Automata theory

÷ Theory Assignment

Section - 2

① To prove :- If L is a CFL over one symbol alphabet then it is regular.

proof :- Let say, $\Sigma = \{a\}$ (because L was over one symbol)
↑ input alphabet.

By pumping lemma for CFL's. Let, $w = a^{2p} \in L$. divide it into $uvxyz$ such that $|vxy| \leq p$, $|vy| \geq 1$, $uv^ixy^iz \in L$. $\forall i \geq 0$

whatever length selected for v & y it belongs to L. as L is CFL so pumping lemma for CFL's satisfied.

$$uv^ixy^iz \Rightarrow a^{2p+(|v|+|y|)*i} \quad -(1)$$

↑ length of string selected for v ; ↑ for y.

Now in pumping lemma for regulars we must divide string into xyz such that $|xy| \leq p$, $|y| \geq 1$, $\forall i \geq 0$, $xy^iz \in L$.

from ①, if we select length of y same as |v| + |y| (in pumping lemma for CFL's).

then: $xy^iz = a^{2p + (|v| + |y|) * i} \in L$

$\Rightarrow$ L is regular because pumping lemma for regulars satisfied.

Hence proved.

there must

(2) for infinite regular language (A) there must be repetition of particular part of a language.
So ∃ atleast one state q_1 that must be visited infinite times. ie a loop exist.
so for 1 language (A_1) consider visiting that state q_1 even number of times and another language (A_2) where it visit the state q_1 odd number of times. Remaining it is completely same.

A_1 :- a particular state q_1 should be visited a given number of times

A_2 :- a particular state q_1 should be visited odd number of times

Here q_1 is any such state which can be visited any number of times in DFA generated by regular language which was given initially

Thus $A_1 \cup A_2 = A$ (given infinite regular language)
$A_1 \cap A_2 = \emptyset$. Then proved.

(4) =. We know that there can be 0's & 1's in a string. So the first symbol should be replaced with the last symbol and last symbol with the first. and so on.

Basic idea :- 0 1 0 1 0 1 0 ... 1 0 1 → This is counter. reverse of the string.

* first we take the pointer to right until we encounter a blank then move left. Replace the blank with rightmost ~~char~~ symbol and the rightmost symbol with X.

  let, initially B 0 1 1 0 B ... → B 0 1 1 X 0 B ...

* Then we search first element left of X & then we ~~push~~ ~~it to right of~~ 0 replace. B with it and that with X

  → B 0 1 X X 0 1 B

  ↓

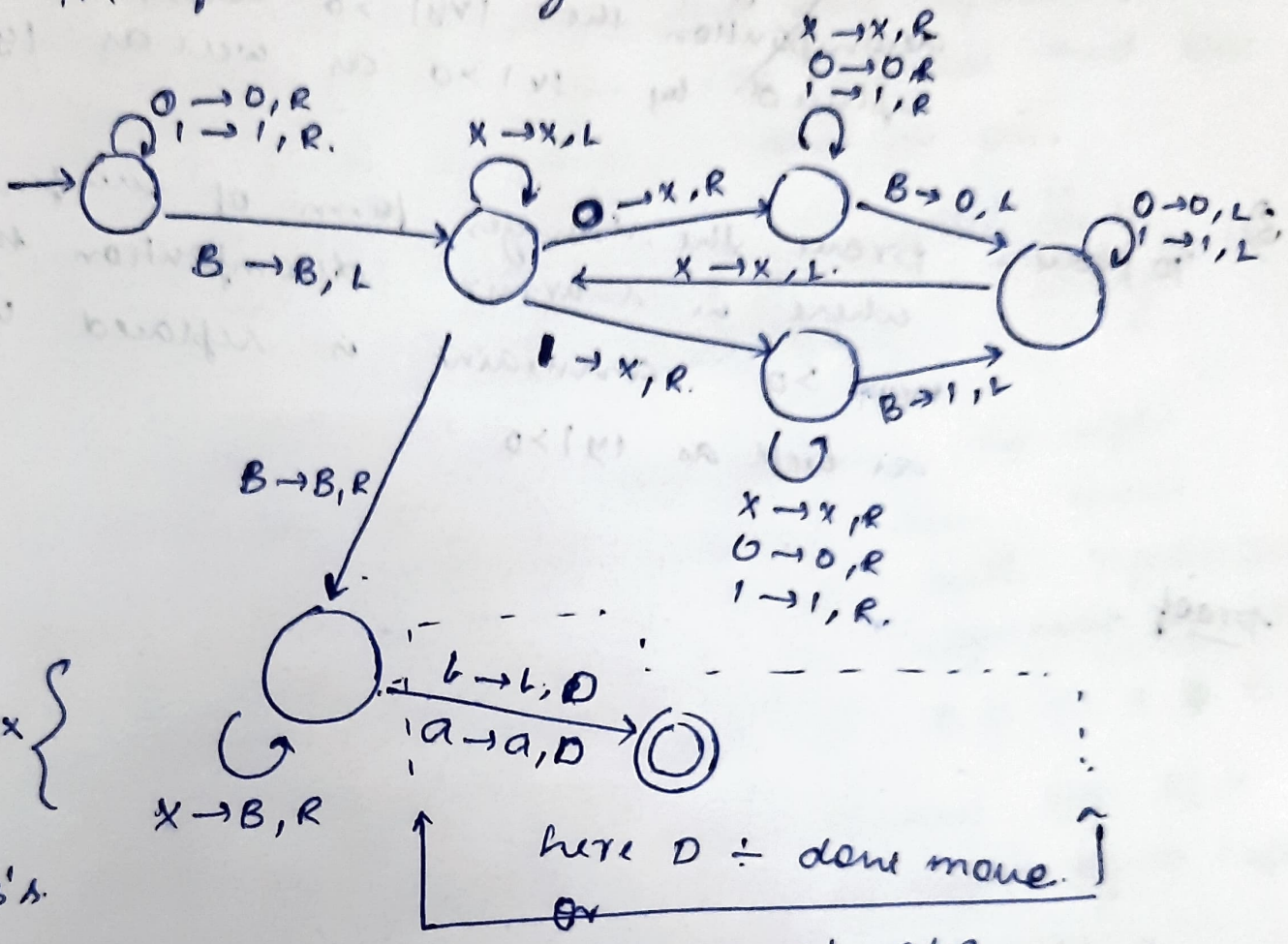
we get → B X X X X 0 1 1 0 B.

Now we make X also B. so

we get ... → B 1 1 1 0 B - - -

we output this

TM for above logic :-



(6) $L = \{a^n b^n c^n \mid n \in \mathbb{N}\}$.

✓ Let L be context free. so it satisfies stronger form of pumping lemma.

✓ Let p be pumping length.

consider a string $s = a^p b^p c^p$

from stronger form of pumping lemma $\Rightarrow$ s must be decomposed into $uvxyz \in L$ such that $|vxy| \leq p$ & $|vy| \geq 0$

(length of vxy)

$uv^i xy^i z \in L \quad \forall \; i > 0$

Case-1

vxy can be 0^k, or 1^k or 2^k such that $k \leq p$.

After pumping one string can be of form. such that number of 0's can be greather than 1's and 2's or number of 1's can be greater than number of 0's & 2's (or) number of 2's can be greater than other two.

Case - 2

vxy can be $0^m 1^n$ or $1^m 2^n$ such that $m + n \leq p$.

After pumping. number of 0's & 1's are greater than number of 2's.

or

number of 1's & 2's are greater than number of 0's.

Thus from case ① ② ..

number of 1's 0's 2's can't be equal by pumping lemma.

$\Rightarrow$ L is not CFL

Hence proved by contradiction that L is not context free. *.

Is it recursive?

Let $L_1 = \{a^i b^i c^j \mid i, j \geq 1\}$

L_1 is context free because we can generate L_1 with following CFG.

$S \rightarrow S_1 S_2$

~~$S_1 \rightarrow 0S_1 1 \mid 01$~~ $S_1 \rightarrow aS_1b \mid ab$.

~~$S_2 \rightarrow$~~ $S_1 \rightarrow aS_1b \mid ab$.

$S_2 \rightarrow cS_2 \mid \varepsilon$.

i.e. L_1 is context free and recursive. language.

NOTE:- every context free is recursive. language.

[we may think at some string may loop forever. But that is false because we can convert this CFG into CNF or GNF where each derivation of string(w) can be derived in $2|w|-1$ steps only. so accepted / rejected]

Thus L_1 is recursive language.

Ex: $L_2 = \{a^i b^j c^j \mid i, j \geq 1\}$

Ily we can construct a CFG for this

$$S \to S_1 S_2$$
$$S_1 \to aS_1 \mid \varepsilon$$
$$S_2 \to bS_2 c \mid bc$$

$\Rightarrow$ L_2 is recursive.

Intersection of two recursive language is also recursive.

$\Rightarrow$ $L_1 \cap L_2 = a^i b^i c^i \mid i \geq 1$ is recursive

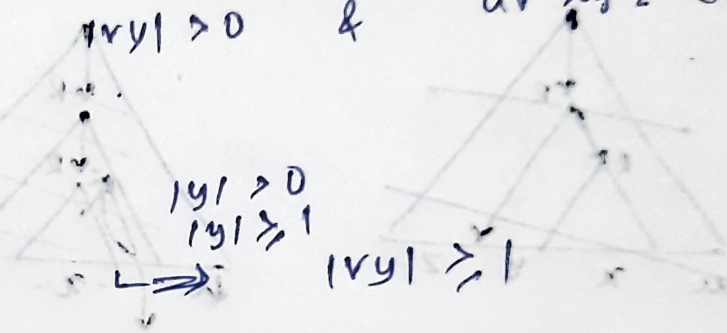
$L \Rightarrow$ recursive ✓

we know recursive languages $\subseteq$ recursively enumerable language

$\Rightarrow$ $L \Rightarrow$ recursively enumerable language ✓

3)

Actual pumping lemma = Decompose s into $uvxyz \in L$ such that $|vxy| \leq p$ & $|vy| > 0$ & $uv^i xy^i z \in L \ \forall i \geq 0$



But $|v| > 0$ $\quad |y| > 0$

$\Rightarrow |v| \geq 1$ $\quad |y| \geq 1$

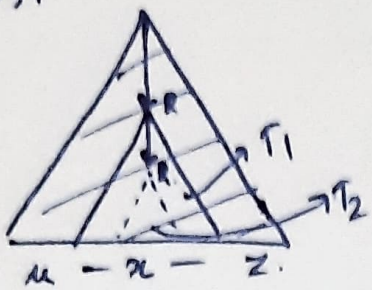
$\Rightarrow |vy| \geq 1$

$\Rightarrow |vy| > 0$.

Thus. pumping lemma. is correct for. $|v| > 0$; $|y| > 0$ and

why $|vy| > 0$?

let $|vy| = 0$



we know

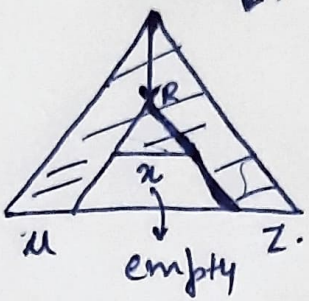
$uv^i xy^i z \in L$

$\forall i \geq 0$

so. for $i = 0$

we replace T_1 with T_2.

$\Rightarrow$



contradiction

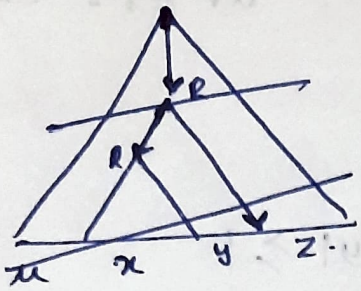
$\Rightarrow$ for smaller number of nodes also $S = uxz$ is same.

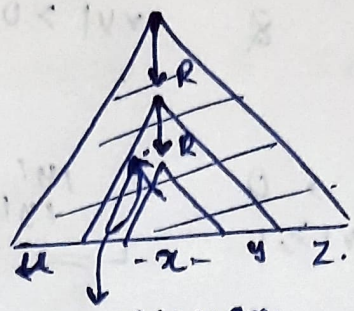
$\Rightarrow$ we said we only consider smallest parse tree. when two or more parse tree exist for string

Thus. $|vy| \neq 0$

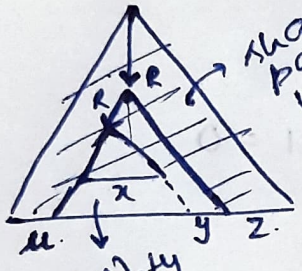
$\Rightarrow$ $|vy| > 0$

114 Let us do for $|v| = 0$





unnecessary paths of variables ending with ε.



shaded part has nodes.

empty

a parse tree with less number of nodes we got for string of same length. uxyz

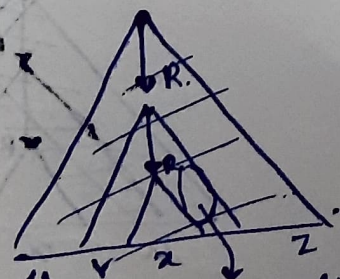
Then consider for the smallest parse tree only

⇒ this dont exist

So, contradicts

⇒ $|v| \neq 0$ ⇒ $|v| > 0$ ✗

Now let us do for $|y| = 0$

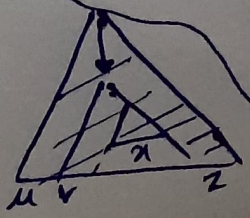


unnecessary paths of variables ending with ε.

⇓

a parse tree with less no. of nodes for a string uvxz found.

⇒ this dont exist



so, contradicts

⇒ $|y| \neq 0$ $|y| > 0$ ✗

Thus proved $|v| > 0$, $|y| > 0$ for pumping lemma

(6)

The set of ~~all~~ halting turing machines is recursively enumerable but not recursive since it may not halts on some input of NO instance.

for not halting turing machine a string dont halts i.e runs indefinitely

for NO instance.

$\Rightarrow$ Not recursive

* It is not possible for a language to be ~~recursive~~ ~~enumerable~~ but not recursively enumerable because in there are set of string which halt but in recursively enumerable for recursive language there are also strings which dont halt.

$\Rightarrow$ Recursive language $\subseteq$ recursively enumerable language

$\Rightarrow$ language which is recursive but not recursively enumerable is not possible.

~~DXO~~

(7)

Let $\Sigma = \{0, 1\}$.

Theorem :- Set of all strings is countably infinite but set of all languages is uncountably infinite.

Proof :- Let write the set of strings in ~~lexicographical~~ increasing order of length i.e $|s_1| < |s_2|$ then s_2 comes after s_1 if $|s_1| = |s_2|$ then arrange in lexicographical order.

Strings = $\{\varepsilon, 0, 1, 00, 01, 10, 11, \ldots\}$.

Now add 1 to the front :-

Strings = $\{\varepsilon, 0, 1, 00, 01, 10, 11 \ldots$
$\{1, 10, 11, 100, 101, 110, \ldots$

mapped to ↓ ↓ ↓ ↓ ↓ ↓

1 2 3 4 5 6 . .

In fact 1, 10, 11, ... are binary representations of 1, 2, 3, ...

✓ as there are no 2 strings which are same.

then binary representation arent same.

✓ As each string have specific binary representation.

$\Rightarrow$ one-one mapping ∞

All natural numbers have unique binary representation. $\Rightarrow$ onto. ∞

Thus there bijective mapping to $\mathbb{N}$ (natural numbers)

$\Rightarrow$ Set of all string in Σ are countably infinite.

Set of all languages ≡ power set of set of all strings.

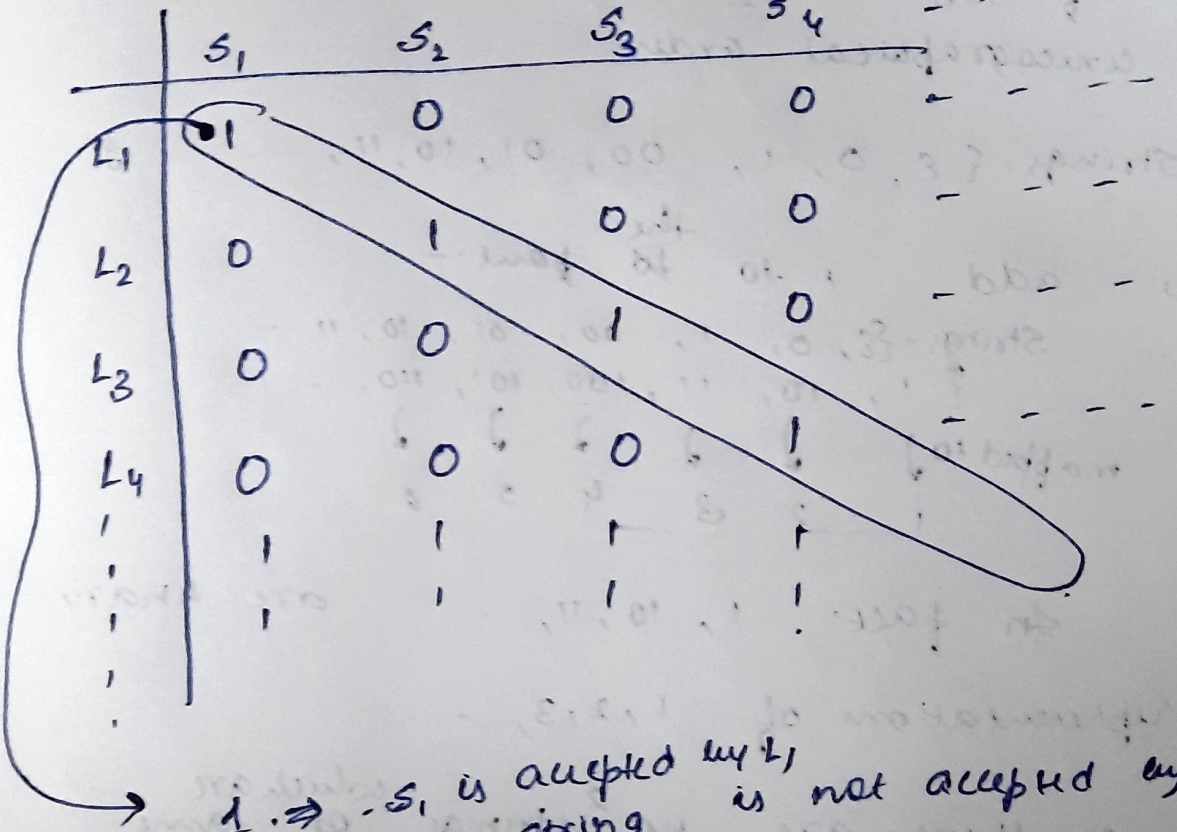
P(set of all strings)

power set

countably infinite.

Set of Languages = $\{L_1, L_2, \ldots$

Set of strings = $\{S_1, S_2, \ldots$



1 ⇒ S_1 is accepted by L_1

0 ⇒ that string is not accepted by that particular language.

by this we by now, we have countably infinite language because of countably infinite string.

Now join its diagonal, that is not there in any language.

~~By for any~~

$\exists$ for any such possible configuration of in the matrix shown. $\exists$ a diagonal combination here

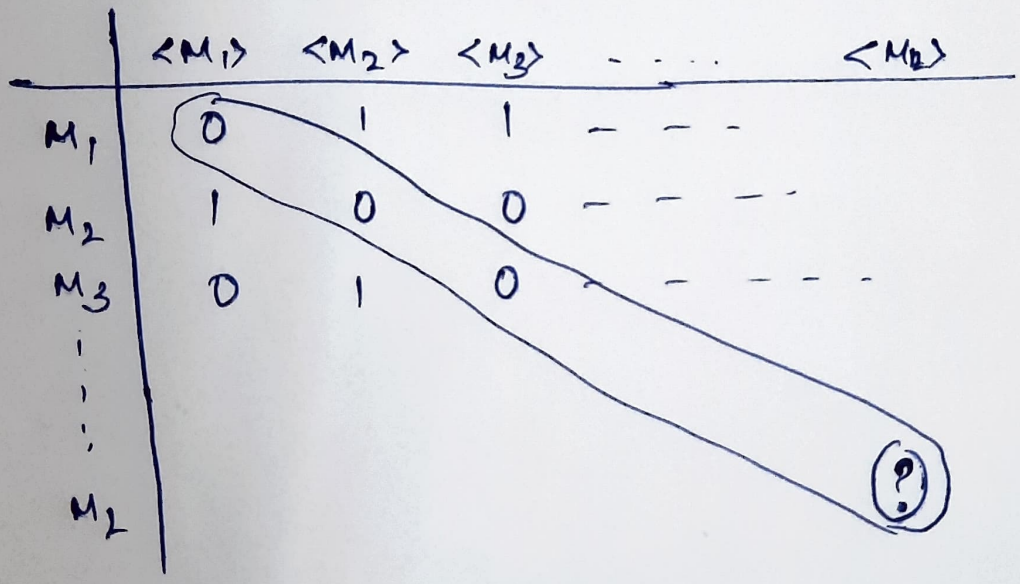
	s_1	s_2	s_3	- -
	1			
		1		
			1	
				- -

there is no bijective mapping ~~from~~ to N from ~~to~~ it.

[set of all language $\Rightarrow$ uncountably infinite.]

(8) Here again we will use diagonalisation technique.

Let $\{M_1, M_2, \ldots\}$ be set of Turing machine. Since turing machine can be described as a string, so we have countable number of turing machines.



here. the element at $(i,j) = 1 \Rightarrow M_i$ accepts $\langle M_j \rangle$
$= 0 \Rightarrow M_i$ dont accept $\langle M_j \rangle$

Now consider the language L to be decidable

so ∃ TM M_L which decides L.

M_L by definition supposed to have opposite value of what's along the diagonal

If we take opposite value of diagonal then we get a contradiction.

for X : what should the value be?

X = diagonal value for TM - A

Thus assumption wrong

⇒ L is not decidable

⇒ L is undecidable

Consider $\bar{L}$:

$\bar{L}$ is recursively enumerable

Turing machine for $\bar{L}$ can be made by.

⇒ simulating M with input <M>

TM for $\bar{L}$ — if M accepts then (accept)

— if else (reject)

Here TM recognizes $\bar{L}$ ⇒ $\bar{L}$ is recursively enumerable.

Now L is undecidable & $\bar{L}$ is RE.

L is partially decidable

⇒ not recursively enumerable

⇒ Not RE

(Hence proved)