13/11/2023

STEP-3

Removing Unit Productions

A productions of the form $A \to B$,

i) Find for each A all non-terminals B such that $A \overset{*}{\to} B$ —— (*)

ii) $G_1 = (V, T, P_1, S)$

Compute P_1 by including all non-unit productions of P.

For all A and B satisfying eqn (*) we add to P_1

$A \to y_1 | y_2 | \ldots | y_n$ where

$B \to y_1 | y_2 | \ldots | y_n$ is in P_1

Ex :- { $S \to Aa | B$

$B \to A | bb$

$A \to a | bc | B$ } Here 3 unit productions

$(S,B), (S,A), (B,A), (A,B)$

$G_1 = (V, T, P_1, S)$ # G_1 will not contain unit production in CNF

$P_1 = \{ S \to Aa | bb | a | bc$

$B \to bb | a | bc$

$A \to ba | bc | bb \}$

(S,B) me

Still $L(G) = L(G_1)$ and unit productions are removed

~~S →~~

STEP-4

Introduce a non-terminal C_1 with given production $C_1 \to a$ ($\because S \to Aa$ is not in CNF)

$\{ S \to C_1 \; || \; C_1 \to a \}$ CNF

$S \to bb$

C_2 with $C_2 \to b$

$S \to C_2 C_2 \; || \; C_2 \to b$ CNF

$S \to bc$

C_3 with production

$C_3 \to C$

$S \to C_2 C_3$

$C_3 \to c$

$C_2 \to b$

$B \to bb$ with production

$B \to C_2 C_2$

$C_2 \to b$

$B \to bc$ with production

$B \to C_2 C_3$

$C_2 \to b$

$C_3 \to c$

$A \to bc$ with production

$A \to C_2 C_3$

$C_2 \to b$

$C_3 \to b$

$A \to bb$ with production

$A \to C_2 C_2$

$C_2 \to b$

Now $G_2 = (V_2, T, P_2, S)$

$P_2 = \{ S \to AC_1 \mid C_2C_2 \mid a \mid C_2C_3$

$B \to C_2C_2 \mid a \mid C_2C_3$

$A \to a \mid C_2C_3 \mid C_2C_2$

$C_1 \to a$

$C_2 \to b$

$C_3 \to c \}$

$V_2 = \{ S, A, B, C_1, C_2, C_3 \}$

14/11/2023

(Q) Find the equivalent grammars in CNF for the given CFG.

$G = (V, T, P, S)$

$P = \{ S \to abAB,$

$A \to bAB \mid \epsilon,$

$B \to BAa \mid \epsilon,$

$B \to A \}$

I $P \to \{ S \to abAB,$

$A \to bAB \mid \epsilon,$

$B \to BAa \mid \epsilon,$

$B \to A \}$

$V = \{S, A, B\}$

II - $V_N = \{A, B\}$

$P_1 = \{S \to abAB \mid abB \mid abA \mid ab$

$B \to BAa \mid Aa \mid Bb \mid a$

$B \to A$

$A \to bAB \mid bA \mid bB \mid b \}$

III (B, A) $B \Rightarrow A$

$P_2 = \{S \to abAB \mid abB \mid abA \mid ab$

$B \to BAa \mid Aa \mid Ba \mid a \mid bAB \mid$

$bA \mid bB \mid b$

$A \to bAB \mid bA \mid bB \mid b \}$

Introduce

$C_1 \to a$

$C_2 \to b$

$S \to abAB$

$S \to C_1C_2AB$

$S \to C_3C_4$

$C_3 \to C_1C_2$

$C_4 \to AB$

$P3 = \{$ $S \to C_3C_4$ | $C_3 \to C_1C_2$ | $C_4 \to AB$ | $C_1 \to a$ | $C_2 \to b$ | $S \to C_3B$ | $S \to C_3A$ | $S \to C_1C_2$ | $B \to BAC_5C_1$ | $C_5 \to BA$ | $B \to AC_1$ | $B \to BC_1$ | $B \to a$ | $B \to C_2C_4$ | $B \to C_2A$ | $B \to C_2B$ | $B \to b$ | $A \to C_2C_4$ | $A \to C_2A$ | $A \to C_2B$ | $A \to b$ $\}$

GNF

A grammar is said to be in GNF if all productions are of the form $A \to a\alpha$ where $A \in V$; $a \in T$; $\alpha \in V^*$

~ $A \to BABC$; $A \to a$ $(\alpha = \epsilon)$

Convert the grammar in GNF:

ELIMINATION OF LEFT-RECURSIVE:

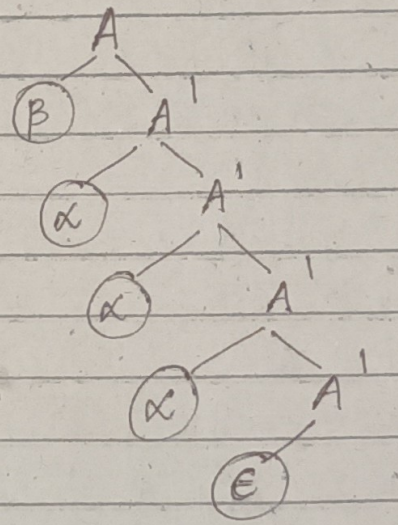
A grammar G is left recursive if it has a non-terminal A such that there is a derivation $A \overset{+}{\Rightarrow} A\alpha$ for $\alpha \in (V \cup T)^*$

DIRECT LEFT RECURSION

$A \to A\alpha \mid \beta$

$A \to \beta A'$

$A' \to \alpha A' \mid \epsilon$



Sure ET

① marks remove left recursion.

x- Is there any left recursion

$E \to E+T \mid T$

$T \to T*F \mid F$

$F \to (E) \mid id$

Non terminals are +, *, (,), id

terminals are E, T, F

Acc to $A \to A\alpha / \beta$

Here $\beta = T$; $A' \to E'$ & $A' \to +T$ $\alpha \to +T$

$$\begin{cases} E \to TE' \\ E \to +TE' / \epsilon \\ T \to FT' \\ T' \to *FT' / \epsilon \\ F \to (E) / id \end{cases}$$

THE PROCESS FOR CONVERSION TO GNF:

$$\begin{cases} S \to A \\ A \to aBa / a \\ B \to bAb / b \end{cases}$$

1 & 2 same

From step-3

$$P_1 \begin{cases} S \to aBa / a \\ A \to aBa / a \\ B \to bAb / b \end{cases}$$

$$P_2 = \begin{cases} S \to C_1BC_1 / a \\ A \to C_1BC_1 / a \\ B \to C_2AC_2 / b \\ C_1 \to a \\ C_2 \to b \end{cases}$$

$$P_3 = \begin{cases} C_3 \to C_1B \\ C_4 \to C_2A \\ S \to C_3C_1 / a \\ A \to C_3C_1 / a \\ B \to C_4C_2 / b \\ C_1 \to a \\ C_2 \to b \end{cases}$$

GNF: Rename the non-terminals of G as $V_1, V_2, V_3 \ldots\ldots$

Represent all productions as $A_i \to A_j \alpha$ such that $i < j$,

$\forall i = 1, 2, \ldots\ldots m$ where $j = 2, 3, \ldots\ldots m$

$V = \{S, A, B, C_1, C_2, C_3, C_4\}$ # set of non-terminal; the rename as $S = V_1$; $A = V_2$; $B = V_3$; $C_1 = V_4$; $V_2 = V_5$; $C_3 = V_6$; $C_4 = V_7$

$P_4 = \{$

$V_1 \to V_6 V_4$

$V_1 \to a$

$V_2 \to V_6 V_4$

$V_2 \to a$

$V_3 \to V_7 V_5$

$V_3 \to b$

$V_4 \to a$

$V_5 \to b$

$V_6 \to V_4 V_3$ ← will create problem

$V_7 \to V_5 V_2$ $\}$ ← as 6 is < than 6

$V_6 \to a V_3$

15/11/2023

Testing Membership in CFL

CYK CYK Algo → Running Time = $O(n^3)$

Given CFG G in CNF $w \in T^*$

Suppose $w = a_1 a_2 \ldots a_n$

X_{ij} is the set of non-terminals A such that

$A \xrightarrow{*} a_i, a_{j+1} \ldots a_j$

X_{ij} we are storing a set of non-terminals

$X_{11} = \{ a_1 \} = a$

$X_{13} = \{ a_1 a_2 a_3 \} = aab$

$$X_{ij} = \bigcup_{K=i}^{j-1} X_{iK} \cdot X_{K+1\, j}$$

$a_i, a_{i+1}, \ldots a_K \mid a_{K+1} \ldots a_j$

Ex Given in CNF

$S \to AB \mid BC$

$A \to BB \mid 0$

$B \to BA \mid 1$

$C \to AC \mid AA \mid 0$

$a_1 a_2 a_3 a_4 a_5 a_6$ and $w = 110100$

Verify the string belongs to the language or not

$a_1 \mid a_2$ (1 1)

$X_{11} \cdot X_{22}$

$\{B\} \cdot \{B\} = \{BB\}$

$a_2 \mid a_3$

$X_{22} \quad X_{33}$

$\{B\} \cdot \{A, C\} = \{BA\}$

$= \{B\}$

$a_3 \mid a_4$

$X_{33} \quad X_{44}$

$\{A, C\} \quad \{B\} = \{AB\}$

X

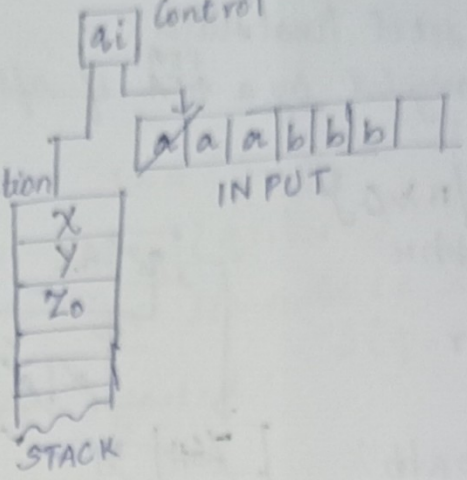
	1	2	3	4	5	6
1	$\{B\}$	$\{A\}$	$\{A, B, C\}$			
2		$\{B\}$	$\{B, S\}$	$\{A\}$		
3			$\{A, C\}$	$\{B, S\}$	$\{S\}$	
4				$\{B\}$	$\{B, S\}$	$\{S, B\}$
5					$\{A, C\}$	$\{C\}$
6						$\{A, C\}$

a_1	a_2	a_3	a_4	a_5	a_6
1	1	0	1	0	0

10/11/2028

PUSHDOWN AUTOMATA : (PDA) Having infinite memory

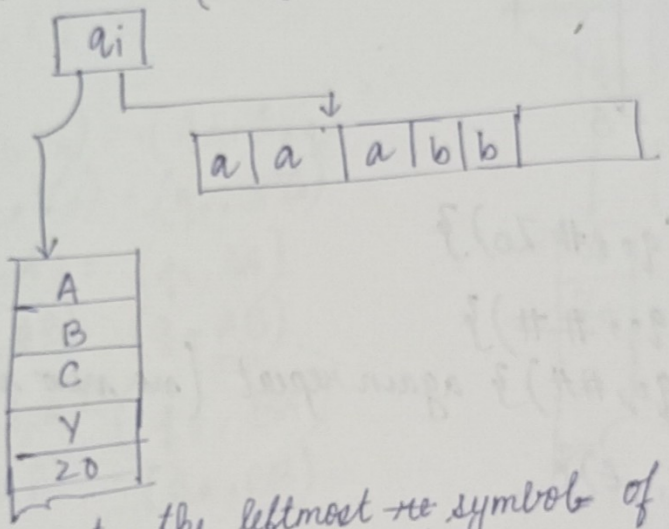
$\{a^n b^n \mid n \geq 0\}$



A standard PDA, is a non-deterministic automata with ϵ-transition

- The PDA having an infinite stack
- The stack having its own alphabet
- The stack alphabet is denoted by Γ 'gamma'.
- Z_0 is a special symbol used to represent empty stack $Z_0 \in \Gamma$
- There may be some common symbol b/w Σ & Γ (Σ is start, current I/P alphabet)

$$\delta(q_i, a, X) = \{(q_j, ABC), (q_i, x) \ldots\}$$



During replacement, the leftmost ~~re~~ symbol of the replacement string will be at the top of the stack

N-PDA - is a 7 tuple M =

$M(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

$Q \rightarrow$ finite set of state

$\Sigma \rightarrow$ input alphabet (finite set of symbol)

$\Gamma \rightarrow$ stack alphabet

$\delta \rightarrow Q \times \Sigma \times \Gamma \rightarrow$ provides a set of pair $\{(q_1, ABC)\}$.

i.e $\rightarrow P(Q \times \Gamma^*)$ → can also be ϵ

↓

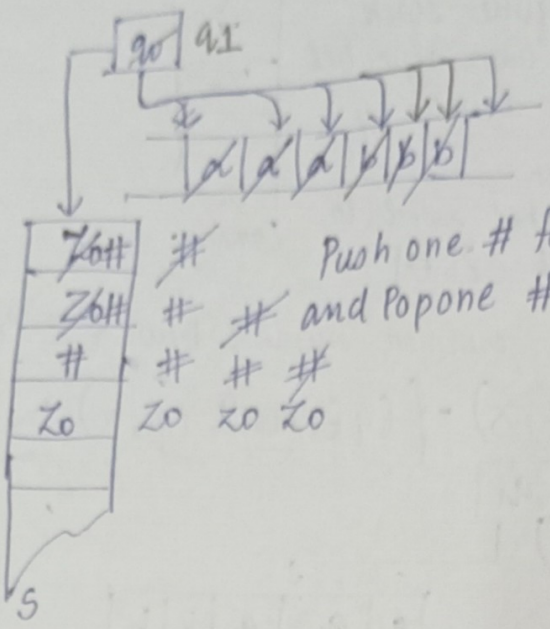
can also be written as

$$Q \times \Sigma \cup \{\epsilon\} \times \Gamma \rightarrow P(Q \times \Gamma^*)$$

$q_0 \in Q$ is the initial state
$Z_0 \in \Gamma$ for initial stack symbol
$F \subseteq Q$ is a set of final states.
A string is accepted by a ~~DFA~~ PDA if after computation, it enters in a final state.

$L = \{a^n b^n / n \geq 0\}$
aaabbbb
or
ab or
aabb
& not aababb

$w = aaabbb$
$\Gamma = \{Z_0, \#\}$



Push one # for every a and Popone # for every b.

Rules
1 - $\delta(q_0, a, Z_0) = \{(q_0, \# Z_0)\}$
2 - $\delta(q_0, a, \#) = \{(q_0, \#\#)\}$
2 - $\delta(q_0, a, \#) = \{(q_0, \#\#)\}$ again repeat (no new rule)

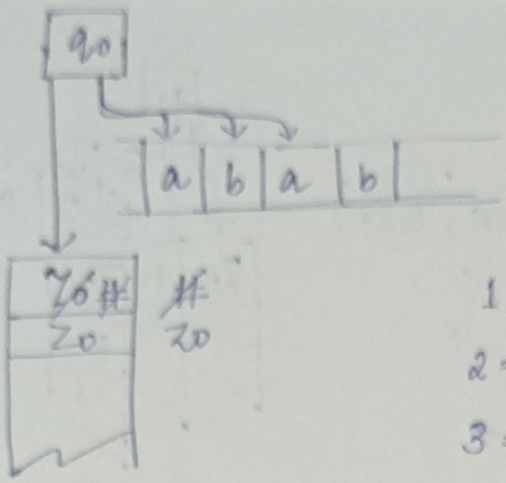
3 - $\delta(q_0, b, \#) = \{(q_1, \epsilon)\}$

4 - $\delta(q_1, b, \#) = \{(q_1, \epsilon)\}$
4 → will repeat again.

5 - $\delta(q_1, \epsilon, Z_0) = \{(q_2, Z_0)\}$ & $F = \{q_2\}$

$Q \rightarrow \{q_0, q_1, q_2\}$
$\Sigma \rightarrow \{a, b\}$
$\Gamma \rightarrow \{Z_0, \#\}$
$q_0 \rightarrow q_0$ $\{Z_0\}$
$F \subseteq Q = \{q_2\}$

Show that abab is reject



1 = $\delta(q_0, a, Z_0) = \{(q_0, \ldots)\}$

2 = $\delta(q_1, b, \epsilon) = \{(q_1, \ldots)\}$

3 = No rule is defined for $\delta(q_1, a, Z_0)$

The string w is rejected.

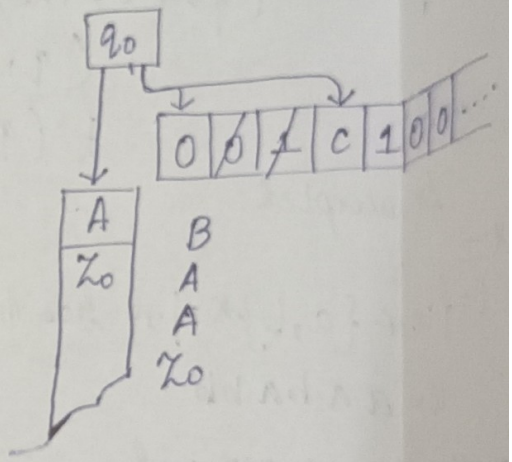
11/2023

Ex: DPDA for $L = \{w c w^R \mid w \in \{0,1\}^*\}$

$\Sigma = \{0, 1, c\}$

A — 0
B — 1

$w = 001c100 \in L$



1 - $\delta(q_0, 0, Z_0) = (q_0, AZ_0)$

2 - $\delta(q_0, 1, Z_0) = (q_0, BZ_0)$

3 - $\delta(q_0, 0, A) = (q_0, AA)$

4 - $\delta(q_0, 0, B) = (q_0, AB)$

5 - $\delta(q_0, 1, A) = (q_0, BA)$

6 → $\delta(q_0, 1, B) = (q_0, BB)$

7 → $\delta(q_0, c, A) = (q_1, A)$

8 → $\delta(q_0, c, B) = (q_1, B)$

9 → $\delta(q_1, 0, A) = (q_1, \epsilon)$

10 → $\delta(q_1, 1, B) = (q_1, \epsilon)$

11 → $\delta(q_1, \epsilon, Z_0) = (q_f, Z_0)$

12 - $\delta(q_0, c, Z_0) = (q_f, Z_0)$

TANTANEOUS DESCRIPTION OF PDA

formally describe the configuration of PDA at a given instanca, we
ne an instantaneous description

ID of PDA is descriped by a triple (q, w, γ) where q → current
of the

w → the remaining string to compute

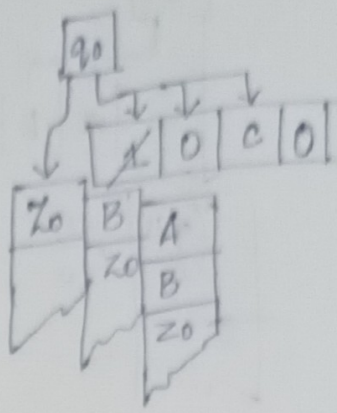
γ → total containts of the STACK.

$w = 10C0$

$(q_0, 10C0, Z_0) \vdash (q_0, 0C0, BZ_0)$

$\vdash (q_0, C0, ABZ_0)$

$\vdash (q_1, 0, ABZ_0) \vdash (q_1, \epsilon, BZ_0)$



So the string 10C0 is rejected

no rule available

Ex $w = 01C10$

$(q_0, 01C10, Z_0)$

$\vdash (q_0, 1C10, AZ_0)$

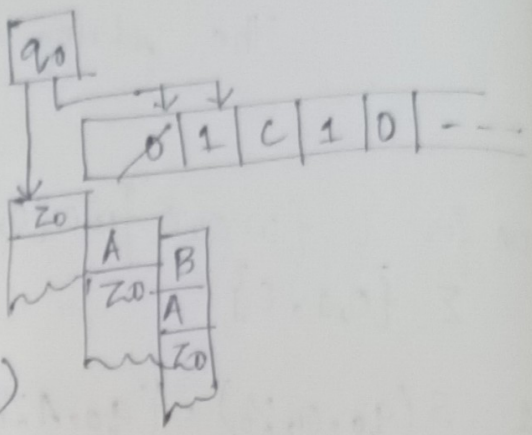
$\vdash (q_0, C10, BAZ_0)$

$\vdash (q_0, 10, BAZ_0) \vdash$

$(q_0, 0, AZ_0)$

$\vdash (q_1, 0 \in ?, Z_0)$

$\vdash (q_f, Z_0)$

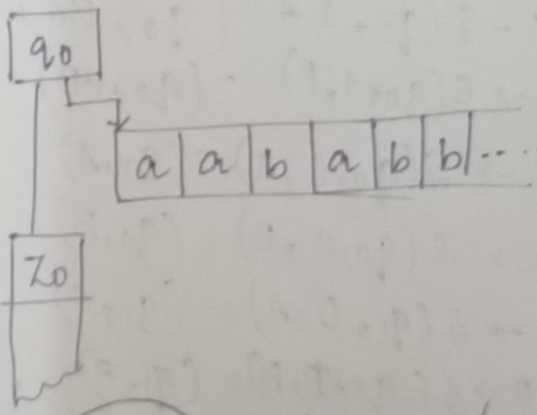


So accepted.

Ex:-

$L = \{w \in \{a,b\}^* \mid w$ has the equal no of a's and b's $\}$

$w = aababb$.



For alternate we pop and for equal symbol we push

1- $\delta(q_0, a, Z_0) = (q_0, aZ_0)$

2- $\delta(q_0, b, Z_0) = [q_0, bZ_0)$

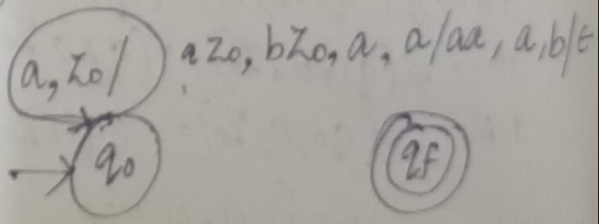
3- $\delta[q_0, a, a) = (q_0, aa)$

4- $\delta(q_0, a, b) = (q_0, \epsilon)$

5- $\delta(q_0, b, a) = (q_0, \epsilon)$

6- $\delta(q_0, b, b) = (q, bb)$

7- $\delta(q_0, \epsilon, Z_0) = (q_f, Z_0)$



present then Yes, else No.

$a_2^3 \{3$

$a_1 \cdot a_2 a_3 \cup a_1 a_2 \cdot a_3$

$\checkmark \quad \searrow \quad \cup \quad \swarrow \quad \searrow$

$x_{11} \cdot \quad x_{23} \quad x_{12} \cdot x_{33}$

$\{\} \cdot \{B,S\} \cup \{A\} \cdot \{A,C\}$

$\{BB, BC, AA, AC\}$

$\{A, C\}$

$a_2 \cdot a_3 \cdot a_4$

$a_2 \cdot a_3 a_4 \cup a_2 a_3 \cdot a_4$

$\swarrow \quad \searrow \quad \swarrow \quad \searrow$

$x_{22} \quad x_{34} \quad x_{23} \quad x_{44}$

$\{B\} \cdot \{S\} \cup \{B,S\}, \{B\}$

$\{BC, BB, SC\}$

$\{A,\}$

$a_3 \cdot a_4 \cdot a_5$

$a_3 \cdot a_4 a_5 \cup a_3 a_4 \cdot a_5$

$\swarrow \quad \searrow \quad \swarrow \quad \searrow$

$x_{33} \quad x_{45} \cup x_{34} \quad x_{55}$

$\{A,C\}, \{B,S\} \cup \{S\} \{A,C\}$

$\{AB, AS, CB, CS\} \cup \{SA, SC\}$

$\{S\}$

$a_4 \cdot a_5 a_6$

$a_4 \cdot a_5 a_6 \cup a_4 \cdot a_5 \cdot a_6$

$\swarrow \quad \searrow \quad \swarrow \quad \searrow$

$x_{44} \quad x_{56} \quad x_{45} \quad x_{66}$

$\{B\} \{C\} \cup \{B,S\} \{A,C\}$

$\{BC, BA, BC, SX, SC\}$

$\{S, B\}$

$x_{14} = x_{11} \cdot x_{24} \cup x_{12} \cdot x_{34} \cup x_{13} \cdot x_{44}$

$x_{26} = x_{22} \cdot x_{36} \cup x_{23} \cdot x_{46} \cup x_{24} \cdot x_{56} \cup x_{25} \cdot x_{66}$

22/11/2023

$L = \{ ww^R \mid w \in \{a,b\}^* \}$

aabbaa | abbaaa

w | w^R

a – push A

b – push B

$\delta(q_0, a, Z_0) = \{(q_0, AZ_0)\}$

$\delta(q_0, b, Z_0) = \{(q_0, BZ_0)\}$

can be used repeatedly:

$\delta(q_0, a, A) = \{(q_0, AA), (q_1, \epsilon)\}$

$\delta(q_0, a, B) = \{(q_0, AB)\}$

$\delta(q_0, b, A) = \{(q_0, BA)\}$

$\delta(q_0, b, B) = \{(q_0, BB), (q_1, \epsilon)\}$

repeated used for removal:

$\delta(q_1, a, A) = \{(q_1, \epsilon)\}$

$\delta(q_1, b, B) = \{(q_1, \epsilon)\}$

$\delta(q_1, \epsilon, Z_0) = \{(q_f, Z_0)\}$

$\delta(q_0, \epsilon, Z_0) = \{(q_f, Z_0)\}$

Check for aab

(q_0, aab, Z_0)

$\vdash$

(q_0, ab, AZ_0)

$\vdash$

(q_0, b, AAZ_0) $\quad$ (q_1, b, Z_0)

$\vdash$ $\quad$ $\vdash$

$(q_0, \epsilon, BAAZ_0)$ $\quad$ (q_1, ϵ, Z_0)

Halting in both cases.

chec Check for abba

$\delta(q_0, abba, Z_0) \vdash \delta(q_0, bba, AZ_0) \vdash$

$\delta(q_0, ba, BAZ_0) \vdash$

$\delta(q_0, a, BBZ_0)$ $\delta(q_0, a, AZ_0)$

Q.) $L = \{a^n b^{2n} \mid n \geq 0\}$

3 a's so 6 b's

a → push # #

B → pop # B A

$\delta(q_0,$

THEORUM:

A language is context free iff some PDA recognise it.

i.e. CFL - CFG

L - ∃G such that L = L(G) PDA - M (Machine)

CFG G ≈ PDA M

$$\boxed{G \equiv M \text{ iff } L(G) = L(M)}$$

Let G = (V, T, P, S) be a context free grammar in GNF.

Construct the PDA M: M (Q, Σ, Γ, δ, q_0, Z_0, F) as follows:

① Language of a PDA M, in Final state:

$$L(M) = \{ w \mid (q_0, w, Z_0) \vdash^* (q_f, \epsilon, \alpha) \}.$$

where $q_f \in F$ $\alpha \in \Gamma^*$

② Language of PDA in empty stack:

$$N(M) = \{ w \mid (q_0, w, Z_0) \vdash^* (q_i, \epsilon, Z_0) \}$$

where $q_i \in Q$.

1 marked

Language accepted by final state

Vs

Language accepted by empty string

27/11/2023

CFG ≡ PDA

Let G = (V, T, P, S) be a CFG in GNF.

Construct a PDA M = (Q, Σ, Γ, δ, q_0, Z_0, ϕ) using the components of G as follows:

Q = {q_0} L(G) = L(M)

Σ = T

The transition

$\delta(q_0, a, A)$ contains (q_0, γ) iff $A \to a\gamma \in P$, $\gamma \in V^*$

$\delta(q_0, a, A) = \{(q_0, \gamma)\}$

$\gamma \in \Gamma^*$

Ex $G = (V, T, P, S)$ $V = \{A, B, C, S\}$ $T = \{a, b\}$

$P = \{ S \to aA,$
$A \to aABC,$
$A \to bB,$
$A \to a,$
$B \to b\}$

$M = (Q, \Sigma, \Gamma, \delta, q_0, \epsilon, \phi)$

$Q = \{q_0\}$

$\Sigma = \{a, b\}$

$\Gamma = \{A, B, C, S\}$ → set of all non-terminals

1- $\delta(q_0, a, S) = \{(q_0, A)\}$

2- $\delta(q_0, a, A) = \{(q_0, ABC)\}$ ✓

3- $\delta(q_0, b, A) = \{(q_0, B)\}$ ..

} same LHS so we can combine also

4- $\delta(q_0, a, A\} = \{(q_0, \epsilon)\}$ ✓

5- $\delta(q_0, b, B\} = \{(q_0, \epsilon)\}$

The PDA is given $\{a^n b^n\}$ ~~be in GNF~~, convert to its equivalent GNF format?

1. $\delta(q_0, a, Z_0) = (q_0, \#Z_0) \approx Z_0 \to a\#Z_0$

1. PUMPING LEMMA FOR CNF

Let L be any CFL, then there exists a constant n depending on L such that if $z \in L$ and $|z| \geq n$, then z can be decomposed as $z = uvwxy$ such that

1 - $|vx| > 0$
2 - $|vwx| \leq n$
3 - $uv^i wx^i y \in L, \forall i \geq 0$

n → is the pumping number

Ex) $L = \{a^n b^n c^n \mid n \geq 0\}$ is not a CFL.

Assume that L is a context-free language, so it must be satisfy pumping lemma. Let p be the pumping number for L. Select the string: $z = a^p b^p c^p \in L$ so $|z| = 3p \geq p$ (Condⁿ 1 satisfied)

Now Split $p = 3$ $u\,|a|_v\,|a|_w\,|a|_x\,|bbb|_y\,ccc$ v & y can be ϵ

$|vwx| \leq p$, $aaa|b|b|b|c|cc$ (OR) v & x cannot simultaneously

$u \quad v \quad w \quad x \quad y$

or $a\,a\,a\,|a|\,b\,|b|\,b\,|c\,e\,c$

$u \quad v \quad w \quad x \quad y$

OR $a\,a\,a\,b\,|b|\,b\,|\,c\,|c\,c$

$u \quad v \quad w \quad x \quad y$

In any of the split v & x cannot contain a's & c's simultaneously.

Case-1 When v & x contain a's only. $u\,|a|_v\,|a|_w\,|a|_x\,|bbb|_y\,ccc$

~~Case-2 When v & x contain b's only.~~

Now $i = 2$ $uv^2wx^2y = aaaaabbbccc$

It is not a string & belonging from the Language.

Case-2 When v & x contain b's only

Case-3 " " " " c's only

Case-4 " " " a & b

Case-5 " " " b & c

28/11/2023

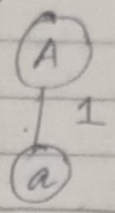
THEORUM:

Suppose we have a parse tree according to a CNF grammar $G=(V, T, P, S)$ and suppose that the yield of the is tree is a terminal string. If the length of the longest path in the tree is 'N'. Then length of the string $\leq 2^{n-1}$

We prove this theorum by PMI applied on n.

Basis

$n=1$: $\{A \to BC, A \to a\}$

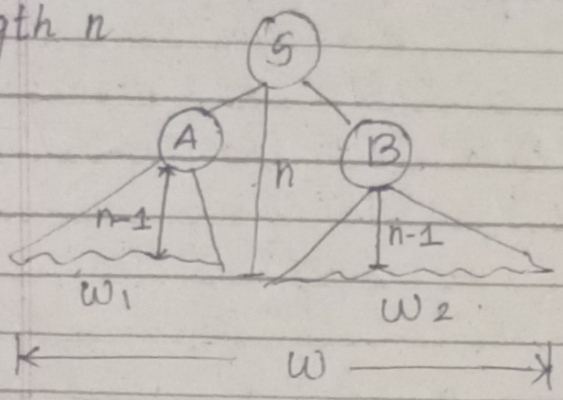


$|a| = 1 = 2^{1-1} = 1$

$|a| \leq 2^{1-1}$ HP!!

INDUCTION:

Let the theorum is true for n-1. Consider a parse with longest pathlength n



$w = w_1 \cdot w_2$.

As $n \geq 2$, the initial non-terminal will substituted by AB.

Let w_1 is the substring generated from A

And w_2 " " " " " B. So the string generated from S is $w = w_1 \cdot w_2$. As per our assumption

$|w_1| \leq 2^{(n-1)-1}$

$|w_2| \leq 2^{(n-1)-1}$ ✓

$|w| = |w_1| + |w_2|$

$= |w| \leq 2^{n-2} + 2^{n-2}$

$|w| \leq 2^{n-1}$

Consider a CNF, $L-\{\epsilon\}$, so we have a CNG in CNF for L.

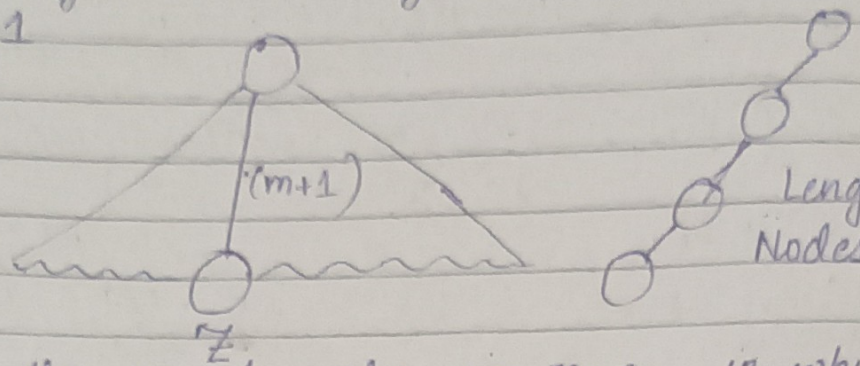
Let in the grammas there are m number of non-terminals

Choose the pumping number $n = 2^m$

Consider a string $Z \in L$ such and $|Z| \geq n$

$\rightarrow |Z| \geq 2^m$

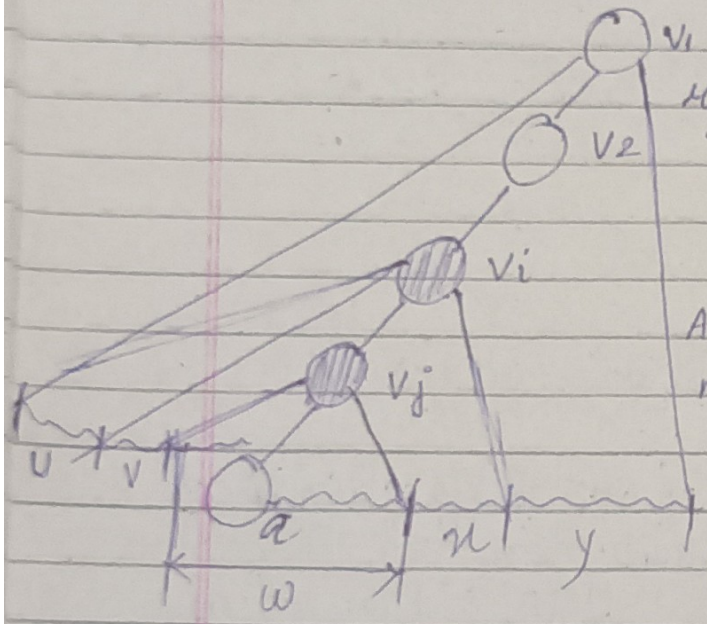
The parse which will generate Z having the longest path with the minimum $m+1$



Along the path the no of nodes is $m+2$, in which the last node is a terminal and rest $m+1$ are non-terminal.

But in our grammar we are having m no of terminal,

$\Rightarrow$ atleast one Non-terminal is repeated twice.



Let the non-terminal repeated along the path is V_i, V_j (i.e $V_i = c$ then $V_j = c$)

The substring generated from $V_i = vw$

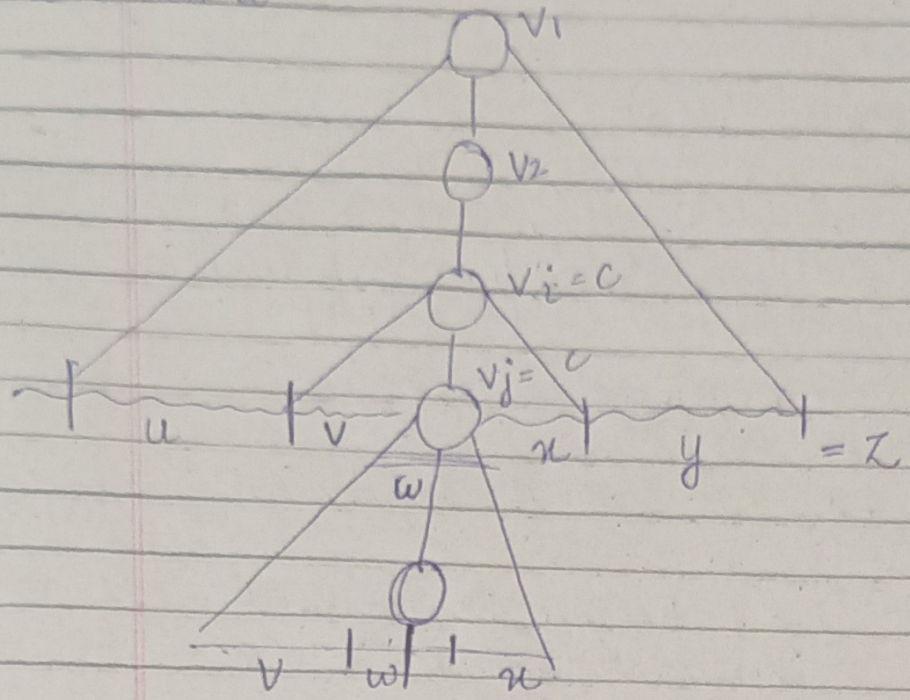
" " " " $V_j = w$

As our grammer is in CNF no non-transition or null production is there. And as V_i & V_j are the 2 distanct nodes along the longest path, so both v & x cannot be epsilon along the longest

Any possi string, acc to the theorum, the parse string which generates the string

According to the theorum, along the longest path of ~~string~~ Z the parse string which generates the string Z will have minimum $m+2$ nodes

so, Among the first m+1 nodes the repeation will occur
→ condition 2



It is clear from the drawn diagram, for any $i \geq 0$, $uv^i wx^i \in L$

For i = 2

If Vi can generate vwx, then Vj can also generate vwx. (∵ Vi = C & Vj = C)

29/11/2023

$L = \{a^i \mid i \text{ is a prime}\}$ is not a CFL

Assume that L is a context free language. Let m be the pumping number for L

Proof: ~~Let m be the pumping lemma~~ Consider a prime $p \geq m$ for the string $z = a^p$; $z \in L$.

and $|z| = p \geq m$

z can be splitted as $z = uvwxy$

such that ① $|vx| > 0$,

② $|vwx| \leq m$.

Let $|u| = r$ and $|x| = q$

③ $uv^i wx^i y \in L$; $i \geq 0$

For $i = p+1$

the string is $uv^{p+1} wx^{p+1} y$ should belongs to L but $|uv^{p+1} wx^{p+1} y| = |uvwxy| + |v^p| + |x^p|$

$= p + pr + pq = p(1+r+q)$

p can never be zero as minimum p is 2 and r & q are $\neq \epsilon$ Here both the factors are greater than 1.

→ $uv^{p+1} wx^{p+1} y \notin L$ because the length of $p(1+r+q)$ is a composite number. As the condition does not satisfy pumping lemma, it ~~does~~ is not a content-free language.

Properties of CFL:-

1. Every CFL satisfies pumping lemma.
2. Closure property of CFL:-

1) CNL are closed under union

Proof: If L_1 and L_2 are CFL. Then ∃ CNG $G_1(V_1, T_1, P_1, S_1)$ and $G_2(V_2, T_2, P_2, S_2)$ describing L_1 & L_2 respectively.

$L_1 \cup L_2$ is CFL

Construct grammar

$G = (V, T, P, S)$ as follows:-

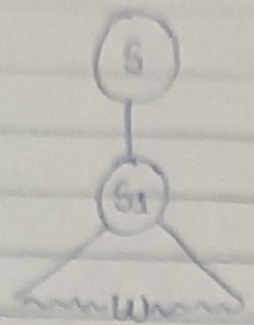
~~V_1~~ $V = V_1 \cup V_2 \cup \{S\}$

$T = T_1 \cup T_2$

$P = P_1 \cup P_2 \cup \{S \to S_1 | S_2\}$

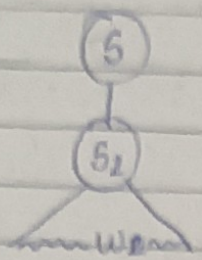
For $w \in L_1 \cup L_2$

Case-1 : $w \in L_1$



as $w \in L_1$ and G_1 is the CNF CFG for L_1

Case-2 $w \in L_2$

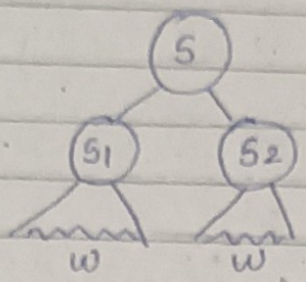


as $w \in L_2$ and G_2 is the CFG for L_2

Case-3 $w \in L_1 \cup L_2$

$w \in L_1$ or

$w \in L_2$



(2) Concatenation property of CFL

L_1 and L_2 are CFL with the CFG $G_1 = (V_1, T_1, P_1, S_1)$ and $G_2 = (V_2, T_2, P_2, S_2)$

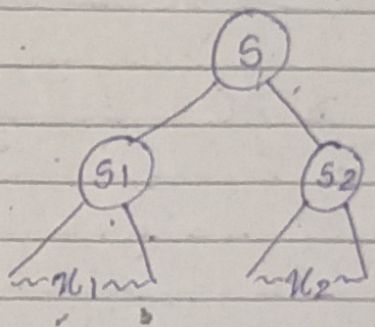
$L_1 \cdot L_2 = \{ w / w = x_1 \cdot x_2$ where $x_1 \in L_1$ and $x_2 \in L_2 \}$.

Consider $G = (V, T, P, S)$

$V = V_1 \cup V_2 \cup \{S\}$

$T = T_1 \cup T_2$

$P = P_1 \cup P_2 \cup \{S \to S_1 S_2\}$



(3) Star Operation of CFL

If L is a CFL then L^* is also a CFL

L is a CFL with CNG $G_1 = (V_1, T, P_1, S_1)$

We can construct a CFG,

$G = (V, T, P, S)$ for L^* as follows:

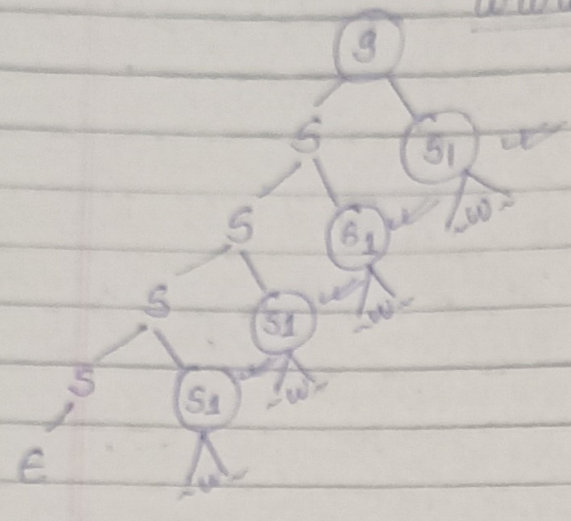
1 $V = V_1 \cup \{S\}$

$T = T$

$P = P_1 \cup \{S \to SS_1 \to \epsilon\}$

For example w ∈ L,

wwww (produce 4 w's)



NEGATIVE PROPERITIES OF CFL :-

- CFL's are not closed under intersection

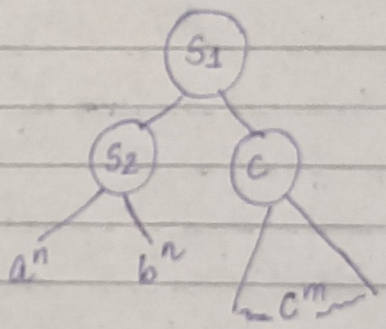
We will prove it by contradiction

$L_1 = \{a^n b^n c^m \mid n, m \geq 0\}$ is a CFL

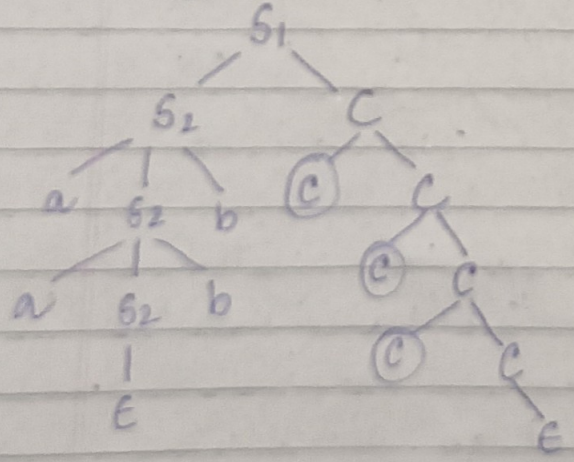
$S_1 \to S_2 C$

$S_2 \to a S_1 b \mid \epsilon$

$C \to cC \mid \epsilon$



Ex aabbccc



Now $L_2 = \{ a^i b^j c^j \mid i, j \geq 0 \}$ is a CFL

CFL

$S_2 \rightarrow AS_3$

$S_2 \rightarrow bS_3C \mid \epsilon$

$A \rightarrow aA \mid \epsilon$

$L_1 \cap L_1 = \{ a^K b^K c^K \mid K \geq 0 \}$ is not CFL

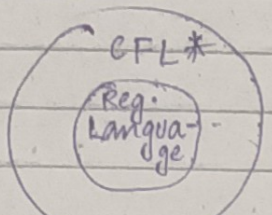
So there is a contradiction and is not under under intersection property

4/12/2023

CLOSED UNDER COMPLIMENTATION

Assume that the class of CFL closed under complementation. If L_1 and L_2 are two CFL's, then

$\overline{L_1} \cup \overline{L_2} = L_1 \cap L_2 \Rightarrow$ which implies that CFL's are closed under intersection which is a contradiction. ∴ Intersection property are not CFL's.

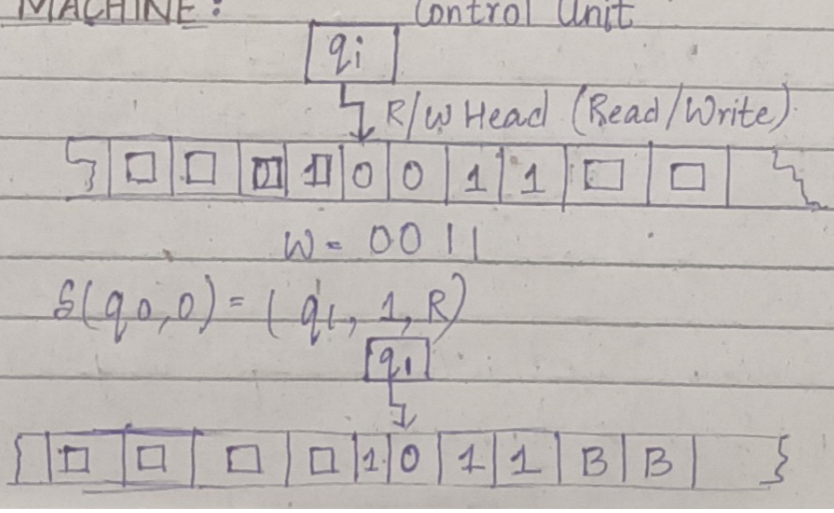


$\Sigma = \{0, 1\}$ $L \subset \Sigma^*$ is a

$\Sigma^* = \{0, 1\}^*$ language.

$L_1 = \{ a^n b^n c^n \mid n \geq 0 \}$ & $L_2 \{ a^i \mid i$ is a prime$\}$

TURING MACHINE:



$\Sigma \subset \Gamma \rightarrow$ (Blank is extra in Σ)

1. TURING MACHINE:

A turing machine M is a 7 tuple: $M(Q, \Sigma, \Gamma, \delta, q_0, \square, F)$

Q - finit finite set of state

Σ -

Γ - input alphabet

δ - T

Q - Finite set of states

Σ - Input Alphabet (alway finite)

Γ - tape "

δ: transition function

$\delta: Q \times \Gamma \rightarrow (Q \times \Gamma \times \{L, R\})$

q_0: Initial state

$\square \in \Gamma$ or blank

$F \subseteq Q$; F: set of final states

ACCEPTANCE OF A STRING BY TURING MACHINE:

~~When the~~ A string w is said to be accepted by a turing machine M, if the turing machine halts in the final state.

EX- 1 - $\delta(q_0, a) = \{(q_1, a, R)$

2 - $\delta(q_0, b) = (q_1, b, R)$

3 - $\delta(q_1, a) = (q_0, \square, R)$

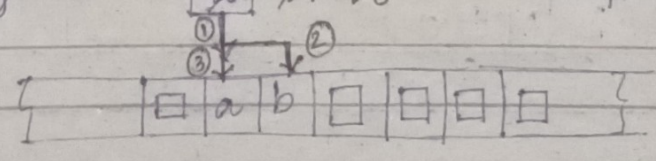
4 - $\delta(q_1, a) = (q_0, a, L)$

5 - $\delta(q_1, b) = (q_0, b, L)$

6 - $\delta(q_1, \square) = (q_0, \square, L)$

$Q = \{q_0, q_1\}$ $\Sigma = \{a, b\}$ $\Gamma = \{a, b, \#, \$, \square\}$ $F = \phi$

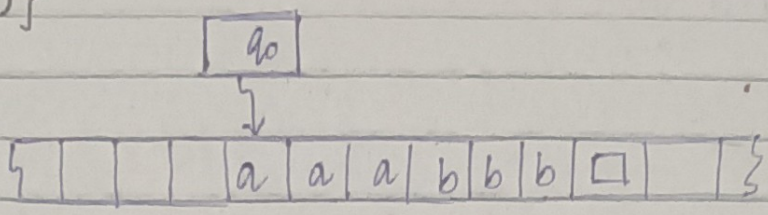
$Q = \{q_0, q_1, q_2, q_3, q_f\}$



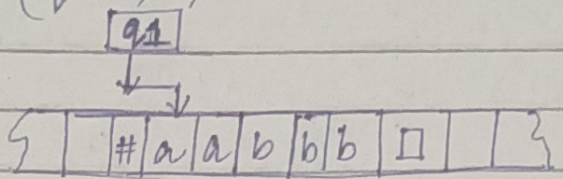
* This turing machine will not halt for any string.
* Will half for infinite loop for all string

Q) $L = \{a^n b^n \mid n \geq 0\}$

Replace
$a \rightarrow \#$
$b \rightarrow \$$

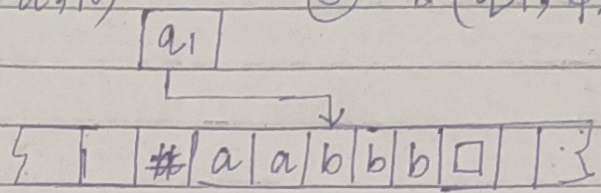


1 - $\delta(q_0, a) = (q_1, \#, R)$

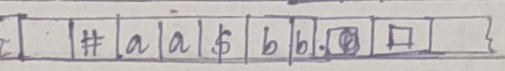


2 - $\delta(q_1, a) = (q_1, a, R)$

③ $\delta(q_1, \$) = (q_1, \$, R)$

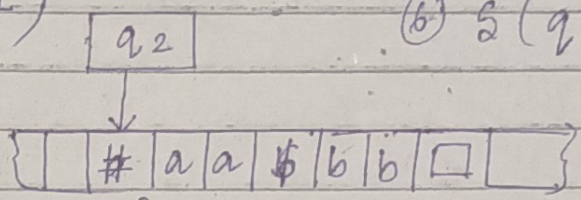


4 - $\delta(q_1, b) = (q_2, \$, L)$ →

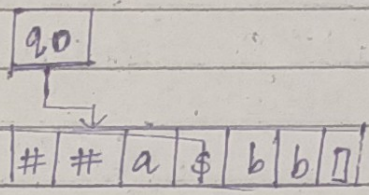


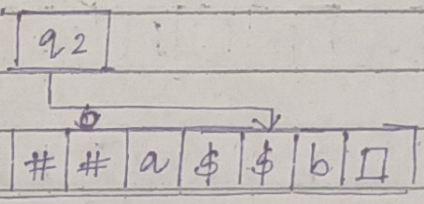
5 - $\delta(q_2, a) = (q_2, a, L)$

⑥ $\delta(q_2, \$) = (q_2, \$, L)$



7 - $\delta(q_2, \#) = (q_0, \#, R)$





8 - $\delta(q_0, \$) = (q_3, \$, R)$

9 - $\delta(q_3, \$) = (q_3, \$, R)$ used repeatedly to skip $

10 - $\delta(q_3, \square) = (q_f, \square, L)$

11 - $\delta(q_0, \square) = (q_f, \square, L)$

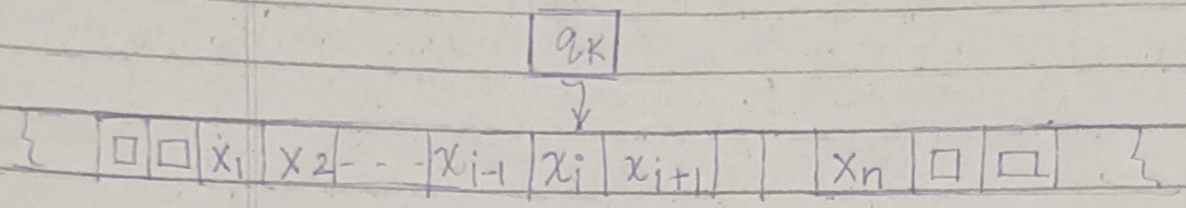
1 $w = a\ a\ b$ · Check for w

5/12/2023

ID OF TURING MACHINE:

ID of turing machine is represented by a string — "$X_1 X_2 X_3 \ldots X_{i-1} q_K X_i X_{i+1} \ldots X_n$"

→ The non-blank portion is X_1 to X_n

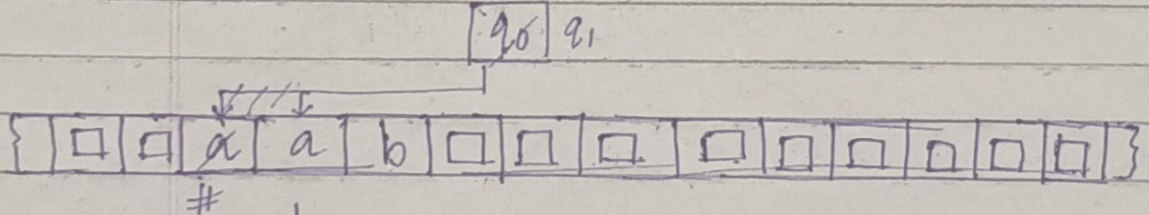


Where q_K is the current state of the TM

$X_1, X_2 \ldots X_n$ is the portion of the tape b/w the leftmost & rightmost non-blank

The read-write head is currently scanning the i^{th} symbol from left (i.e X_i)

Q) $L = \{a^n b^n \mid n \geq 0\}$ compute aab



$q_0\ a\ a\ b \vdash \#\ q_1\ a\ b \vdash \#a\ q_1\ b \vdash \#\ q_2\ a\ \$$

$\vdash q_2\ \#\ a\ \$ \vdash \#\ q_0\ a\ \$ \vdash \#\#\ q_1\ \$$

$\vdash \#\#\$\ q_1\ \square$

no rule defined for $q_1 \square$, so not in final state. So the string - aab is recieved rejected.

w- aabb

$q_0 aabb \vdash \# q_1 abb \vdash \# a q_1 bb \vdash \# a \$ q_2 b \$$

$\vdash \# a q_2 \$ b \vdash \# a q_2 \$ \$ b \vdash$

2.) $L = \{ w \mid w \in \{0,1\}^* \text{ and have even no. of 1's} \}$.
(Initial state is the final state)

$\Sigma = \{1\}$ $w = 111$

$q_0 w \vdash q_f w w$

1 - $\delta(q_0, 1) = (q_0, \#, R)$
2 - $\delta(q_0, \square) = (q_1, \square, L)$
3 - $\delta(q_1, \#) = (q_1, \#, L)$
4 - $\delta(q_1, \square) = (q_2, \square, R)$
5 - $\delta(q_2, \#) = (q_3, 1, R)$
6 - $\delta(q_3, \#) = (q_3, \#, R)$
7 - $\delta(q_3, \square) = (q_1, 1, L)$
8 - $\delta(q_1, 1) = (q_2, 1, R)$
9 - $\delta(q_3, 1) = (q_3, 1, R)$
10 -

Step 1 - replace 1 by #.

Step 2 - move left replace leftmost hash to 1 and then move right and replace blank with 1

Step 2 - replace the rightmost # by 1 and move right and create # 1 at blan

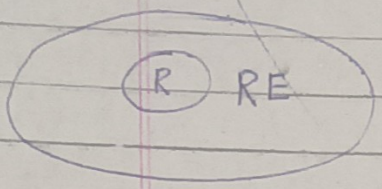
Step 3 - Repeat step 2 until no more # is found

1. LANGUAGE OF TM:

~~TURN~~ Turing Recognizable Language / Recursively Enumerable Language

A language L is called Turing Recognizable/R.E if there exit a TM M such that ① for all $w \in L$, TM M halts in a final state/accept it.

② for all $w \notin L$, TM M', TM M halts in a non-final state reject it.



- Recursively language are subset of recursive

CHURCH-TURING HYPOTHESIS:

Acc to the

- Anything that can be computed on any digital computer can also be computed on turing machine.
- No one has yet been able to suggest a problem solvable by a algorithm for which a turing machine cannot be designed.
- Alternative models have been for mechanical compution like lambda-calculus but none of them are more powerful that turning machine

1 ENCODING A TM

$M(Q, \Sigma, \Gamma, \delta, q_0, \square, F)$

$Q = \{q_1, q_2, \ldots, q_n\}$

$\Gamma = \{a_1, a_2, \ldots, a_n\}$

$q_0 = q_1 = 1$

q_2 is the only Final State

$q_1 = 1$

$q_2 = 11$

$q_n = \underbrace{11\ldots1}_{n\text{-times}}$

Γ

$a_1 = 1$

$a_2 = 11$

$a_K = \underbrace{11\ldots1}_{K \text{ nos}}$

$L = 1$

$R = 11$

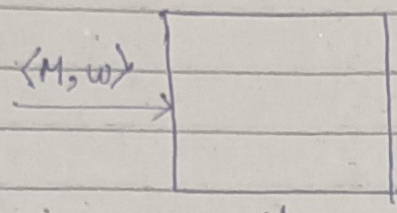
$\delta(q_2, a_3) = (q_4, a_2, R)$

00110101 11

0011 0111 0111 1 0 11011000

UNIVERSAL TURING MACHINE (M_U):-

An universal turning machine Mu is an autometon that "given as input the description of any TM M and a string w, can simulate the computation of M on w"



THEOREM:

Let S be an infinite countable set then a power set (P.S) is not countable or innumerable.

PROOF: We will prove this theorem by contradiction

Given that S is infinite countable

so 'S' can be enumerated as $S = \{s_1, s_2, s_3, \ldots\}$

Let $t \in P(S)$

$\Rightarrow t \subseteq S$

we can represent t as a string of 0 and 1 as follows:

$t = \{s_3, s_8, s_9\}$

" 001 00001100 ....

Assume P(S) is countable, so $P(S) = \{s_1, s_2, s_3, \ldots\}$

$P(S) = \{t_1, t_2, t_3, t_4, \ldots\}$

where each $t_i \in P(S)$, $t_i \subseteq S$

Each t_i can be also be represented as a string of 0 & 1

$t_1 = (0)1001 0100 \ldots$

$t_2 = 1(1)10100 \ldots$

$t_3 = 000(0)110 \ldots$

$t_4 = 1111(1)11 \ldots$

1. $t_K = 0101$ - - - -

$\overline{t_K} = 1010$ - - - -

$S = \{s_1, s_2, s_3, \ldots\}$

$\overline{t_K}$ cannot t_K because the the first is flipped. $\therefore \overline{t_{11}} \neq t_{11}$ so $t_K \neq P(S)$. Hence our assumption is wrong. $\therefore P(S)$ is not uncountable.

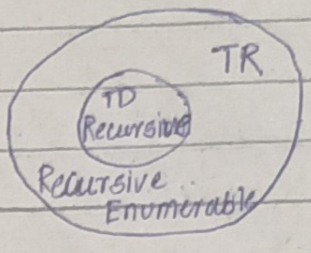
12/12/2023

UNRESTRICTED GRAMMAR:

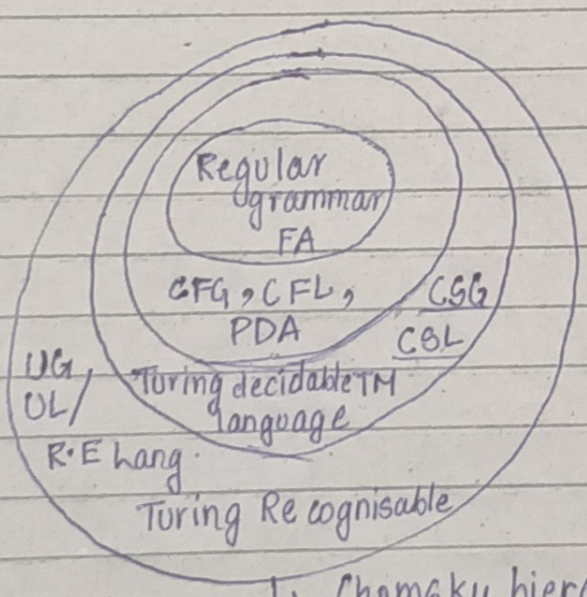
A grammar $G = (V, T, P, S)$ is called an unrestricted if all productions are of the form

$u \rightarrow \alpha$ such that

$u \in (V \cup T)^+$



→ Language generated by an unrestricted grammer is recursively enumerable.



↳ Chomsky hierarchy of languages/ grammar.

THEORUM:

The set of turing machine although ~~infith~~ infinite is countable.
We can encode each turning machine as a string of 0's & 1's and construct the following enumeration procedure 1- Generate the string ~~for~~ over $\{0,1\}^+$ in proper order.
2- Check the generated string is a TM, If it is, write it on the tape and put t_1 and # as separator.
3- Go to step-1

The set of turning machine is countable we can list them ~~to~~ as ~~infinity~~ M_1, M_2, M_3 ....

So, their language : $L(M_1), L(M_2), L(M_3)$ - - - - is also countable.

⇒ The set of recursively enumerable languages are countable (proved).

~~Theorum: On any~~

THEORUM: On any alphabet Σ, there exists languages that are not recursively enumerable.

Σ, $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, - - - - - - \}$

$P(\Sigma^*)$ not countable (using theorum - if S is an infinite & countable set then P(S) is not countable)

$L \in P(\Sigma^*)$
$L \in \Sigma^*$

Set of all language over Σ is $P(\Sigma^*)$ is not countable.

↓
Set of all languages over Σ.

→ But the set of ~~the~~ recursively enumerable languages are countable
→ There are languages which are not recursively enumerable.

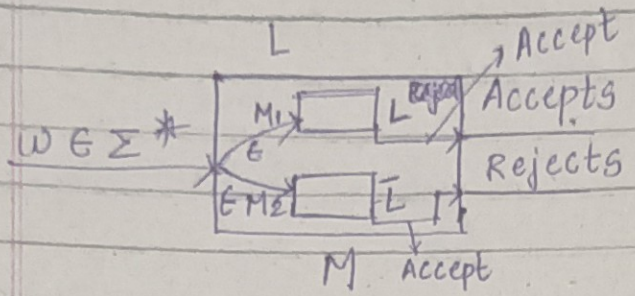
THEORUM: If a language L and its compliment $\bar{L}$ are both recursively enumerable then both languages are recursive.

1. RE (R) L over Σ^* is called recursively enumerable (R.E) if for $w \in L$ M accepts w

For L over Σ is recursive iff $\exists$ a TM M such that

1- for all $w \in L$, M accepts w

2- for all $w \notin L$, M rejected w.



As $\bar{L}$ and L are recursively enumerable: there exist TM M_1 & M_2 which will recognise L and $\bar{L}$ respectively. Construct a TM M as per the following diagram. (see above)

As the diagram it can be seen that for any string $w \in \Sigma^*$, the TM M is accepting and rejecting it. (accepts when M_1 accepts and rejects when M_2 accepts)

M is the decider for L, and hence L is decidable.

~~Jan~~ 13/12/2023.

THEORUM: There exist a recursively enumerable language where compliment is not recursively enumerable.

Let the alphabet $\Sigma = \{a\}$ and consider the set of all TM with this I/P alphalet.

We know that, the set of TM is countable.

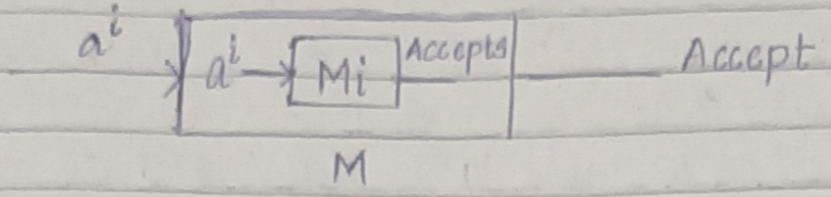
So, the set of TM here ~~is~~ also is enumerable/countable.

Let, enumerate the set of TM as $M_1, M_2, \ldots$

$\Rightarrow L(M_1), L(M_2), L(M_3), \ldots$ is countable

Create a language L as follows:

$L = \{a^i \mid a^i \in L(M_i)\}$



L is a recursively enumerable language.

$\bar{L} = \{a^i \mid a^i \notin L(M_i)\}$

Prove that $\bar{L}$ is not recursively enumerable by contradiction.

Assume that $\bar{L}$ is recursively enumerable.

$\bar{L}$ must be in the list of all "enumed" languages over a or Σ

Let $\bar{L} = L(M_K)$

For the string a^K

CASE-I: If $a^K \in \bar{L} \Rightarrow a^K \in L(M_K) \Rightarrow a^K \in L$

A same string cannot belong to L & $\bar{L}$, at the same time, so there is a contradiction.

CASE-II: If $a^K \notin \bar{L} \Rightarrow a^K \in L \Rightarrow a^K \in L(M_K) \Rightarrow a^K \in \bar{L}$

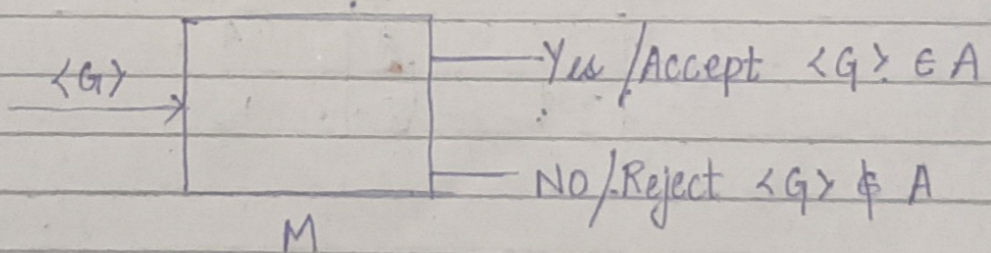
Hence our assumption is contradiction and $\bar{L}$ is not recursively enumerable.

HIGHLEVEL DESCRIPTION OF TURING MACHINE:

EX: $A = \{\langle G \rangle \mid G \text{ is a connected, undirected graph}\}$

ST language A is decidable.

* $\langle G \rangle$ → The string enclosed in $\langle\rangle$, implicitly encoded as a string



$\langle G \rangle$ is not a graph
or $\langle G \rangle$ is not a directed
or $\langle G \rangle$ is not connected
} all three will be Reject

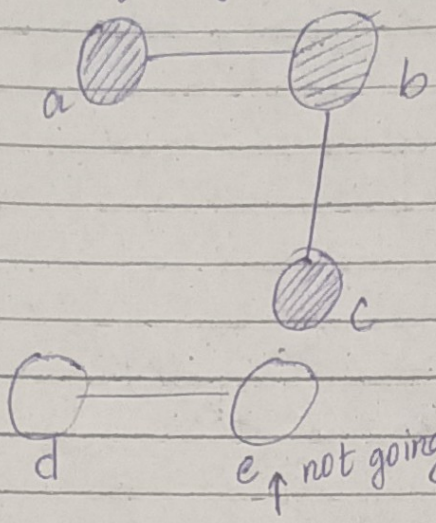
1. The name of TM is M = "On input $\langle G \rangle$:

STEP 1: Select the 1st node of G and randomly mark it.

STEP 2: Repeat the step no-3 until no new nodes are marked.

STEP 3: for each node in G mark it if it is connected to a node through a edge that is already marked.}

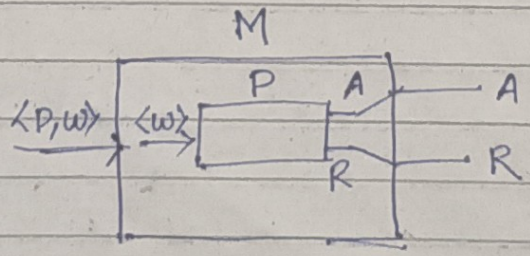
STEP 4: Out Scan all the nodes to determine whether they all are marked. If they are marked, accepts else reject.



The M is a decides of $\mathcal{L}$ for a, and hence a is decidable.

1 18/12/2023 Decidability

Theorum $A_{DFA} = \{ \langle D, w \rangle \mid D \text{ is a DFA that accepts } w \}$



The name of the machine is M M = " On input $\langle D, w \rangle$: D is a DFA and w is a string

STEP1: Stimulate the computation of D on string w

STEP2: If the simulation ends in an accept state then accept otherwise reject

∴ M is the decider for A_{DFA} so A_{DFA} is decidable

Theorum $A_{NFA} = \{ \langle N, w \rangle \mid N \text{ is an NFA that accepts } w \}$

M = "On input $\langle N, w \rangle$: N is an NFA and w is a string

STEP1: Convert it NFA 'N' to its equivalent DFA 'D'

STEP2: Stimulate the computation of D on string w

STEP3: If the st simulation ends in an accept state then accept otherwise reject

Theorum: $A_{REX} = \{ \langle R, w \rangle \mid R \text{ is a R.E. that describe } w \}$

M : "On input $\langle R, w \rangle$: R is a RE and w is a string :

STEP1: Convert the regular expression R to equivalent DFA 'D'

STEP2: Simulate the computation of D on string w

STEP3: If the simulation ends in an accept state then accept otherwise reject

Theorum $A_{CFG} = \{ \langle G, w \rangle \mid G \text{ is a CFG that grammar } w \}$

For any string w, where $|w| = n$ can be derived in $2n-1$, steps using a CNF CFG

"On input $\langle G, w \rangle$:

STEP1: Convert in the given CFG G to its equivalent CNF grammar

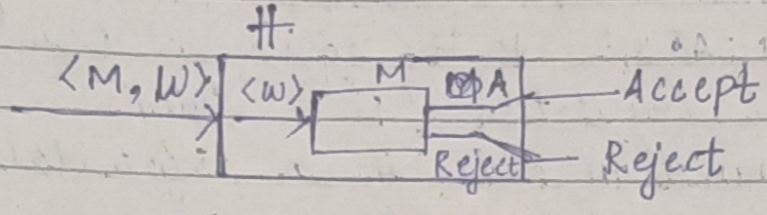
STEP2: Generate all strings in $2n-1$ steps using G where n is the length of the string

STEP 3: If any of these derivation generates w then accept if not reject.
M is the # decider for ACFG, ∴ ACFG is decidable

DIRECT NON-DECIDABLE LANGUAGES:

1. $A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts } w\}$ - PT A_{TM} is undecidable.

Prove this theorum by contradiction.
Assume the A_{TM} is decidable and let 'H' is the decider for A_{TM}.



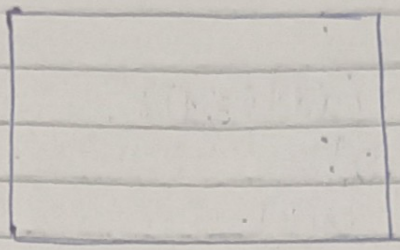
$$H(\langle M, \langle M\rangle\rangle) = \begin{cases} \text{accept if M accepts } \langle M\rangle \\ \text{Reject if M reject } \langle M\rangle \end{cases}$$

Construct an TM 'D' with H as input

$$D(\langle D\rangle) = \begin{cases} \text{accept if D reject D} \\ \text{reject if D accepts D} \end{cases}$$

∴ This is a contradiction. So A_{TM} is non-decidable

THEORUM $HALT_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM that halts on } w\}$
PT $HALT_{TM}$ is undecidable.
~~Let HALT TM is decidable. Design a decider~~



~~On input S=~~ " S = " on input $\langle M, w \rangle$: where M is TM and w is a string

Run the TM 'R' on $\langle M, w \rangle$
If R rejects then rejects,
If R accepts " o/p is accepts.

Here S is the decider for A_{TM} and hence A_{TM} is decidable but (as seen above A_{TM} is non-decidable which is a contradiction.
∴ $HALT_{TM}$ is undecidable.

19/12/2023

Basic Primitive Recursive function:

1 - Zero function : $Z(x) = 0$
2 - Successor function: $S(x) = x + 1$
3 - Projection function: $P_1(x_1, x_2) = x_1$
$P_2(x_1, x_2) = x_2$.

A function build from basic recursive function is called a primitive recursive function.

Composition:

$$f(x,y) = h(g_1(x,y), g_2(x,y))$$

Ex
$$add(x, 0) = x$$
$$add(x, y+1) = s(add(x,y))$$

Compute add (3,2)

$$add(3,1+1) = S(add(3,1))$$
$$= S(S(add(3,0))$$
$$= S(S(3)) = S(4) = 5$$

Function is considered primitive recursive if it can be obtained from basic functions and through finite number of composition and recursion steps.

~~ACK~~

ACKERMAN'S FUNCTION

$A(0,y) = y+1$ (neither zero, nor ~~predict~~ injection nor successor)

$A(x,0) = A(x-1,1)$

$A(x,y+1) = A(x-1, A(x,y))$

Compute $A(1,2)$ or $A(2,1)$

$A(0, A = A(1,1+1) = A(0, A(1,1))$

$A(0, A(1, 0+1))$

$A(0, A(0, A(1,0)))$

$= A(0, A(0, A(0,1)))$

$= A(0, A(0,2))$

$= A(0,3) = 4$

Compute $A(4,3) = 2^{2^{65536}} - 3$

Gödel - Numbering is a function that assigns to each symbol an well-formed formula of Formal language a unique natural number called ~~Gold~~ Godel Numbering.

$\Sigma(a,b) \rightarrow a^2$

$L = \{ w \mid w \in \Sigma^* \text{ and ends with: } a \}$

$= \{a, aa, ba, \dots\}$

$2^1 \times 3^2 \times 5^1 = 2 \times 9 \times 5 = 90$

aba $1^2 * 2^3 * 1^5 = 1 * 8 * 1 = 8$

{2, 3, 5 - - -} mapping function → maps to a distinct natural no.

* Given a sequence $(x_1, x_2, \ldots x_n)$ then Godel encoding of the sequence is $enc(x_1, x_2, \ldots x_n) = (2^{x_1} \cdot 3^{x_2} \cdot 5^{x_3})$

a) What is the godel no for aba?

$(2, 3, 5) = 2^1 \times 3^1 \times 5^1 = 6 \times 5 = 30$

Σ {0, 1} · Godel nomber for 1101

0 1 $1^1 \times 1^1 \times 0^0 \times 1^1 = 1$

20/12/2023

POST'S CORRESPONDENCE PROBLEM (PCP)

PC - soln : Given two sequence of n strings on some alphabet Σ;

Say $A = w_1, w_2, \ldots, w_n$

and $B = \gamma_1, \gamma_2, \ldots, \gamma_n$

We say that $\mathcal{F}$ a PC solution for the pair (A, B) if $\mathcal{F}$ a non-empty sequence of integer/indices i, j, ..., K such that

$w_i, w_j, \ldots, w_K = \gamma_i, \gamma_j \ldots \gamma_K$

sequence of string

$A = \underset{w_1}{11}, \underset{w_2}{1}, \underset{w_3}{100}, \underset{w_4}{111}$ $\quad \Sigma = \{0, 1\}$

$B = \underset{\gamma_1}{111}, \underset{\gamma_2}{00}, \underset{\gamma_3}{001}, \underset{\gamma_4}{11}$ Find the sequences of indices

(1, 4) is one of the solution

$(w_1, w_3, w_4) = \langle 111001111 \rangle$ } same

$(\gamma_1, \gamma_3, \gamma_4) = \langle 111001111 \rangle$ }

$\langle 1, 3, 4 \rangle$

The post-correspondance problem is to divide an algorithm that will tell us if for any given pair of (A, B) of sequence J is a PC soln or not.

- PCP is undecidable which can be proved in contradiction (not in syllabus). If we assume PCP is decidable then we conclude the A_{TM} is decidable But A_{TM} is not decidable (proved earlier) which is a contradiction.

∴ PCP is undecidable.

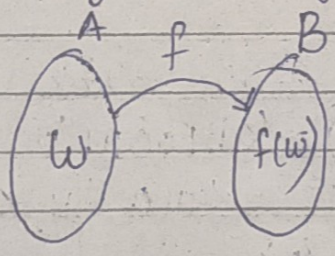
~~COMPLETED~~

REDUCIBILITY:

A function f from: $\Sigma^* \rightarrow \Sigma^*$ is a computable function, if some TM 'M' exists such that for every w, M halts in f(w) on the tape.

MAPPING REDUCIBLE

A language A is called mapping reducible to language B, written as: $A \leq_m B$ if J a computable function f such that for every w belongs to A we compute f(w) ∈ B



THEORUM: If $A \leq_m B$ and B is decidable then A is decidable.

N = "w∈A,

1 - Compute $f(w) \in B$
2 - ~~Rum~~ Run $f(w)$ on M
3 - O/P as per the O/P of M.

M is decider for N
if M accepts $f(w)$ then N accepts w
if M rejects $f(w)$ then N rejects w