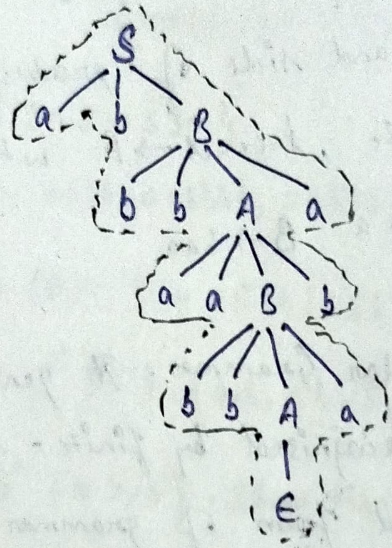
Assignment - 3

Ques 1. Give the derivation tree for $w = abbbaabbaba$ for the grammar:

$S \rightarrow abB$, $A \rightarrow aaBb$, $B \rightarrow bbAa$, $A \rightarrow \epsilon$



Ques 2. Describe Chomsky Hierarchy with all levels

As According to Chomsky Hierarchy, a grammar is divided into 4 types as follows:

i) Type 0-Grammar: Unrestricted grammar. It includes all formal grammar. Type 0 Grammar languages are considered by turing machine, and are also known as the Recursively Enumerable languages
Grammar Production: $\alpha \rightarrow \beta$, where
$\alpha \in (V+T)^* V (V+T)^*$
$\beta \in (V+T)^*$

Ex- $Sab \rightarrow ba$ $A \rightarrow S$

ii) Type 1-Grammar: Context-Sensitive Grammar Language generated by this grammar is recognised by the linear bound automata.
In type 1: • First all type 1 grammar should be type 0.
• Grammar Production: $\alpha \rightarrow \beta$
where $\alpha \in V^*$, $\beta \in (V+T)^+$, $|\alpha| \leq |\beta|$

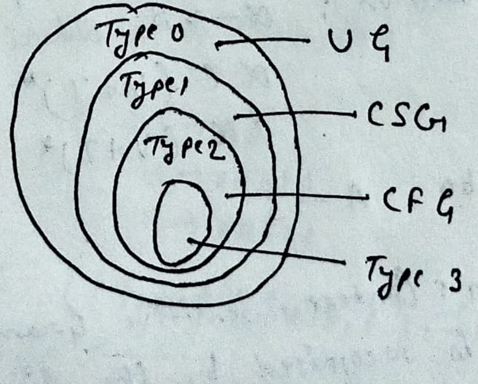
Ex- $S \rightarrow AB$ $AB \rightarrow abc$ $B \rightarrow b$

iii) Type 2-Grammar: Context free grammar. It generates context free language which is recognised by a pushdown automata.
In type 2 : • first, it should be type 1.
• The left hand side of production can have only one variable, i.e, $\alpha \to \beta$ where $\alpha \in V$

Ex- $S \to AB \quad A \to a \quad B \to baa$

iv) Type 3-Grammar: Regular Grammar. It generates regular language which is recognised by finite-state automata. It is a most restricted form of grammar.
It should be given in a form:
• Left-regular : $V \to VT/T$
• Right-regular : $V \to TV/T$

Ex- $S \to a$
$S \to Aa$ (LRG)
$S \to aA$ (RRG)



Ques 3. Find context free grammar for the following:

i) $L = \{a^n b^m \mid n \leq m+3\}$

$L = \{e, a, aa, aaa, b, ab, aab, aaab, aaaab, \ldots\}$

Grammar $G = (\{S, A, B\}, \{a, b, \epsilon\}, S, P)$

Production Rule (P) : $S \rightarrow AAB$

$A \rightarrow \epsilon \mid a$

$B \rightarrow \epsilon \mid bB \mid aBb$

iii) $L = \{a^n b^m \mid 2n \leq m \leq 3n\}$

$L = \{\epsilon, abb, abbb, aabbbb, aabbbbb, \ldots\}$

Production Rule (P): $S \rightarrow aSbb \mid aSbbb \mid \epsilon$

Grammar $G = (\{S\}, \{a, b, \epsilon\}, S, P)$

__Ques 4.__ Let $\{a^n b^n \mid n \geq 0\}$. Show that L^2 is context free.

$L = \{\epsilon, ab, aabb, aaabbb \ldots\}$

$L^2 = \{\epsilon, ab, aabb, aaabbb \ldots\} \times \{\epsilon, ab, aabb, aaabbb, \ldots\}$

$= \{\epsilon\epsilon, \epsilon ab, \epsilon aabb, \epsilon aaabbb, ab\epsilon, abab \ldots\}$

$= \{\epsilon, ab, aabb, aaabbb, ab, abab, \ldots\}$

Grammar, $G = (\{S\}, \{a, b, \epsilon\}, S, P)$ for L, where

P = production rules $\Rightarrow S \rightarrow aSb \mid \epsilon$

Then, for L^2 : $P' \Rightarrow S \rightarrow AB$

$A \rightarrow aAb \mid \epsilon$

$B \rightarrow aAb \mid \epsilon$

$G = (\{S, A, B\}, \{a, b, \epsilon\}, S, P')$

$\therefore L^2$ is a CFG.

__Ques 5.__ Consider the grammar $G = (V, T, E, P)$ with $V = \{E, I\}$

$T = \{a, b, c, +, *, (,)\}$ and productions $P \Rightarrow$

$E \rightarrow I$, $E \rightarrow E * E$, $I \rightarrow a \mid b \mid c$,

$E \rightarrow E + E$, $E \rightarrow (E)$

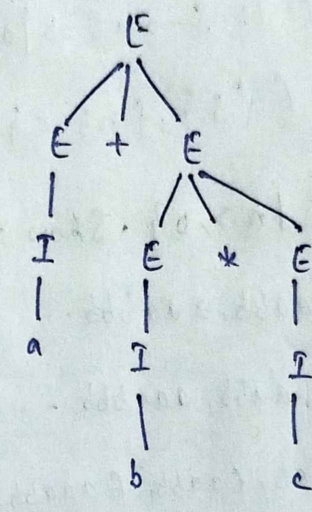
Show that the grammar is ambiguous for the string a+b*c.

Ans **Ambiguous Grammar**

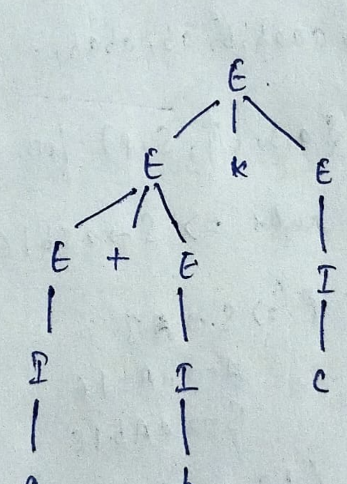
If for a grammar there are more than 1 derivation tree is possible, then it is called an ambiguous grammar.

$w = a+b*c$.

Derivation tree 1



Derivation Tree 2



Ques 6. Find an S-grammar for $L = \{a^n b^n \mid n \geq 1\}$

Ans A simple grammar (S-grammar) is one in which every production is of the form:

$A \rightarrow a B_1 B_2 \ldots B_n$ where $a \in$ Terminals (T)

$n \geq 0$

$B_i \ (i \geq 1) \in$ Variables (V)

S- grammar for L : $S \rightarrow aSB \mid aB$

$B \rightarrow b$

Ques 7. Show that a regular language can not be inherently ambiguous.

A regular grammar generates regular languages which can be recognised by deterministic finite automata (DFA).

Inherent ambiguity means that every DFA recognising the language has multiple distinct accepting paths for at least one string in the language.

For regular languages, DFAs are always unambiguous. This is because in a DFA, for any given input string there's always exactly one path that the automata can take.

Therefore, regular grammar cannot be inherently ambiguous.

Ques 8. Show that the two grammars:

G_1 : $S \rightarrow abAB \mid ba$; G_2 : $S \rightarrow abAaA$

$A \rightarrow aaa$; $S \rightarrow abAbb \mid ba$

$B \rightarrow aA \mid bb$; $A \rightarrow aaa$

are equivalent

G_1 : $S \rightarrow abAB \mid ba$

$A \rightarrow aaa$

$B \rightarrow aA \mid bb$

$\Rightarrow$ $S \rightarrow abAaA \mid abAbb \mid ba$

$A \rightarrow aaa$ by putting B

after putting A : $S \rightarrow abaaaaaaa$

$S \rightarrow abaaabb$

$S \rightarrow ba$

Now,

G_2 $S \rightarrow abAaA$

$S \rightarrow abAbb \mid ba$

$A \rightarrow aaa$

after putting A $\Rightarrow$ $S \rightarrow abaaaaaaa$

$S \rightarrow abaaabb$

$S \rightarrow ba$

As, we can see from above expansion G_1 and G_2 are equivalent

Ques 8. Remove all unit-productions, all useless productions, and all ϵ-productions from the grammar.

$S \to aA \mid aBB$
$A \to aaA \mid \epsilon$
$B \to bB \mid bbC$
$C \to B$

What does this grammar generate?

- Removing unit-productions:

Unit productions: $C \to B$

To remove unit-productions we will replace left-hand side variable in other production rules with right hand variable

So, $S \to aA \mid aBB$
$A \to aaA \mid \epsilon$
$B \to bB \mid bbC$
$C \to B$

$\xrightarrow[\text{removal}]{\text{after}}$

$S \to aA \mid aBB$
$A \to aaA \mid \epsilon$
$B \to bB \mid bbB$

- Removing all useless productions

Useless production: • If it doesn't produce terminal or non-reachable from start symbol.
• If it doesn't contribute to generate any terminal string.

In this case, useless productions: $B \to bB \mid bbB$

So, $S \to aA \mid aBB$
$A \to aaA \mid \epsilon$
$B \to bB \mid bbB$

$\xrightarrow[\text{removal}]{\text{after}}$

$S \to aA$
$A \to aaA \mid \epsilon$

- Removing ϵ Productions

To remove ϵ-production, we replace variable producing ϵ with ϵ to create new string and write that string along with initial production.

So,
$$\begin{aligned} S &\rightarrow aA \\ A &\rightarrow aaA \\ A &\rightarrow \epsilon \end{aligned} \xrightarrow[\text{removal}]{\text{after}} \begin{aligned} S &\rightarrow aA \mid a \\ A &\rightarrow aaA \mid aa \end{aligned}$$

This grammar is producing strings of a's of odd length and string 'aa'

i.e. CFL = { a, aa, aaa, aaaaa, aaaaaaa, ... }

Ques 10. Convert the grammar : $S \rightarrow aSb \mid bSa \mid a \mid b$ into GNF.

GNF : Greibach Normal Form

In GNF, production rule has the form:

$A \rightarrow a\alpha$ where, $A \in$ Variable
$a \in$ Terminal
$\alpha \in V^*$

Conversion

CFG	GNF CFG
$S \rightarrow aSb$	$S \rightarrow aSB$
$S \rightarrow bSa$	$S \rightarrow bSA$
$S \rightarrow a$	$S \rightarrow a$
$S \rightarrow b$	$S \rightarrow b$
	$B \rightarrow b$
	$A \rightarrow a$