Varun Khadayate A016 16/9/20

Camlin Page
Date

F.L.A.T

Assignment - IV

Q2 Decidabienmeralitity

A problem is said to be decidable if we can always construct a corresponding algorithm that can answer the problem correctly. A problem is said to be a decidable problem if there exist a corresponding Turing Machine which halts on every input with an answer yes or no.

Undecidabimerability

The problem for which we cant construct an algorithm that can answer the problem correctly in finite time are termed as Undecidabimerability problem. These problems may be partially partially decidable problems but their will be never decidable that is there will ~~be~~ always be a condition that will lead the turing machine into an infinite loop ~~whithout~~ without providing an answer at all.

Eg:- 1. Whether a CFG generates all the strings or not.

2. Ambiguity of CFG.

Q3 Recursive Enumerable Language

RE language or Type-0 Language are generated by type-0 grammer. An RE language can be accepted or recognized by Turing Machine which means it will enter into final state for the strings are may be or may not enter into ~~a~~ rejecting state strings of language. It means Turing Machine can loop the strings which are not part of the language.

Recursive Language

A recursive language can be decided by machine which means it will enter finite state for the strings of language and real state for the strings which are not part of language

Eg: $L = \{a^n b^n c^n \mid n \geq 1\}$, is recursive f because construct a turing machine which move to state if the strings that Turing Machine will also halt in this case. RE language are also as Turing Decidable Language.

Q4. Rice Theorem states that any non-trival semantic property of a language which is recognized by a Turing Machine is Undecidable. A property, P, is the language of all Turing Machine that satisfy that property.

Formal Defination

If P is a non-trival property, and the language holding the property, L_p, is recognised by Turing Machine M, then $L_p = \{\langle M \rangle \mid L(M) \in P\}$ is Undecidable.

Properties

1. Property 1: There exists Turing Machine, M1 and M2 that recognize the same language, i.e, either $(\langle M1 \rangle, \langle M2 \rangle \in L)$ or $(\langle M1 \rangle, \langle M2 \rangle \notin L)$
2. Property 2: There exists Turing Machine M1 and M2 where M1 recognizes the language while M2 does not, i.e, $\langle M1 \rangle \in L$ and $(\langle M2 \rangle \notin L$.

Camlin Page
Date / /

Proof

Suppose, a property P is non-trivial and $\varphi \notin P$.
Since, P is non-trivial, at least one language ~~st~~ satisfies P, i.e., $L(M_o) \in P$; $\exists$ Turing Machine M_o.
Let, w be an input in a particular instant and N is a Turing Machine which follows - on input x :
- Run M on w
- If M does not accept, then do not accept x.
- If M accepts w then run M_o on X. If M_o accepts x, then accept X.

A function that maps an instance ATM = {<M,w> | M accepts input w} to a N such that :
- If M accepts w and N accepts the same language as M_o. Then $L(M) = L(M_o) \in P$.
- If M does not accept w and N accepts φ, Then $L(N) = \varphi \notin P$.

Since, A_{TM} is undecidable and it can reduce to L_p, L_p is also ~~undeci~~ undecidable.

Q5 The Post Correspondence Theorem ~~is~~ was introduced by Emil Post in year 1946. Its a type of undecidable decision problem.
Let, $A = w_1, w_2, \ldots, w_k$ and $B = x_1, x_2, x_3, \ldots, x_k$ be strings over some alphabet Σ.
The PCP is to find the correspondence sequence of integers $i_1, i_2, \ldots, i_m$ for $m \geq 1$ such that :

$$w_{i_1}, w_{i_2}, w_{i_3}, \ldots, w_{i_m} = x_{i_1}, x_{i_2}, x_{i_3}, \ldots, x_{i_m}.$$

The sequence, '$i_1, i_2, \ldots, i_m$' is considered as the solution for the PCP instance. Each PCP instance is constituted by some set of values for A and B.

Camlin Page

Date / /

If we consider the PCP as a generic class of all such instances, then it is unsolvable. Furthermore, there exist no generic algorithm that can find a solution for any such PCP instance: hence, it is also an undecidable problem. However, for a few values of A and B, it might have a solution.

Eg: Find whether the list M = (abb, aa, aaa) & N = (bba, aaa, aa) have a PCS?

->

	x_1	x_2	x_3
M	abb	aa	aaa
N	bba	aaa	aa

Here,

$x_2 x_1 x_3$ = 'aaabbaaa'

and, $y_2 y_1 y_3$ = 'aaabbaaa'

Hence, we can see that

$x_2 x_1 x_3 = y_2 y_1 y_3$

Hence, the solution is $i = 2, J = 1$ & $K = 3$.

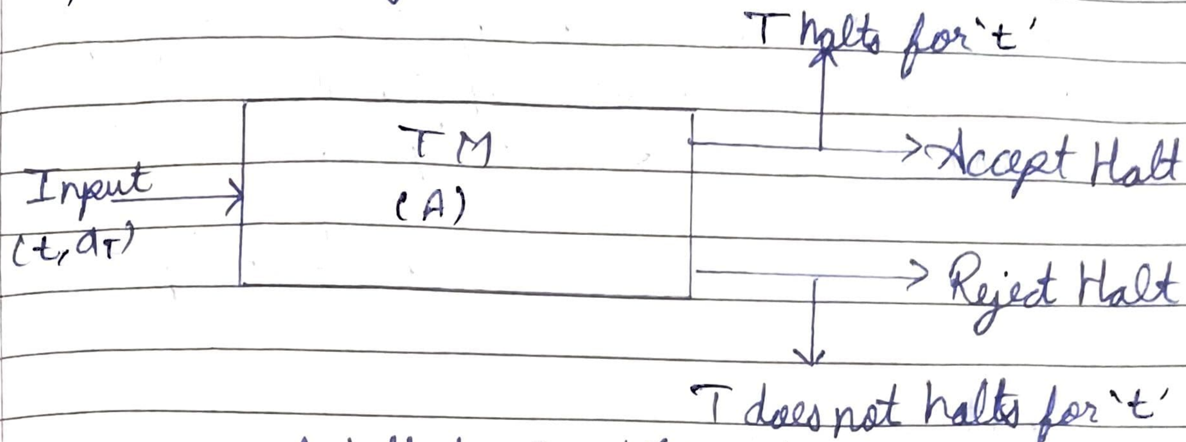
③

Q1. To Prove: The Halting Problem Is Unsolvable.

Proof: Let us prove the halting problem is unsolvable by contradictions.

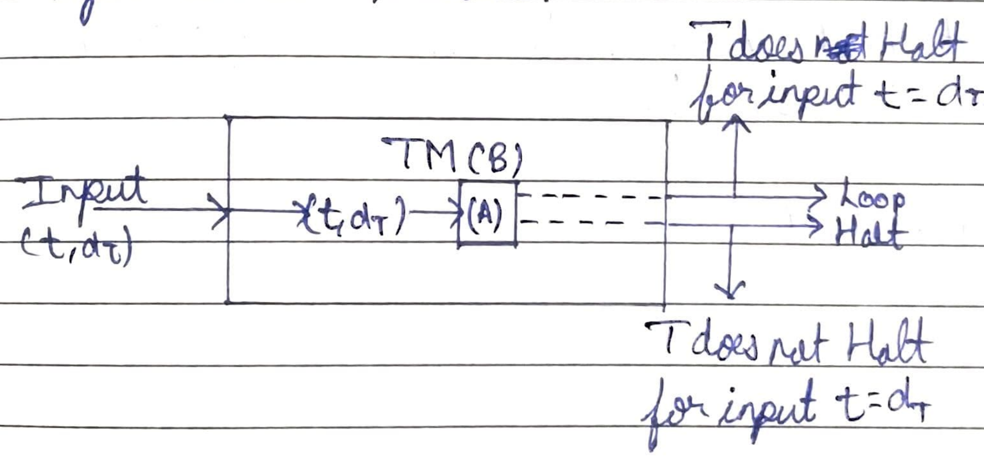
Let us assume that there exist a TM (A) which decides whether or not any computation by any TM (T) will ever halt given that the description d_T (SFM) of T, and the input tape t of T. Then for every input (t, d_T) to A, if T halts, then A reaches an 'accept halt' state;

else, A reaches a 'reject halt' state.



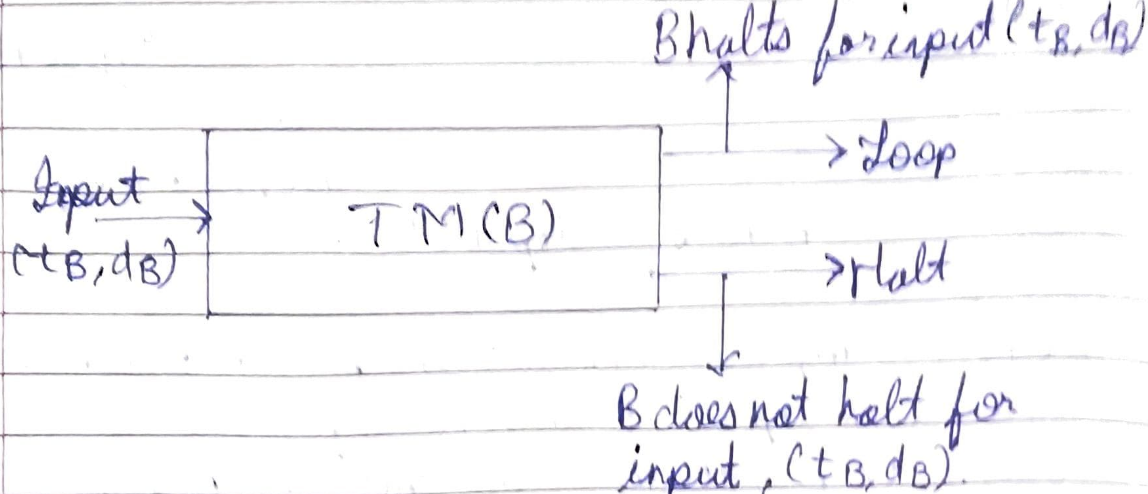
A halts for input (t, d_T)

We now attempt to construct another TM(B) which takes (t, d_T) as the input; it functions as follows:
- First, it copies the input and duplicates the same onto its tape. Then it takes this duplicated information tape as the input to A. Whenever A reaches the 'accept halt' state, B loops forever, and, ~~wheneer~~ whenever A reaches the 'reject halt' state, B halts.



Considering the orignal behavior of A, we find that B acts as follows:
It ~~fails~~ loops if T halts for input t, and halts if T does not halt for the input t. Thus, the working of TM(B) is exactly opposite of that of TM(T).

~~Now~~ Now, since B itself is a TM, let us set T = B. In this case, B halts for the input if and only if B doesnot halt for the same input, and loops forever if and only if B halts for the input.



Well, this is a ~~const~~ contradiction.
Hence, we conclude that the machine A, which can decide whether or not any other TM will ever halt cannot exist. Therefore, we conclude that the halting problem is unsolvable.

Hence, Proved.