Theory of Computation
CS-501

Pooja Singh
0902CS181030

Assignment-5

1. What do you mean by Turing Machine ? Explain multiples tapes Turing Machine.

Ans- Turing machine recognize the recursive enumerable language.

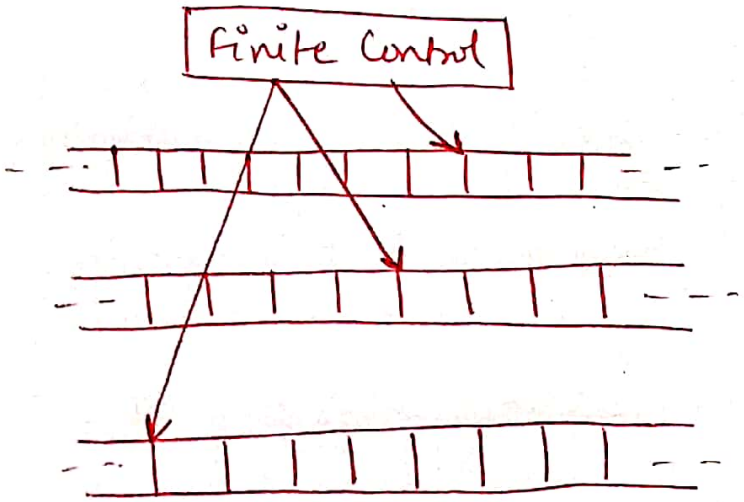
Turing machine is more powerful than any other automata such as finite automata, PDA & LBA.

→ Turing machine accepts recursive enumerable language by using universal acceptance mechanism Turing machine enumerates the recursive enumerable language.

→ Turing machine computes the partial recursive function. Turing Machine can be modelled as Deterministic (DTM).

Multi Taps Turing Machine

→ It consists of a finite control with several tape with its own controlled R/W heads as



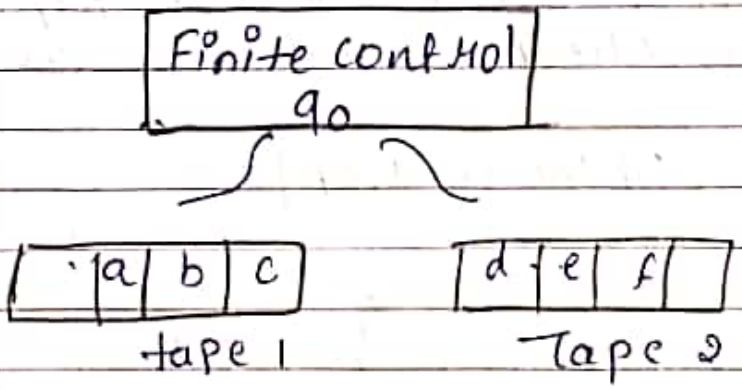
Each tape is infinite in both direction one single move depending upon the state of the finite control and the symbol scanned by each of the tape heads

— change state
— Print a new symbol on each of the cells scanned by its tape head

Move each of its tape heads independently One cell to the RL
Initially the input appears on the first tape and the other tapes are blank

Formal defination of Multitape TM

It requires a modified transition function. Typically we defined an n-tape Machine by M
$(Q, \Sigma, \Gamma, \delta, q_0, B, F)$



Q. Design a Turning Machine to accept the language

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

The Turing machine M which accept L $\{a^n \# a^n \mid n \geq 1\}$ is defined as

$M = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}, \{abc\}, abc \times B, \delta, q_0, B, \{q_6\})$

present state	0 a	1 b	0 c	X	B
→ q_0	(q_1, X, R)			(q_0, X, R)	(q_1, B, R)
q_1	(q_1, a, R)	(q_2, X, R)		(q_1, X, R)	
q_2		(q_3, X, R)			
q_3		(q_3, b, R)	(q_4, X, R)	(q_3, X, R)	
q_4		(q_5, C, L)			(q_6, B, L)
q_5	(q_5, a, L)			(q_5, X, L)	(q_0, B, R)
q_6				(q_6, B, L)	(q_7, B, R)
(q_7)		accept			

Steps of processing string

→ q_0 is inital state If input = a then replace (a, X) and change state $q_0 \rightarrow q_1$ (direction → right)

→ State = q_1 for each a (0) → no changes

→ State 2 = q_2 for each 1 → no change

→ q_3 is the state to check wether string is completely traversed or not

→ q_4 is the state to bring the head of turning machine to the start of the string

Q.3 Explain properties of recursive and recursively enumerable languages

Recursively Enumerable Languages :-

A language L is said to be recursively enumerable if their exists a turning machine that accept L

This defination implies only that there exist a turing machine M such that every $w \in L$.

$$q_0 w \vdash^* M^{w_1} q_f {}^{w_2}.$$

q_f — final state

Recursive Language :- A language L on Σ is said to be recursive if their exist a turning machine M that accept L and that halts on every $w \in \Sigma^*$ In other word, we can say that a language is recursive iff there exists a membership algorithum for it or if ther TM that recognize

If a language is recursive than their exist an easily constructed M enumeration procedure

Suppose M is a turning machine that determines membership in a recursive language L. First we construted another turning machine say 'M' that Generate all string in Σ^+ in proper order say $w_1, w_2, \ldots$ as these string are generated they become the input to M.

There is also an enumeration procedure for every recursively enumerable language but it is not easy to see it. It's e that every language for which an enumeration procedure exists is recursively enumerable.

We simply compare the given input string against successive strings generated by the enumeration procedure.

If $w \in L$ than will eventually find a match and the process can be permitted.

Q. 4 Explain P, NP and NP complete with example

P :-

→ P is the class of problem which we can solve by a single tape deterministic turing machine in polynomial time

$$P = \bigcup_{K>1} DTIME(n^K)$$

Example of P-class problem is - the Path Problem s to t ?

NP **

→ NP is the class of problem which can be solve by a non-deterministic turing machine or it is class of these languages which can be accepted by a non-deterministic turning machine is polynomial time

so

$$NP = \bigcup_{K>1} NTIME(n^K)$$

The type of NP classes problem are given below

Example

→ Subset sum problem
→ Graph colouring problem
→ Vertex cover problem

NP Complete :-

The study of the relation between the complexity classes P & NP has generated particular interest among computer scientists At the root there is the equat

$$P = NP$$

This is one of the fundamental unsolved problem is the theory of compution

NP complete problem is one that is as hard as any NP problem and is somesence is equivalent to all of them.

A language L_1 is said to be polynomial time reducible to some language L_2 if their exist a determinti

Turning Machine by which any U_1 in the alphabet of L_1 can be transformed in polynomial time to a U_2 in the alphabet of L_2 in such a way that $U_1 \in L$

A language $L \subseteq \{0,1\}^*$ is NP complete

Q.5 Write a short note on vertex cover problem and Hamiltonian path problem

Vertex cover problem :-

→ Let G = (V, E) be an (undirected) graph with set of vertices V and edge E

→ A subset A ⊆ V is said to be vertex cover of G if for every edge (v, w) in E at least one of v or w is in A

→ the vertex cover of size k or less ?

→ To represent this problem as a language L_VC consisting of string of the form ⟨G, k⟩ in binary

→ v_i represented by v followed by i in binary add a list of edge where

→ (v_i, v_j) is represented by the codes for v_i, v_j surrounded by parenthesis. L_VC consists of all such string representing G and G such that G has a vertex cover of size k or less