Name : Sindhuja Yerramalla

UUID : U00839259

mailid : syrrmlla@memphis.edu.

Foundations of Computing HW5

References:- Referred to many websites, Text book, lecturenotes Chegg & worked with few classmates.
[worked with vamshidhar reddy]

Problem 1: Problem 9, sec 10.4 from Linz

Sol :- Given that S_1, S_2 are countable sets.

let $S_1 \rightarrow a_1, a_2, a_3, \ldots$ and $S_2 \rightarrow b_1, b_2, b_3, \ldots$

To enumerate $S_1 \cup S_2$ we can map $S_1 \cup S_2$ with N that is there is a bijection from $N \longrightarrow S_1 \cup S_2$ such that if n in N is odd it maps to $(n+1)/2$ element of S_1. If n in N is even then it maps to $n/2$ element of S_2.

$\therefore$ we can enumerate $S_1 \cup S_2 \longrightarrow a_1, b_1, a_2, b_2, a_3, b_3, \ldots$

$\Rightarrow$ $S_1 \cup S_2$ is countable.

Now consider the set $S_1 \times S_2$. $S_1 \times S_2$ is of form (s_n, s_m)

where $s_n \rightarrow$ string enumeration of S_1
$s_m \rightarrow$ string enumeration of S_2.

we can arrange these in the way we have formed the enumeration for p/q quotient. This give us order pairs of all elements in both S_1 & S_2

$$
\begin{array}{ccc}
(1,1) \rightarrow (1,2) & & (1,3) \\
 & & \downarrow \\
(2,1) & (2,2) & (2,3) \\
(3,1) & (3,2) & (3,3) \\
\vdots & \vdots & \vdots
\end{array}
$$

$\rightarrow$ Here we are following the order by following the arrows.

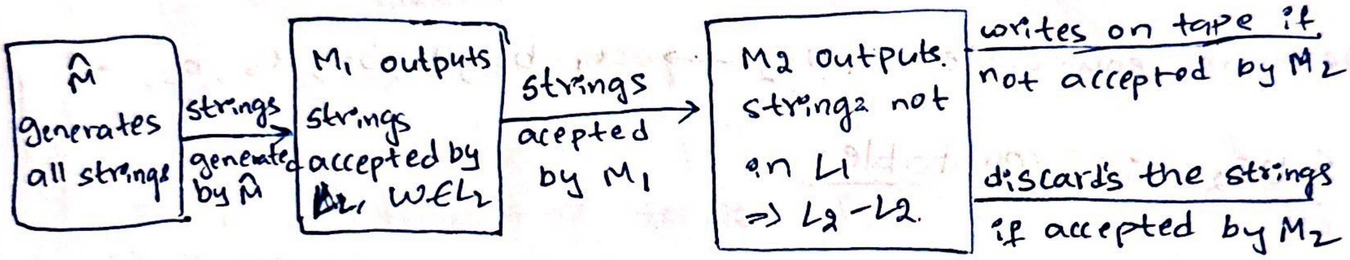
$\rightarrow$ we can enumerate $S_1 \times S_2$ with this order.

$\therefore$ $S_1 \times S_2$ is also countable.

Problem 2 : Problem 12, Sec 11.1 from linz

Sol :- Given L_1 is recursive & L_2 is recursively enumerable

We know how to generate & enumerate a recursively enumerable language. we take a machine $\hat{M}$ which accepts the string w where $w \in L$, To avoid infinite loop we move 1step in each string for every string given. by $\hat{M}$, so this will generate all the strings that are accepted by M & goes into infinite loop if not accepted.



In this way we can enumerates all strings of $L_2 - L_1$.

hence we can say that the language is recursively enumerable

Problem 3 :- Problem 14, sec 11.1 from linz.

Yes, if L is recursive then L^+ is also recursive.

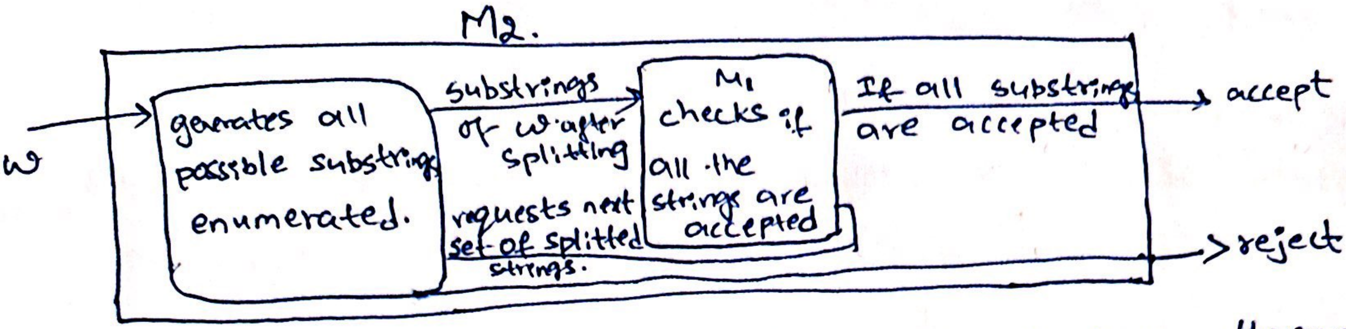
By dividing the strings in L^+ into substrings & check if each substring is in L we can prove the statement given.

If all the substrings are in L then the string gives is in L^+.

This division should check all possible substrings of the given string. As the length of the input string is not infinite we can enumerate. all possible splits of the string.

Let M_1 be machine that accepts L or rejects if not in L.

M_2 be the machine that decides w on L^+



M_2 accepts if all the possible splits are run through M_1 & if any split accepts all the substrings of w.

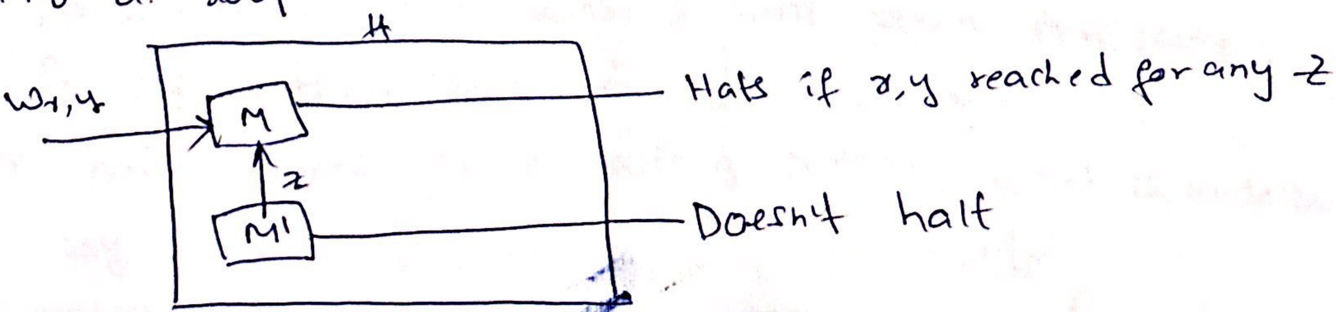
If the machine halts after checking all the possible splits then M_2 rejects the string w

<u>Problem 4:</u> Problem 10, Sec 12.1 from linz

Given M & x, y are instantaneous description of M.

We need to show that $x \vdash^*_M y$ is undecidable by reducing this problem to a halting problem.

Let H be a turing machine that take x, y (instantaneous descriptions of M) & decides if $x \vdash^*_M y$. Let M' be a turing machine that generates all the possible strings of Machine M. For a string z if M reaches both the descriptions x, y then the turing machine H halts else it gets into a loop.. so we can say that $x \vdash^*_M y$ is undecidabl



Problem 5 : Problem 4, Sec 12.2 from linz.

Sol Given M_1, M_2. are arbitrary turing machines

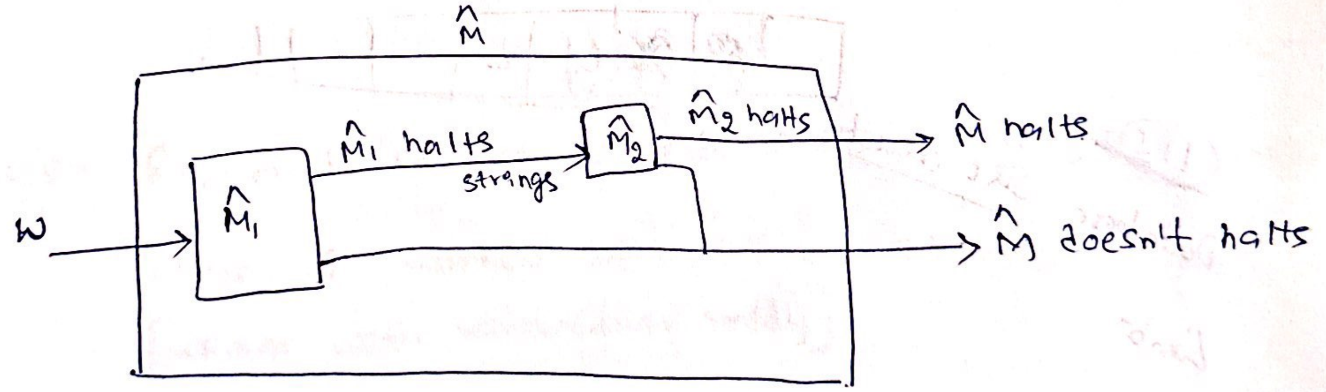
We need to prove that $L(M_1) \subseteq L(M_2)$ is undecidable

A Language L is decidable if a turing machine M exist such that the turing Machine halts to a finite state when the input is of length w, where $w \in L$.

We need to prove every string $w \in L(M_1)$ is accepted by M_2.

let $\hat{M}_1$ be a turing machine that halts on $L(M_1)$. & $\hat{M}_2$ be a turing machine that halts on $L(M_2)$.

Let $\hat{M}$ be a turing machine that halts when given an arbitrary string w. $\hat{M}$ is designed in such a way that the given string w is given to $\hat{M}_1$ first & if it halts then given to $\hat{M}_2$. If $\hat{M}_2$ also halts then machine $\hat{M}$ halts



Now we generate string w such that $w \in L(M_1)$ & run $\hat{M}$ on w. If $(\hat{M}, w)$ halts, $\hat{M}$ will reach final state.
So. If $\hat{M}_1$, $\hat{M}_2$ does not halt $\hat{M}$ also not halts.
Since we have reduced to a halting problem which is undecidable we can say $L(M_1) \subseteq L(M_2)$ is undecidable //