Unit-3

1) context free Grammars
2) Left most and right most derivations
3) parse tree
4) Ambigious Grammars
5) Simplification of CFG
   → elimination of useless symbols
   - elimination of ε production
   - elimination of unit production
6) Normal forms
   → chomsic Normal form
   → Greibach Normal forms
7) pumping lemma
8) closure properties
9) Applications of CFG

① Context free grammar:- A CFG is formally defined as four tuple $G = \{V, T, P, S\}$
where V = set of finite set of variables or non terminals
T = finite set of terminals
P = set of productions
S = starting symbol
Here each production of the form $\alpha \to \beta$ where $\alpha \in V$ and $\beta \in (V \cup T)^*$

Derivation:- Deriving the input string from non terminals using production is called derivation

eg:- $A \to 1AO$

$A \to B$

$B \to \#$

Given Grammar CFG, $G = \{V, T, P, S\}$

$V = \{A, B\}$, $T = \{1, 0, \#\}$,

$S = \{A\}$, $P = \{A \to 1AO, A \to B, B \to \#\}$

Derive input string $W = \{111\#000\}$

Sol:- $A \to 1AO$ $\{\because A \to 1AO\}$

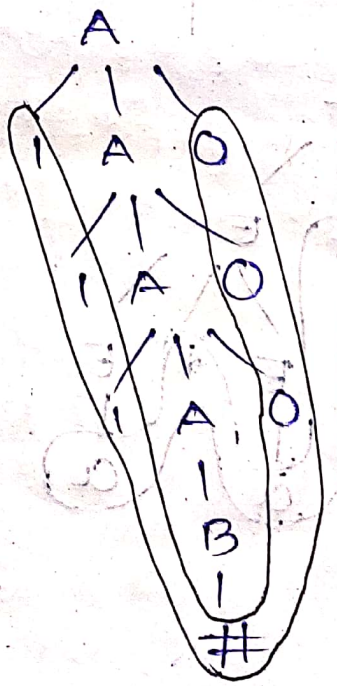
$A \to 11AOO$ $\{\because A \to 1AO\}$

$\to 111AOOO$ $\{A \to B\}$

$\Rightarrow 111BOOO$ $\{B \to \#\}$

$\Rightarrow 111\#000$

Parse tree:- A tree representation for the derivation based on the given productions



Left most derivation: A derivation is said to be left most if in each step left most variable can be replaced.

Right most derivation: A derivation is said to be Right most if in each step right most variable can be replaced.

Illustrate LMD & RMD for the following

Grammar
$E \to E+E$
$E \to E*E$
$E \to a$

Input string
$W = a+a*a$

Grammar $G = \{V, T, P, S\}$

$V = \{E\}$, $T = \{+, *, a\}$, $S = \{E\}$

LMD

$E \to \underline{E}+E$ {$\because E \to E+E$}

$\to a+\underline{E}$ {$\because E \to E*E$}

$\to a+\underline{E}*E$ {$\because E \to a$}

$\to a+a*\underline{E}$ {$\because E \to a$}

$\to a+a*a$

RMD

$E \to E+\underline{E}$ {$E \to E*E$}

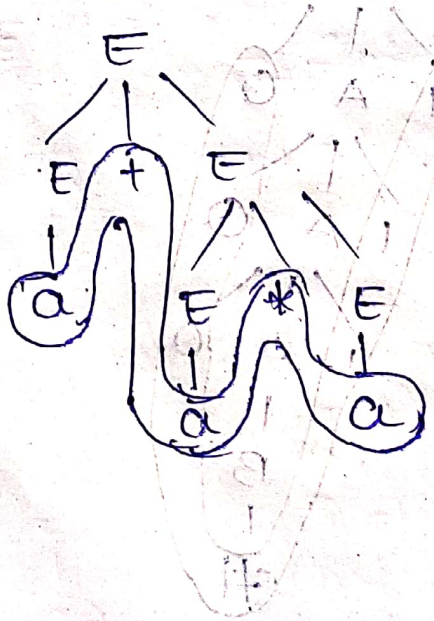
$\to E+E*\underline{E}$ {$E \to a$}

$\to E+\underline{E}*a$ {$E \to a$}

$\to E+a*a$ {$E \to a$}

$\to a+a*a$

parse tree:



Consider the following data draw the parse tree in leftmost and right most derivation and derive the input string w = ababaaba

$S \to SbS \mid a$

The Given Grammar $G = \{V, T, P, S\}$

$V = \{S\}$, $T = \{a, b\}$,

$P = \{S \to SbS \mid a\}$

$S = S$

Left most Derivation

$S \to SbS$

$\to abS$

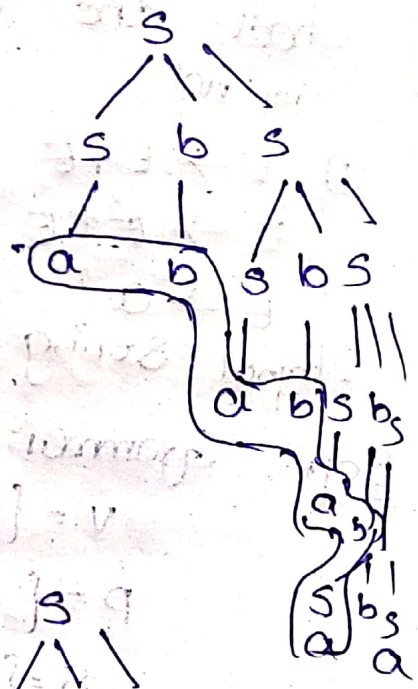
$\to abSbS \to ababS$

$\to ababSbS$

$\to abababS$

$\to ababaaba$

parse tree



Right most Derivation

$S \to SbS$

$\to SbSba$

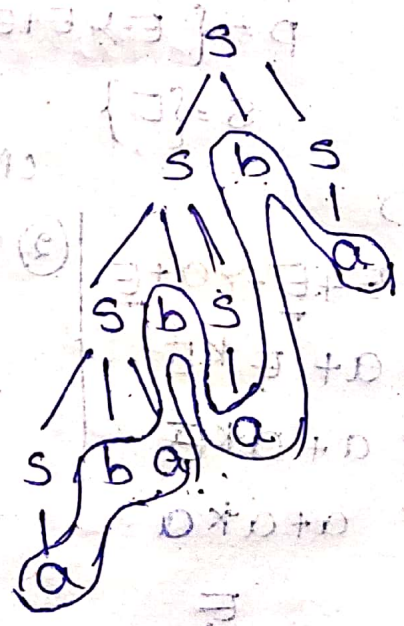
$\to SbSbaba$

$\to Sbabaaba$

$\to SbSbaaba$

$\to SbabSbaaba$

$\to ababaaba$



Ambigaious Grammar: A Grammar is said to be ambigious if it produces more than one parse trees (either left most (or) Right most), otherwise the grammar is said to be unambigiow

In the above example we have two left most parse trees & two right most parse trees for the same string, so that grammar is ambigious Grammar.

check the following Grammar is ambigiow or not

① $E \to E+E$
$E \to E*E$
$E \to a$

Input string $w = a+a*a$

Sol:- Grammar $G = \{V, T, P, S\}$

$V = \{E\}$, $T = \{a\}$

$P = \{E \to E+E, E \to E*E, E \to a\}$

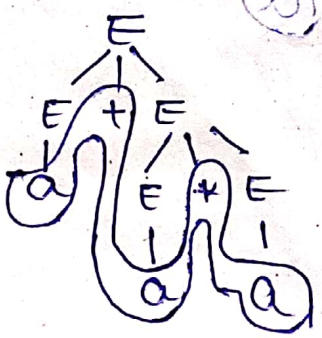
$S = \{E\}$

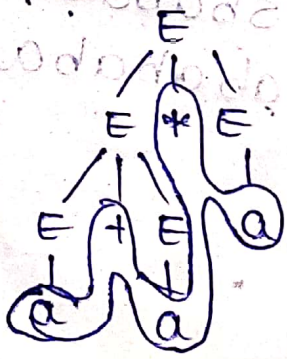
LMD

① $E \to E+E \to a+E$
$\to a+E*E$
$\to a+a*E$
$\to a+a*a$

LMD

② $E \to E*E$
$\to E+E*E \to a+E*E$
$\to a+a*a \to a+a*E$





The given grammar is ambiguous because it generates more than one parse trees.

② Derive the string aabbabba using in left most and Right most derivations using following CFG

Sol $S \to aB/ba$

$A \to a/as/bAA$

$B \to b/bs/aBB$

Sol Grammar $G = \{V, T, P, S\}$

$V = \{S, A, B\}$, $T = \{a, b\}$

$P = \{S \to aB/ba, A \to a/as/bAA, B \to b/bs/aBB\}$

$S = \{S\}$

$W = \{aabbabba\}$

$S \to aB$ - ①

$S \to ba$ - ②

$A \to a$ - ③

$A \to aS$ - ④

$A \to bAA$ - ⑤

$B \to b$ - ⑥

$B \to bS$ - ⑦

$B \to aBB$ - ⑧

LM.D

$S \to aB$

$\to aaBB$ ⑧

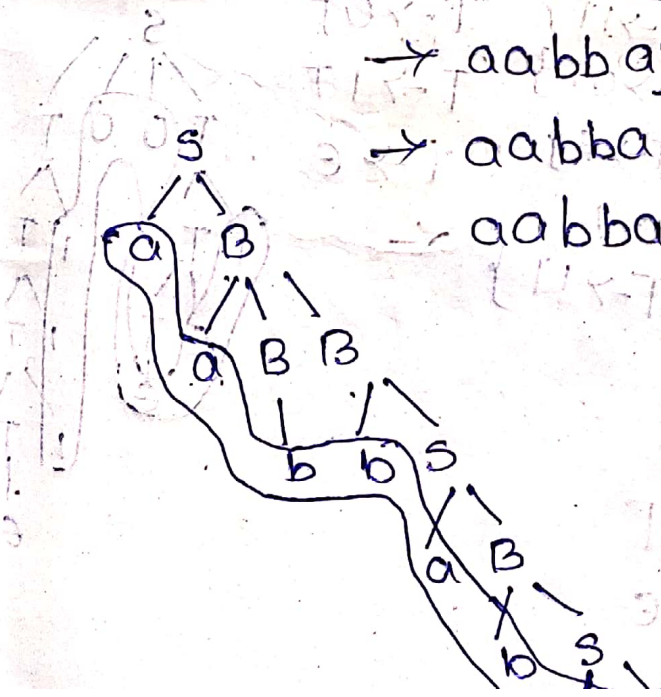
$\to aabB$ ⑥

$\to aabbS$ ⑦

$\to aabbaB$ ①

$\to aabbabS$ ⑦

$\to aabbabba$ ②



R.M.D

$S \to aB$ - ①

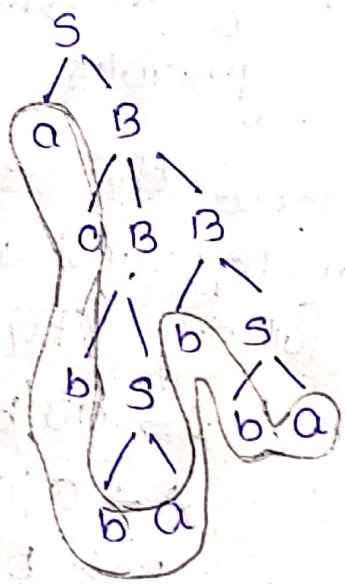
$\to aaBB$ - ③

$\to aaBbS$ - ⑦

$\to aaBbba$ - ⑤

$\to aabSbba$ - ⑦

$\to aabbabba$ - ②



Derive the string 1000111 for the following Grammar and draw the parse tree

$S \to TOOT$

$T \to 0T \mid 1T \mid \epsilon$

Sol: Grammar $G = \{V, T, P, S\}$

$V = \{S, T\}$, $T = \{0, 1\}$,

$P = \{S \to TOOT,\ T \to 0T \mid 1T \mid \epsilon\}$

$S = \{S\}$

$S \to TOOT$

$S \to 1TOOT$ $\{T \to 1T\}$

$\to 10TOOT$ $\{T \to 0T\}$

$\to 1000T$ $\{T \to \epsilon\}$

$\to 1000 1T$

$\to 10001 1T$

$\to 100011T$

$\to 1000111T$

$\to 1000111\epsilon$

$\to 1000111$

$S \to TOOT$

$T \to 0T$

$T \to 1T$

$T \to \epsilon$

Simplification of CFG (or) minimizing the Context free Grammars.

(i) Use less symbol: A symbol is useless if it cannot derive a terminal. (or) it is not reachable from start state

Eg: 1 $S \to AB|0A$
$S \to BC|AB$
$A \to a$
$C \to aB|b$

$S \to AB$
$S \to 0A$
$S \to 0a$ ✓

$S \to AB$
$\to aBX$
useless symbol

The useless symbol is 'B' So, we can eliminate productions which are included with "B"

$\therefore$ $S \to 0A$, $A \to a$

Eg (2) $S \to aS/A|BC$
$A \to a$
$B \to aa$
$C \to acb$

$S \to aS$
$\to aA$
$\to aa$ ✓

$S \to A$
$\to a$

$S \to BC$
$\to aac$
$aaacb$ ✗

$S \to aS$
$\to aBC$
$\to aaacb$

The use less Symbol is 'X' So, we can eliminate productions which are included with "C"

$\therefore$ $S \to aS/A$, $A \to a$, $B \to aa$

Not reachable from the start state

$S \to A|B$
$A \to a$
$B \to AS$
$S \to b$
$D \to C$

Sol:- $S \to AB$
$\to aB$
$\to aAS$
$\to aAb$
$\to aab$

D is welen.

Convert the following Grammar on GNF

$S \to AB$

$A \to BSB$

$A \to a$

$B \to b$

step-1: There is no ϵ productions, useless symbols, and unit productions. So Automatically the given grammar is Simplified itself.

step-2: Transform the Grammar G into CNF form (i.e) $A \to BC$

$A \to a$

CNF

$S \to AB$ ✓

$A \to BSB$ $\{\because SB = K\}$

$A \to BK$

$K \to SB$

$A \to a$

$B \to b$

step-3: Transform the grammer G into an equivalent GNF (i.e) $\begin{cases} A \to aX \\ \text{or} \\ A \to a \end{cases}$

step-4: In the given Grammar G,

Variables (V) = {S, A, K, B}

Rename Variables

$V_1 = S$, $V_2 = A$, $V_3 = K$, $V_4 = B$

Replace New Variables in the Productions

$V_1 \to V_2 V_4$

$V_2 \to V_4 V_3$

$V_3 \to V_1 V_4$

$V_2 \to a$

$V_4 \to b$

Condition: L subscript > R subscript

As per the condition Consider

$V_3 \rightarrow V_1 V_4$

Substitute V_1 in above production

$V_3 \rightarrow V_2 V_4 V_4$

$V_3 \rightarrow aV_4 V_3 V_4 V_4 \mid aV_4 V_4$ ✓

$V_3 \rightarrow V_4 V_3 V_4 V_4$

$V_3 \rightarrow bV_3 V_4 V_4$ ✓

(2) $V_1 \rightarrow V_2 V_4$

$V_1 \rightarrow V_4 V_3 V_4 \mid aV_4$ ✓

$V_1 \rightarrow bV_4 V_3 V_4$

$V_1 \rightarrow bV_3 V_4$ ✓

(3) $V_2 \rightarrow V_4 V_3$

$V_2 \rightarrow bV_3$ ✓

Final set of productions

$V_1 \rightarrow bV_3 V_4 \mid aV_4$

$V_2 \rightarrow bV_3 \mid a$

$V_3 \rightarrow bV_3 V_4 V_4 \mid aV_4 V_4$

$V_4 \rightarrow b$.

(2) Find a GNF Grammar Equivalent to the following CFG

$S \rightarrow AA \mid 0$

$A \rightarrow SS \mid 1$

step-1: check whether the given Grammer has ∈ productions, useless symbols and unit production.

→ The given Grammer is Simplified itself

step-2: Transform the given Grammar int CNF form (ie) $A \to BC$, $A \to a$

CNF → $S \to AA/0$

$A \to SS/1$

The given Grammer is in CNF form

step-3: Transform the Grammar G into an equivalent GNF

(i.e) $\{ A \to aX, (or) A \to a \}$

step-4: In the given Grammer G

Variables (V) = {S, A}

Rename Variables $\{V_1 = S;\ V_2 = A\}$

Replace $V_1 \to V_2V_2/0$

$V_2 \to V_1V_1/1$

condition: Lsub → Rsub

As per condition. consider

$V_2 \to V_1V_1/1$

$V_2 \to V_2V_2V_1/0V_1/1$

$\{A \to A\alpha/B_1/B_2\}$ left recursive grammer ∴ Ambigous.

$\Rightarrow \{ A \to B/BB,\ B \to \alpha/\alpha B \}$

$V_2 \to 0V_1 / 1 / 0V_1B / 1B$ — new

$B \to V_2V_1 / V_2V_1B$

Consider $B \to V_2V_1 / V_2V_1B$

$B \to 0V_1V_1 / 1V_1 / 0V_1BV_1 / 1BV_1$

$B \to 0V_1V_1B / 1V_1B / 0V_1BV_1B / 1BV_1B$

Consider $V_1 \to V_2V_2 / 0$

Substitute V_2 in above production.

$V_1 \to 0V_1V_2 / 1V_2 / 0V_1BV_2 / 1BV_2 / 0$

Pumping lemma for CFL:

Pumping lemma is used for to prove the given language is CFL or not

LEMMA: Let L be any context free language then $\exists$ a constant n, which depends only upon L. such that $\exists$ a string $w \in L$ and $|w| \geq n$ where $w = pqrst$ such that

1. $|qs| \geq 1$
2. $|prs| \leq n$
3. $\forall i \geq 0$, $w = pq^i rs^i t$ is in CFL

① prove that $L = \{a^n b^n c^n \mid n \geq 0\}$ is not CFL

Sol Let us assume that the language is CFL

the given language $L = \{abc, aabbcc, aaabbbccc, \}$

let W be any string such that $W \in L$

Let $W = pqrst$ and $|qs| \geq 1$ & $|prs| \leq n$

by pumping lemma Consider $W = pq^i rs^i t$,

that means additional occurances of qs

Consider the various cases

case i: take $i = 0$, $W = pq^i rs^i t \Rightarrow pq^0 rs^0 t$

if $i = 0$ then q^0 and s^0 Means qs are absent then $W = prt$

Eg: $W = aabbcc$ (p q r s t)

$|bc| = 2 \geq 1$ ✓

$|prs| = 4 \leq 6$ ✓

$W = pq^i rs^i t \{i = 0\}$

$W = prt$

$= aabc \notin L$

case 2: take $i = 2$

$W = pq^i rs^i t$

$= aab^2bc^2c$

$= aabbbbccc \notin L$.

Hence our assumption of L being CFL is wrong

Hence, this proves that the given language L is not a CFL.

FOR - CFL:

Ex $W = aabbcc$ (p q r s t)

$\Rightarrow pq^2rs^2t$

$= aaab(bc)^2c$

$\Rightarrow a^3b^3c^3$

Show that the following language is not CFL $L = \{ wxw \mid w \in (a,b)^* \}$

Sol: Given $L = \{ WWW \mid w \in (a,b)^* \}$