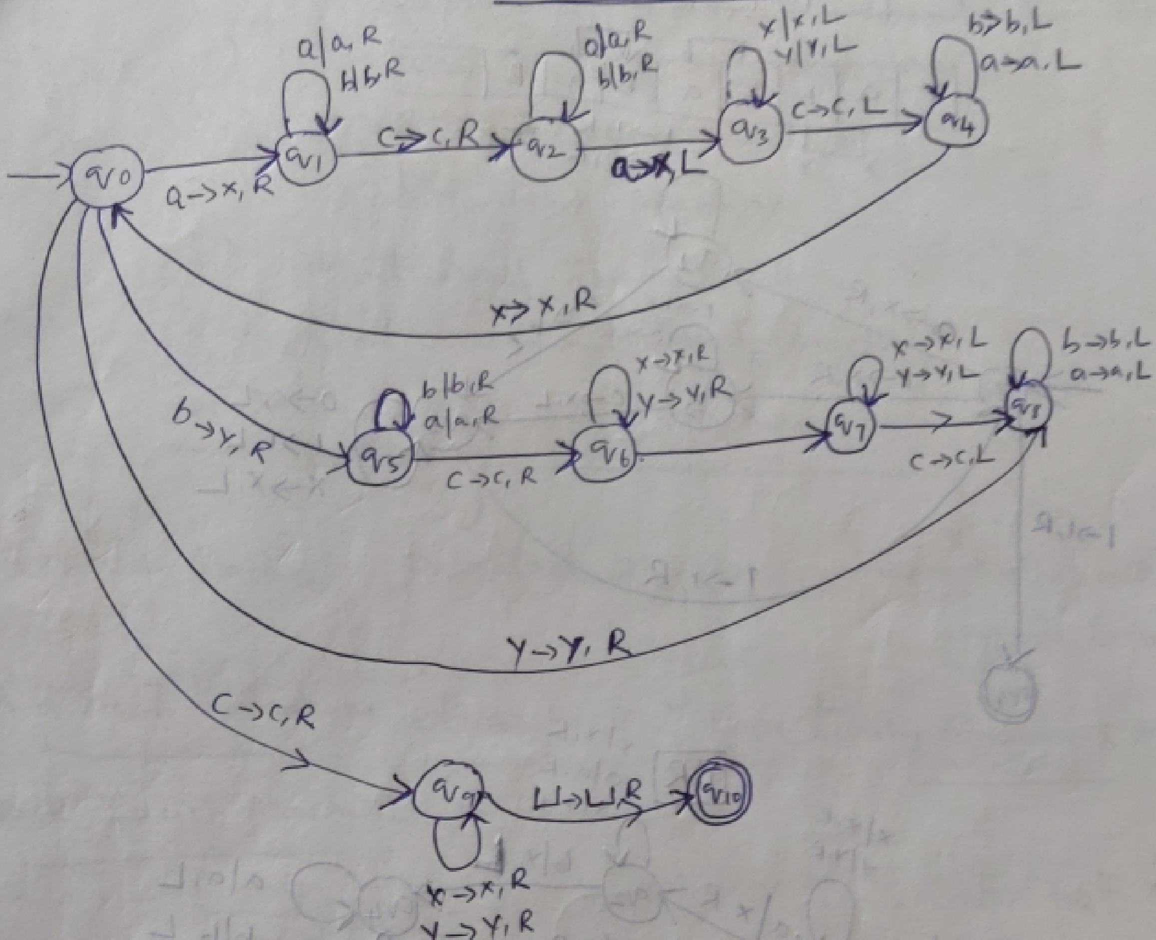






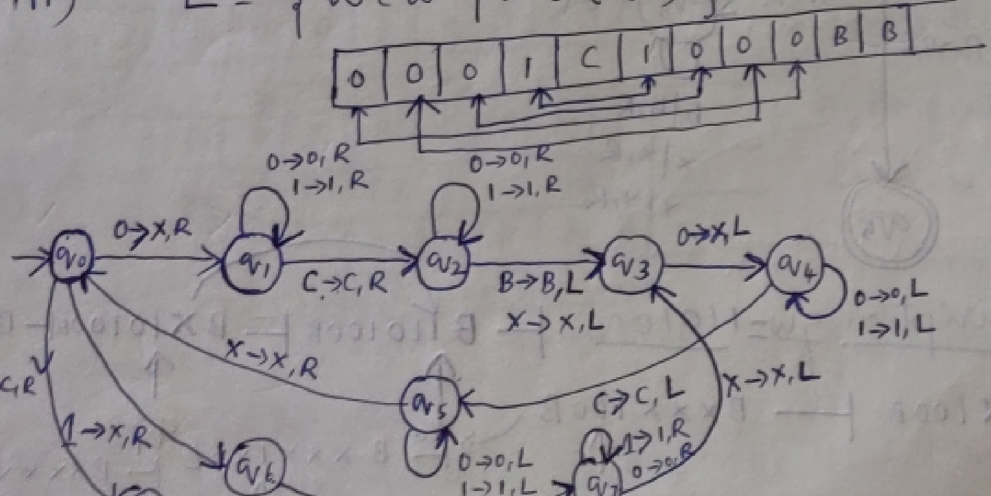
ii) $L = \{ wcw^R \mid w \in (a+b)^* \}$
$= \{ waw^R \mid wbw^R \mid w \in (a+b)^* \}$ } odd length palindrome

$w = \underline{aabbabcaabbab}$ $\{\underline{wcw}\}$



$w = \underline{aabbabcaabbab}$

iii) $L = \{ wcw^R \mid w \in (0+1) \}$

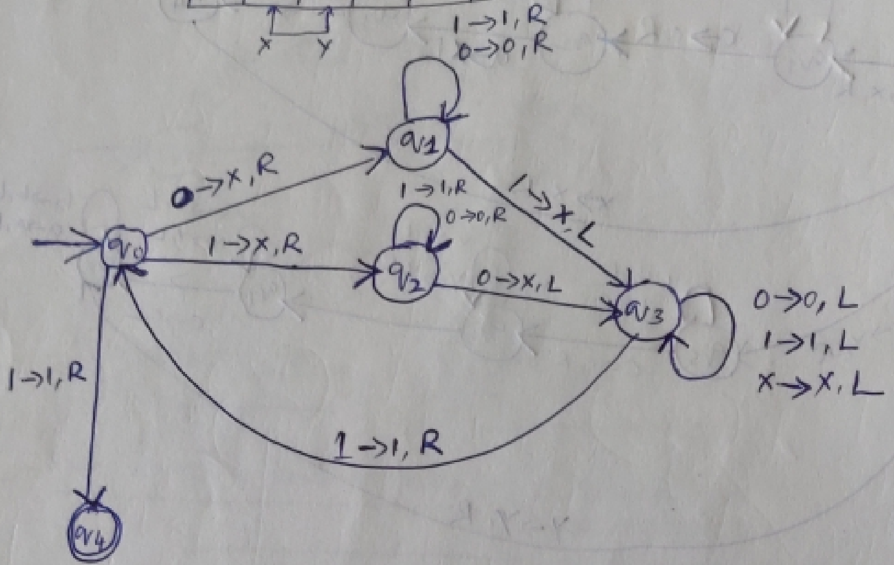


4) Design a turing machine to accept Equal numbers of 1's.
$L = \{0^n 1^n\}$

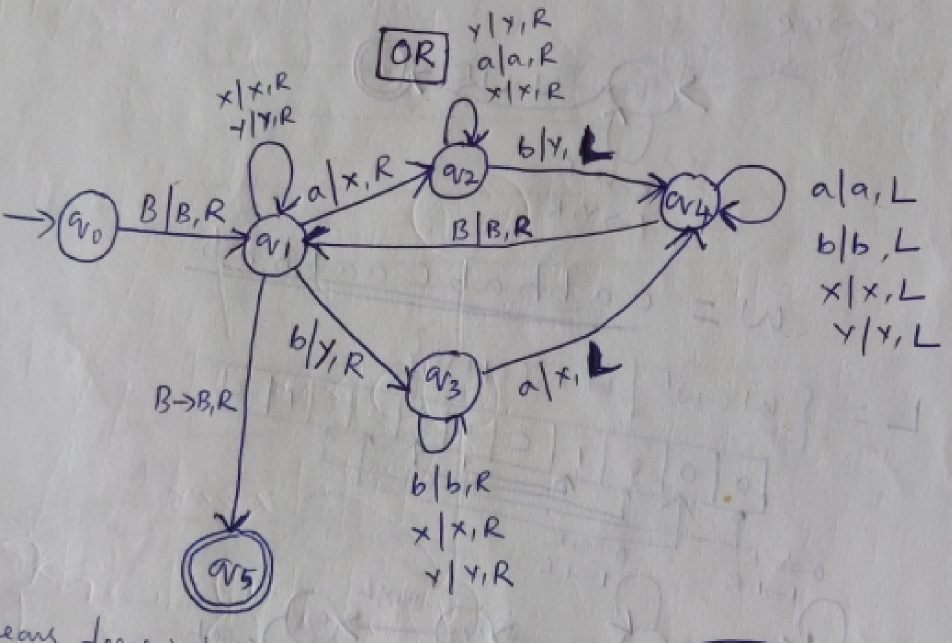
0	0	1	1	1	1	0	0	B

B	a	b	b	a	b	a	B	...

X Y



OR



Instataneous description ζw = 110100

q0 100B ⊢ BX1X100B ⊢ BXXX100B ⊢ B 10100B ⊢ BX10100B ⊢ BX10
q1, q2, q0, q1, q2

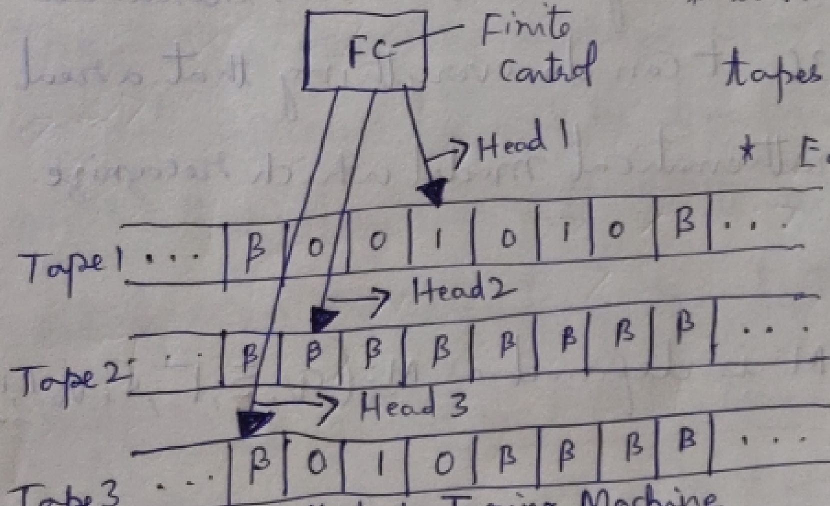
⊢ BXXX1X0B ⊢ BXXXXX0B
q1

⊢ BXXXXXXB → q1 Halt.

Different types of Turing Machines :- The basic model of the turing machine is Equivalent to many other modified versions. They are :

i) Non Deterministic turing Machine
ii) Two way infinite tape.
iii) Multitape turing Machine.
iv) offline Turing Machine
v) Multi head turing Machine
vi) Multidimensional Turing Machine
vii) Restricted Turing Machine
- Counter Machine
- Multistack Machine
- Turing Machine with semi infinite tapes.

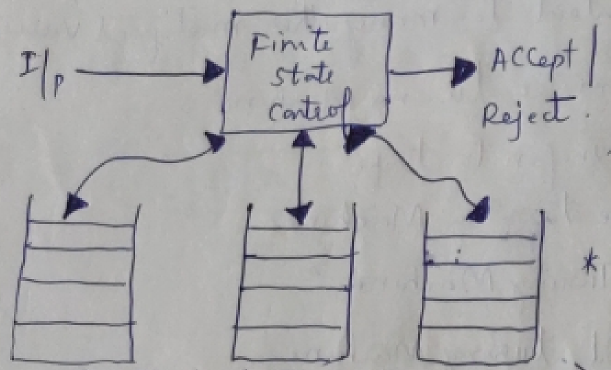
With a neat diagram, describe Multitape turing Machine & Multi-stack turing Machine :-



Multitape Turing Machine

* it has a finite Control with many tapes say 'k' tape heads.
* Each tape is two way infinite.
* Initially the input will be on the first tape and the other tapes are kept blank.
* Depending on the states of the system and the symbol scanned by each of the tape heads, the TM can
i) Change the state.
ii) print a new symbol on each of cells scanned by the tape heads
iii) Move each tape head independently (some to right, some to left and some other stationary)

Multi-stack turing machine :- A 'k' stack machine is a deter-ministic PDA with k stacks.



Multi-stack Turing Machine

* it has a finite control which is in one of the state.
* it has a finite stack alphabet which is used for all stacks.
* A move of the multistack machine is based on:

i) The state of the finite control.
ii) The input symbol read.
iii) The top stack symbol on each stack.

In one move the multi stack machine can:

i) Change to a new state.
ii) Replace the top symbol of each with a string or more stack symbols.

With a neat diagram, describe the turing machine :- it is a more powerful model than many models. it was introduced by Alan Turing in 1936. it can do everything that a real computer can do. it is mathematical model which recognize enumerable languages.

The basic model of a TM is defined as $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ where

Q - A finite set of states.

Γ - A finite set of allowable tape symbols.

Σ - a subset of Γ not including B is the set of input symbols

$\delta \rightarrow$ The next move function (transition function) from

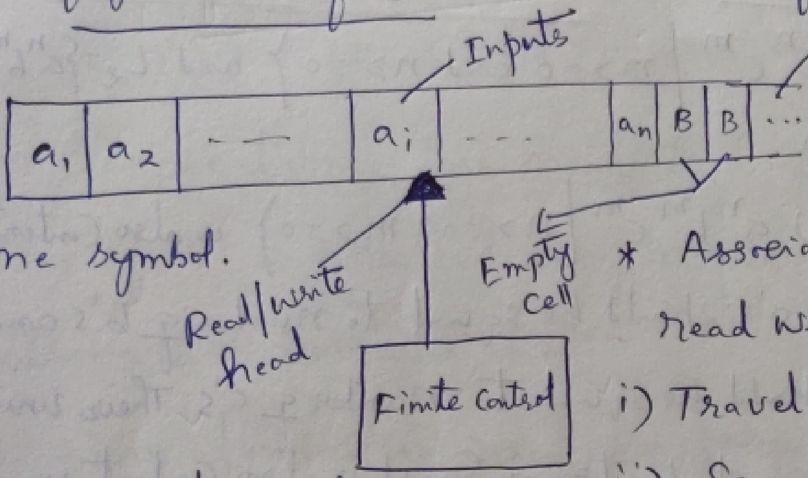
$$Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

$q_0 \rightarrow$ Starting state

$B \rightarrow$ a Symbol of Γ, is the blank.

$F \rightarrow$ Set of accepting states.

Configuration of TM :- It has a finite control, an input tape is divided into cells, each of which is capable of holding one symbol.



A general model of Turing Machine

* Associated with the tape head is a read write head (tape head) that can:

i) Travel right or left on the tape.

ii) Scan one cell of the tape at a time.

iii) Read and write a single symbol on each move.

The TM's temporary storage is on the tape. Whatever input & output is necessary will be done on the tape. The tape has a left most cell but is infinite to its right. The tape head is initially set to scan the content of the leftmost cell.

Differences between FA & TMs

1. A TM can both write on the tape and read from it.
2. The read write head can move both left and to the right.
3. The tape is infinite.
4. The special states for rejecting accepting take immediately effect.

Moves of the TM :- Depending on the state of the infinite control the symbol on the cell scanned by the tape head, TM changes state, print a new symbol on the tape cell scanned & moves left or right or stationary.

problem 1 :- Let L_1 and L_2 are two context free languages, their union $L_1 \cup L_2$ and concatenation $L_1 L_2$ are also context free.

Ans :- Assume $L_1 = \{a^n b^n c^m \mid m >= 0 \text{ and } n >= 0\}$ and $L_2 = \{a^n b^m c^m \mid n >= 0 \text{ and } m >= 0\}$. Then

$L_3 = L_1 \cup L_2 = \{a^n b^n d^m \cup a^n b^m c^m \mid n >= 0, m >= 0\}$ is also context free. L_1 says number of a's should be equal to number of b's and L_2 says number of b's should be equal to number of c's. Their union says either of two conditions to be true. So it is also context free languages.

Let $L_1 = \{a^n b^n \mid n >= 0\}$ and $L_2 = \{c^m d^m \mid m >= 0\}$. Then $L_3 = L_1 . L_2$
$= \{a^n b^n c^m d^m \mid m >= 0 \text{ and } n >= 0\}$.

L_1 says number of a's should be equal to number of b's and L_2 says number of c's should be equal to number of d's. Their concatenation says first number of a's should be equal to number of b's, then number of c's should be equal to number of d's. So, we can create a PDA which will first push for a's, pop for b's, push for c's then pop for d's. So it can be accepted by PDA, hence it is a context free.

State & prove pumping lemma for regular languages.

For any regular language L, there exists an integer n, such that for all $x \in L$ with $|x| \geq n$, there exists $u, v, w \in \Sigma^*$, such that $x = uvw$ and

(i) $|uv| \leq n$ —— ①

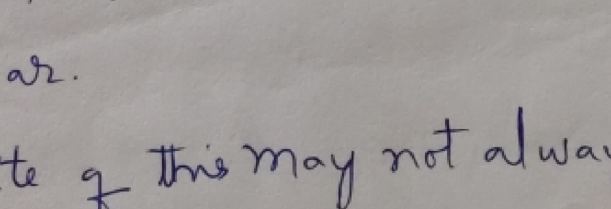
ii) $|v| \geq 1$ —— ②

iii) for all $i \geq 0$: $uv^iw \in L$ —— ③

In Simple terms, this means that if a string v is 'pumped', i.e., if v is inserted any number of times, the resultant string still remains in L.

pumping lemma is used as a proof for irregularity of a language. Thus, if a language is regular, it always satisfies pumping lemma. If there exists atleast one string made from pumping which is not in L, then L is surely not regular.

The opposite of this may not always be true. That is, if pumping lemma holds, it does not mean that the language is regular



For example, let us prove $L_{01} = \{0^n 1^n \mid n \geq 0\}$ is irregular.

Let us assume that L is regular, then by pumping lemma the above given rules follow

Now let $x \in L$ and $|x| \geq n$, So, by pumping lemma the above given rules follow: There exists u, v, w such that ① ② & ③ hold.

we show that for all u, v, w ① — ③ does not hold:

If ① and ② hold then $x = 0^n 1^n = uvw$ with $|uv| \leq n$ and $|v| \geq 1$.

So, $u = 0^a$, $v = 0^b$, $w = 0^c 1^n$ where: $a + b \leq n$, $b \geq 1$, $c \geq 0$, $a + b + c = n$

But, then ③ fails for $i = 0$

$uv^0w = uw = 0^a 0^c 1^n = 0^{a+c} 1^n \notin L$, Since $a + c \neq n$.