Name :- Sumit Kumar Yadav
Roll No. :- 18CS30042

Page No.
Date

Sol^n :- 1 :- Consider P : property on P
s.t : $P : C \to \{T, \perp\}$
where, C is set of partially rec fn.
now, $P = \{ i \mid f_i \in C \text{ \& } P(i) \}$
where i is the index of partially rec. fn.
(encoding of M (TM) that computes f)
$x_1, x_2 \in N$, $P(x_1) = T$
$P(x_2) = \perp$

contradiction :-
P is decidable
now $\exists$ a total computable fn σ that computes P
now, define σ_1 total computable fn.
$$\sigma_1(y) = \begin{cases} x_2 & P(y) = T \\ x_1 & \text{otherwise} \end{cases}$$

Since P is decidable, σ_1 is total computable
by fixed point then.
$\exists x_0$, $f_{x_0}(y) = f_{\sigma(x_0)}(y)$.

$\therefore$ If $P(x_0) = T$,
f_{x_0} satisfies P
& $f_{\sigma(x_0)}$ satisfies P
& f_{x_2} satisfies P
$\therefore P(x_2) = T$
& $P(x_2) = \perp$.

Sol^n 2:- consider that

writes1 = { M | M is a TM with alphabet {0, 1} |
M writes 1 at some point on its tape }

Now, we have to prove that
reduction HP to write1

$\therefore$ $HP \leq_m WRITE1$

$(M, x) \mapsto N$

$M, x \longrightarrow HP$

$N \longrightarrow WRITE1$

Now, consider N to be as

→ description of M & x is in N's finite control.

→ Also reduction algo,

→ N replaces 1 if it occurs in M's
→ ~~N replaces~~ input alphabet by \$ (not in alphabet)
→ N replaces transition functions having 1
with \$

Now, N → input y over $\{0, 1\}^*$
→ erase y, replace with blank
→ simulate M on x
→ if M halts on x
then N writes 1 on its tape & halts

else
N continues simulating M on x.

$\therefore$ we can say that M halts on $x \Leftrightarrow$ N writes1
at some place

$\Rightarrow HP \leq_m WRITE1$ is valid reduction

$\Rightarrow$ WRITE1 is undecidable as HP is undecidable.

Sol 3:-(a)

let $\neg(a)$: P is not a monotonic

$\therefore$ TP is not re by directly application of rice's theorem

So, TP is re $\neq$ (a)

(b) $\neg(b)$: A is a finite language Sp and there's no finite subset of A is Sp.

→ assume Lp was r.e

→ let M, accepts L

Now construct a reduction : $\langle M, W \rangle \Rightarrow M'$ st x' accepts L if w is not in L(M) and accepts ~~the~~ else accepts finite subset of L.

Now, on input M, M' with input x will simulate M on W for $|x|$ moves & if it fails to accept w after $|x|$ moves. M' accepts x.

Now, let say M accepts w after k moves,

$L(M') = \{x \mid x \in L \text{ and } |x| < k\} \subseteq L$

⇒ if M does not accept w. L(M') bring a finite subset of L not in Sp using $\neg(b)$

now, if T_p in re then membership problem is decidable hence T_p is not in re.

(c) Finite language in Sp can be represented in binary encoding

now ~~on~~ as T_p is r.e.

→ Let an ennumerable machine N enumerates T_p, we will make $M^{(i)}$ to enumerate binary encodings of finite language (i) is S_p.

→ Then we will enumerate the binary pairs (i, j) using K, i is a binary encoding of finite language.

→ now, if N has printed $M^{(i)}$ in j steps, K prints (i) followed by a delimeter symbol to differentiate from previous and next step encodings.

→ So, K enumerates all binary codes for finite set in S_p.

$\therefore$ T_p in r.e

So, we ~~a~~ can conclude that (a), (b) and (c) imply T_p is r.e.

Sol^n:- 4:- Let us consider A and B are the two instance of given problem.

ie, $A = \{w_1, \_\_\_\_, w_n\}$

& $B = \{x_1, \_\_\_\_, x_n\}$

such that

$|w_1| = |w_2| = \dots = |w_n| = |x_1| = |x_2| = \dots = |x_n| = 5$

$\therefore$ all strings are of length 5

$\therefore$ Solutions exists if there exist a sequence

$i_1, \dots i_k$ and $j_1, \dots j_k$ st

$w_{i_1} = x_{j_1}, \dots$ & $w_{i_k} = x_{j_k}$

also let C be variant of PCP as PCPV

$C = (A, B) \in PCPV \Leftrightarrow \{ \exists$ sequence $i_1, \dots i_m$

C is decidable using decides to decide PCPV

now, on input (A, B), C:

$\to$ checks if $\exists$ any index i_1 to i_k : $w_1 = x_1$

$\to$ if such index exists, solution to C is found

$\Rightarrow$ C halts for input (A, B)

$\to$ else c reject c halts.

Now, checking single index is sufficient as w_i & x_i are strings of length 5

$\therefore$ $w_{i_1}, w_{i_2}, \dots w_{i_m} = x_{i_1}, x_{i_2} \dots$

$\Rightarrow w_{i_1} = x_{i_1}$ & so on

$\therefore$ if it matches for multiple i_s, it must match for one of i_s.

$\therefore$ its sufficient to check on index $\Rightarrow$ decidable PCP

$\therefore$ The variant given is decidable.

Soln:- 5:- $S = \{G : G$ is a CFG and $L(G) = L(G)^R\}$

to prove : S is undecidable.

$\because$ we already prove that $\neg VALCOMPS(M, x)$ is a CFG.

let $L(G) = \neg VALCOMPS(M, x)$

now, we make the following reduction :

$$\neg HP \leq_m \neg VALCOMPS(M, x)$$

consider the following cases:-

case 1:- M does not halt on x

$\Rightarrow VALCOMPS(M, x) = \{\}$

$\Rightarrow \neg VALCOMPS(M, x) = \Sigma^*$

$\therefore L(G) = \Sigma^*$ so $L(G)^R = \Sigma^*$

$\Rightarrow L(G) = L(G)^R$

case 2:- M halts on x

$\Rightarrow VALCOMPS(M, x) = \Sigma^n$

$\Rightarrow \neg VALCOMPS(M, x) \neq \Sigma^*$

so, for some $j \in VALCOMPS(M, x)$ and $j = j^R$, the start symbol will be at the end of j^R resulting in an invalid configuration.

Here, ~~$j^R \notin L$~~ $j^R \notin L(G)$

$\Rightarrow L(G) \neq L(G)^R$

so, we can conclude that

S is undecidable.

Sol^n:- 6:-

Consider $VALCOMPS^t_{M,x}$ be the set of valid computation histories of M on input x. ending in accepting ~~submission~~ configuration

Now, If the set accepted by M is finite, the set of valid computations of M is finite & hence a CFL. assume the set of accepted by M is infinite & the set L of valid computation is a CFL.

Since, M accepts an infinite set, there exists a valid computation

$$\omega_1 \# \omega_2^R \# \omega_3 \# ---$$

where the ω_i's are ID's & $|\omega_2|$ is ~~greather~~ greater than the constant n in Ogden's lemma.

Now, mark the symbols of ω_2 as distinguished Then we can pump ω_2 without pumping both ω_1 & ω_2 thus getting an invalid computation that must be in L.

Thus, we conclude that the valid computations do not form a CFL.

Soln:- 7:- Let us consider a fn. f be a partial recursive function as follows:-

→ let M be i/p for f then output will be N with M hardcoded it.

→ Let us define $x \leq$ as lexicographic ordering of strings.

→ now. non i/P y:-

→ simulate M on every i/P $x \leq y$

→ accept y is M halts on all x.

→ If M is total, $L(M) = \Sigma^* \Rightarrow L(N)$ is recursive.

→ If M is not total, let z be the smallest string such that M doesn't halt on z.

$\therefore \quad L(N) = \{ z' \mid z' < z \}$

→ As L(N) is finite ⇒ L(N) is recursive.

$\therefore \forall M, \; L(f(M))$ is recursive.

→ Also, f is total recursive function.

∴ we can iterate over $\{0,1\}^*$ & find f(x) for every x we get during iteration & list f(x) if x is valid encoding of a T.M.

→ Let this enumerating machine be called E.

$\therefore L(E) = \{ f(i) \mid i \in$ binary encoding of a TM $\}$

∴ L(E) is the required r.e list of TMs that accepts recursive sets.

→ Also, as L(E) contains list of all possible TMs accepting recursive sets, ∃ M accepting x ∀ recursive set.

⇒ ∴ L(E) is the required r.e. list.

Sol^n:- 8:-

let us consider $\varphi = Q_1 x_1 Q_2 x_2 \cdots Q_l x_l \psi$ where $Q \in \{\exists, \forall\}$ & ψ is quantifier free.

also, let $\varphi_i = Q_{i+1} x_{i+1} Q_{i+2} x_{i+2} \cdots Q_l x_l \psi$ for $i = 0$ to l

$\Rightarrow \varphi_0 = \varphi$ & $\varphi_l = \psi$.

$\therefore$ The formula φ_i has i free variables. for $a_1, \ldots, a_i \in N$ we can say that $\varphi_i [a_1, \ldots, a_i]$ to be the sentence obtained by substituting the constants $a_1, \ldots, a_i$ for the variables $x_1, \ldots, x_i$ in φ_i.

→ The algorithm builds finite automata A_i which recognize the collection of strings representing i-tuples of numbers that make φ_i true.

→ It first builds A_l directly. Then for each $i = l, \ldots, 1$ It uses A_i to build A_{i-1}.

Now, once the algorithm has A_0, it tests whether A_0 accepts ϵ, in which case it show that φ is true as A_0 accepts any inputs iff φ_0 is true. Therefore if A_0 accepts ϵ, φ is true and A_0 accepts otherwise it rejects.