UNIT-II GRAMMAR.

Grammar Introduction:

A grammar of a language (G) is defined as,

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

V - finite set of objects called Variables (Non-terminal)

T - finite set of objects called Terminals

$S \in V$ - start symbol

P - finite set of productions.

Types of Grammar:

- Type 0 grammars / unrestricted grammars (Recursively Enumerable Language)
- Type 1 grammars / context sensitive grammars
- Type 2 grammars / Context free grammars
- Type 3 grammars / Regular grammar.

1) Type 0 grammars

- No restrictions on the production rules
- Production rule is of the form

$$\alpha \rightarrow \beta \mid \alpha \neq \beta$$

Where $\alpha, \beta \rightarrow$ can be strings composed by terminals and non-terminals.

- This grammar can be modeled using Turing Machine.

2) Type 1 grammars

- Context sensitive grammar (CSG):
- Production rule is of the form

$$\alpha A \beta \Rightarrow \alpha \gamma \beta$$

Here $A \rightarrow$ Non-terminal symbol

$\alpha, \beta, \gamma \rightarrow$ combination of terminals and non-terminals.

- This grammar can be modeled using linear bounded automata.

3) Type 2 grammar:

- Content Free Grammar (CFG).
- production rule is of the form $A \to \alpha$

    A - Non terminal Symbol

    α - Terminal or non-terminal symbol.

- This grammar can be modeled using push down automata.

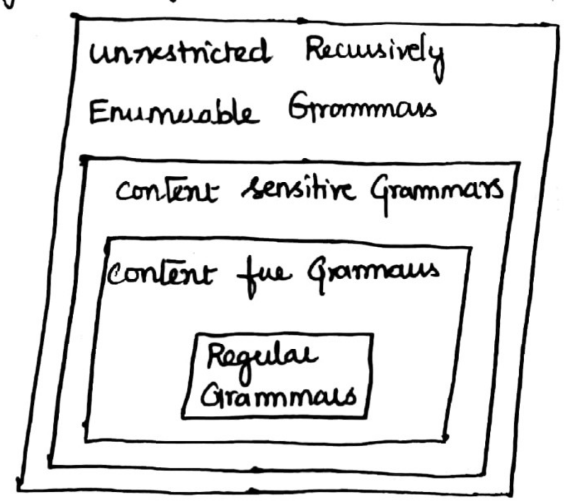
4) Type 3 grammar:

- Regular grammar that describe regular/formal languages.
- production rule consist of,
    - only one terminal at the left hand side.
    - Right hand side having a single terminal and may or may not be followed by non terminals.

    $A \to a, \quad A \to aB$

- This grammar can be modeled using finite automata.

Chomsky hierarchy.



$\Rightarrow$ Regular grammars
$\subseteq$
Content free grammars
$\subset$
Content sensitive grammars
$\subset$
Unrestricted grammars.

Content free Language and Grammars.

Definition: The content free grammar can be formally defined as a set denoted by $G = (V, T, P, S)$ where V and T are set of non-terminals and terminals respectively

P is set of production rules. $NT \to NT$
$NT \to T$

S is a start symbol.

Example:

$P = \{ S \to S+S$
$S \to S*S$
$S \to (S)$
$S \to 4 \}$

Syntax of any English statement is,

SENTENCE → NOUN VERB

NOUN → Rama / Seeta / Gopal

VERB → goes / Writes / sings.

Dervied strings: "Rama sings"

Problems:

1) Contruct the CFG for the regular expression $(0+1)^*$.

solution: $CFG = (V, T, P, S)$

$= (\{S\}, \{0,1\}, P, S)$

where $P = \{ S \to 0S \mid 1S$
$S \to \epsilon \}$

Example:

$(0+1)^* = \{\epsilon, 0, 1, 01, 10, 00, 11, \cdots\}$

2) Construct CFG for the language L which has all the strings which are all palindrome over $\Sigma = \{a,b\}$

Solution: $G = (\{S\}, \{a,b\}, P, S)$

$S \to aSa \mid bSb \mid a \mid b, \epsilon$

Example: abaaba

$S \to a\underline{S}a$
$\to ab\underline{S}ba$
$\to aba\underline{S}aba$
$\to aba\epsilon aba$
$\to abaaba$

3) Construct CFG for $\{0^m 1^n \mid 1 \le m \le n\}$

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

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

$P = \{ S \to 0S1 \mid 0A \mid 01$
$A \to 1A \mid 1 \}$

Example: 00111

$S \to 0\underline{S}1$
$\to 00\underline{S}1$
$\to 001\underline{A}1$
$\to 00111$

4) Construct CFG for $L = \{a^m b^n c^P \mid m+n = P \text{ and } P \ge 1\}$

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

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

$P = \{ S \to aSc \mid bAc \mid ac \mid bc$
$A \to bc \mid bc$

5) Consider the alphabet $\Sigma = \{a, b, (,), +, *, \cdot, /, \ldots \epsilon\}$. Construct a CFG that generates all strings in E^* that are RE over $\{a, b\}$.

Solution: $E \rightarrow E+E$

$E \rightarrow E*E$

$E \rightarrow E \cdot E$

$E \rightarrow E/E$

$E \rightarrow a|b|\epsilon$.

Problems: Grammar to Language

1) If $S \rightarrow aSb | aAb$, $A \rightarrow bAa$, $A \rightarrow ba$ is a CFG then determine CFL.

Solution:

$S \rightarrow a\underline{S}b$

$\rightarrow aa\underline{S}bb$

$\rightarrow aaa\underline{A}bbb$

$\rightarrow \underbrace{aaa}_{a^n}ba\underbrace{bbb}_{b^n}$

$L(G) = \{a^n b^m a^m b^n \mid m, n \geq 1\}$

2) If $S \rightarrow aSa | bSb | \epsilon$ is CFG. Find $L(G)$.

Solution: $S \rightarrow a\underline{S}a$

$\rightarrow aa\underline{S}aa$

$\rightarrow aab\underline{S}baa$

$\rightarrow \underbrace{aab}_{W}\underbrace{baa}_{W^R}$

$\therefore L(G) = \{WW^R \mid W \in (a, b)^*\}$

3) Find the context free language for the following grammars.

(i) $S \rightarrow aSbS | bSaS | \epsilon$

(ii) $S \rightarrow aSb | ab$.

Solution

(i) $S \rightarrow a\underline{S}bS$

$\rightarrow ab\underline{S}aSbS$

$\rightarrow aba\underline{S}bS$

$\rightarrow abab\underline{S}$

$\rightarrow abab$

$S \rightarrow b\underline{S}aS$

$\rightarrow ba\underline{S}$

$\rightarrow ba$

$\therefore$ L containing equal number of a's and b's

(ii) $S \rightarrow aSb | ab$

$S \rightarrow a\underline{S}b$

$\rightarrow aa\underline{S}bb$

$\rightarrow aaabbb$

L containing equal number of a's followed by by equal number of b's

$L = \{a^n b^n \mid n \geq 1\}$

Derivations, Ambiguity, Derivation tree

Derivations: Use the productions from head to body (ie) from start symbol expanding till reaches the given string.

Two types of derivations are,

(i) Left Most Derivation (LMD)

(ii) Right Most Derivation (RMD)

⇒ LMD is a derivation in which the leftmost non-terminal is replaced first from the sentential form.

(ie) $S \overset{*}{\underset{lm}{\Longrightarrow}} \alpha$, then α is left sentential form.

⇒ RMD is a derivation in which rightmost non-terminal is replaced first from the sentential form.

(ie) $S \overset{*}{\underset{rm}{\Longrightarrow}} \alpha$, then α is right sentential form.

Derivation tree (parse tree)

- It is a graphical representation for the derivation of the given production rules for a given CFG.

Properties

i) Root node is always a node indicating start symbol.

ii) Derivation is read from left to right

iii) Leaf nodes are always terminal nodes.

iv) Interior nodes are always non-terminal nodes.

Example:

Consider the grammar G has the production

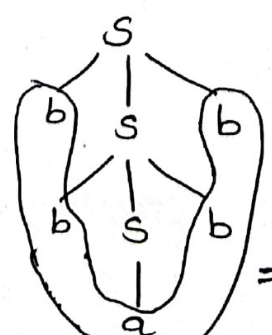
$S \rightarrow bSb \mid a \mid b$ and string "bbabb"

Derivation:

$S \rightarrow b\underline{S}b$

$\rightarrow bb\underline{S}bb$

$\rightarrow bbabb$



⇒ bbabb

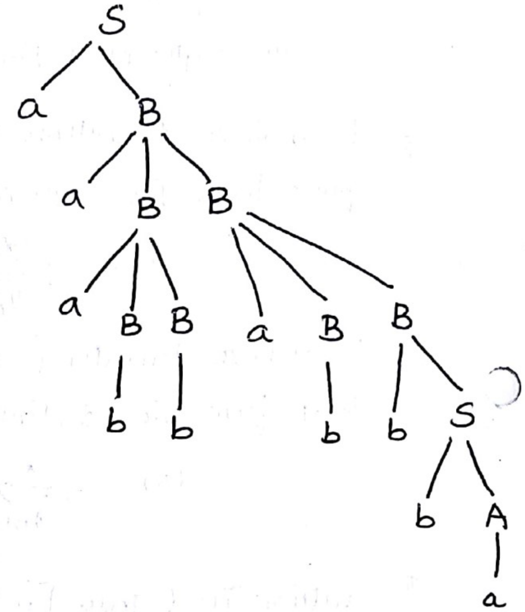
Problems 1: Construct the derivation tree for the string "aaabbabbba" using LMD and RMD. using $S \to aB \mid bA$, $A \to a \mid aS \mid bAA$, $B \to b \mid bS \mid aBB$

Solution:

LMD:

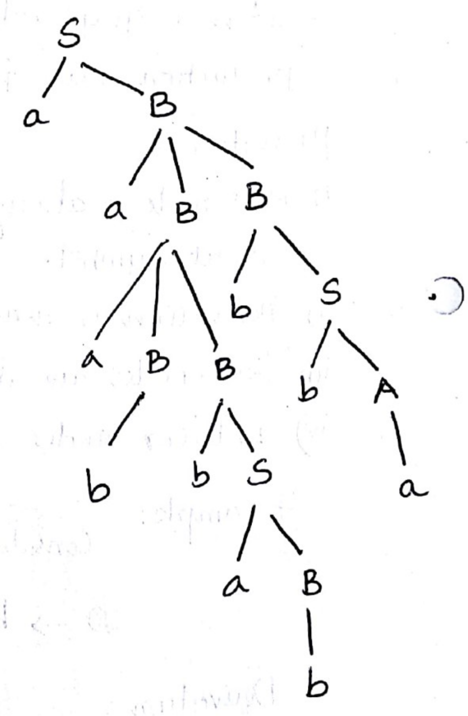
$S \Rightarrow aB$
$\Rightarrow aaBB \quad (B \to aBB)$
$\Rightarrow aaaBBB \quad (B \to aBB)$
$\Rightarrow aaabBB \quad (B \to b)$
$\Rightarrow aaabbB \quad (B \to b)$
$\Rightarrow aaabbaBB \quad (B \to aBB)$
$\Rightarrow aaabbabB \quad (B \to b)$
$\Rightarrow aaabbabbS \quad (B \to bS)$
$\Rightarrow aaabbabbbA \quad (S \to bA)$
$\Rightarrow aaabbabbba \quad (A \to a)$

Parse tree:



RMD:

$S \Rightarrow aB$
$\Rightarrow aaBB \quad (B \to aBB)$
$\Rightarrow aaBbS \quad (B \to bS)$
$\Rightarrow aaBbbA \quad (S \to bA)$
$\Rightarrow aaBbba \quad (A \to a)$
$\Rightarrow aaaBBbba \quad (B \to aBB)$
$\Rightarrow aaaBbSbba \quad (B \to bS)$
$\Rightarrow aaaBbaBbba \quad (S \to aB)$
$\Rightarrow aaaBbabbba \quad (B \to b)$
$\Rightarrow aaabbabbba \quad (B \to b)$



Scanned by CamScanner

Pbm 2: Write a grammar G to recognize all prefix expressions involving all binary arithmetic operators. Construct the parse tree for the sentence '$-*+abc/de$'.

Solution:

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

$V = \{S, A, B, C, D, E\}$

$T = \{+, -, *, /, a, b, c, d, e\}$

S is a start symbol.

productions are,

$S \rightarrow A | B | C | D | E$

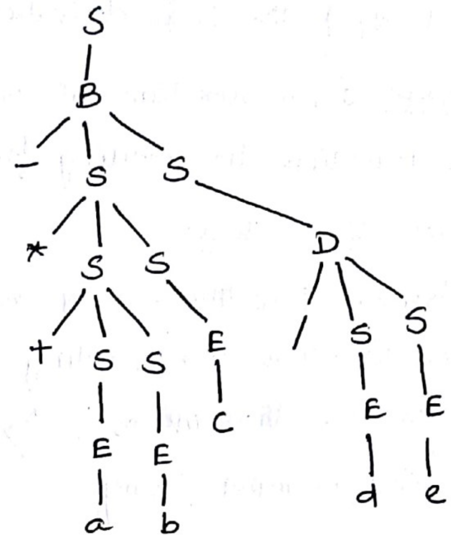
$A \rightarrow +SS$

$B \rightarrow -SS$

$C \rightarrow *SS$

$D \rightarrow /SS$

$E \rightarrow a | b | c | d | e$



Ambiguity:

If there exists more than one parse trees for a given grammar, that means there could be more than one leftmost or rightmost derivation possible and then that grammar is said to be ambiguous grammar.

Problems:

1) The CFG is given by $G = (V, T, P, S)$ where $V = \{E\}$, $T = \{id\}$, $S = \{E\}$

$P = \{E \rightarrow E+E, E \rightarrow E*E, E \rightarrow id\}$. Is the grammar ambiguous?

Solution:

Consider the string $id*id+id$.

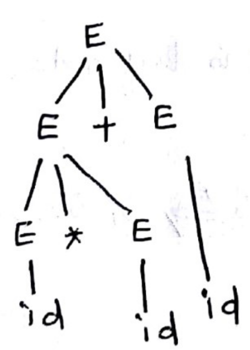
$E \Rightarrow E+E$

$\Rightarrow E*E+E$

$\Rightarrow id*E+E$

$\Rightarrow id*id+E$

$\Rightarrow id*id+id$



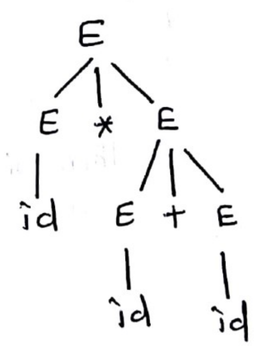
$E \Rightarrow E*E$

$\Rightarrow E*E+E$

$\Rightarrow id*E+E$

$\Rightarrow id*id+E$

$\Rightarrow id*id+id$



Here, we obtain two different parse tree for the string $id*id+id$.

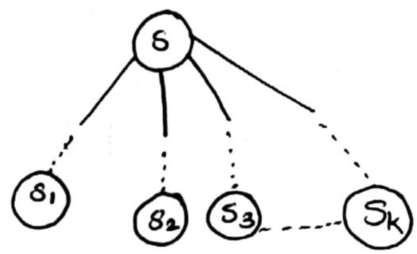
$\therefore$ The given grammar is ambiguous.

Relationship between derivation and derivation trees.

Theorem: Let $G = (V, T, P, S)$ be a context free grammar. Then $S \overset{*}{\Rightarrow} \alpha$ if and only if there is a derivation tree in grammar G which gives the string a.

Proof: For a non-terminal S there exists $S \overset{*}{\Rightarrow} W$ if and only if there is a derivation tree starting from root S and yielding W.

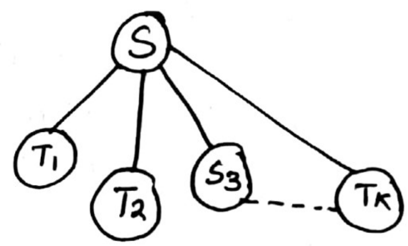
Basis of induction:



Assume that there is only one interior node S. The derivation tree yielding $S_1, S_2, S_3 \ldots S_n$. From S is that means $S \overset{*}{\Rightarrow} S_1, S_2 \ldots S_n \overset{*}{\Rightarrow} a$ is input string.

Induction hypothesis:

⇒ We assume that for k-1 nodes the derivation tree can be drawn. We then prove that for k vertices also we can have a derivation tree.



⇒ That means the input string can be derived as $S \rightarrow S_1 S_2 S_3 \ldots S_k$.

⇒ There are two cases,

(i) S_i may be a leaf variable

(ii) S_i may be an interior node yielding a.

⇒ The S derives 'a' by fewer number of k steps then $a \in S_1 S_2 S_3 S_4 \ldots S_K$.

If $a_i = S_i$ then S_i is leaf node (terminal) and if $S_i \overset{*}{\Rightarrow} a_i$ then S_i is an interior node.

This proves that $S \overset{*}{\Rightarrow} S_1, S_2, S_3, \ldots S_n \overset{*}{\Rightarrow} a$ can be obtained.

Simplification of CFG.

Simplification of grammar means reduction of grammar by removing useless symbols.

- Elimination of useless symbols.
- Elimination of unit productions.
- Elimination of Null production(ϵ).

Elimination of useless symbols.

Any symbol is useful when it appears on the right hand side, in the production rule and generates some terminal string. If no such derivation exists then it is supposed to be an useless symbol.

Example: Eliminate the useless symbol from the following grammar.

$S \rightarrow aS | A | C$
$A \rightarrow a$
$B \rightarrow aa$
$C \rightarrow aCb$

Solution: Production with terminal symbols are

$A \rightarrow a$
$B \rightarrow aa$

Start symbol $S \rightarrow aS | A | c$

Here, there is no production for B

$\therefore$ B is useless symbol.

III^ly $S \rightarrow C \rightarrow aCb \rightarrow aaCbb \rightarrow aaaCbbb \rightarrow \cdots$

Here, no terminating symbol for C.

$\therefore$ C is useless symbol.

$\therefore$ Eliminate B and C, we get

$$\boxed{\begin{array}{l} S \rightarrow aS | A \\ A \rightarrow a \end{array}}$$

Elimination of ε production:

If there is ε production, remove it, without changing the meaning of the grammar.

Example: Eliminate ε-productions from the CFG.

$A \rightarrow 0B1 \mid 1B1$

$B \rightarrow 0B \mid 1B \mid \varepsilon$

Solution:

$A \rightarrow 0B1 \mid 1B1$

$B \rightarrow \varepsilon$, $A \rightarrow 01 \mid 11$

$B \rightarrow 0B \mid 1B$

$B \rightarrow \varepsilon$, $B \rightarrow 0 \mid 1$

∴ After Elimination,

$A \rightarrow 0B1 \mid 1B1 \mid 01 \mid 11$

$B \rightarrow 0B \mid 1B \mid 0 \mid 1$

Removing Unit productions

The unit productions are the productions in which one non-terminal gives another non-terminal.

$X \rightarrow Y$

Example: Eliminate the unit production from following grammar,

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow C \mid b$

$C \rightarrow D$

$D \rightarrow E \mid bC$

$E \rightarrow d \mid Ab$

Here, $B \rightarrow C$, $C \rightarrow D$, $D \rightarrow E$ are unit productions.

∴ $D \rightarrow E \mid bc$ can be written as $D \rightarrow d \mid Ab \mid bc$

III^ly $C \rightarrow E \mid bc$, B becomes $B \rightarrow d \mid Ab \mid bc \mid b$.

After removing unit productions.

$S \rightarrow AB$ $\quad B \rightarrow d \mid Ab \mid bC \mid b$ $\quad D \rightarrow d \mid Ab \mid bC$ ✗

$A \rightarrow a$ $\quad C \rightarrow d \mid Ab \mid bC$ $\quad E \rightarrow d \mid Ab.$ ✗

Chomsky normal form (CNF)

A context free grammar $G = (V, T, P, S)$ is said to be in CNF if each production in G is of the form

[$NT \rightarrow NT.NT$
$NT \rightarrow T$]

$X \rightarrow YZ$

$X \rightarrow \alpha$, where $X, Y, Z \in V$, and $\alpha \in T$.

Problems:

1) Convert the given CFG to CNF $S \rightarrow aSa \mid bSb \mid a \mid b$

Solution:

Productions are

$S \rightarrow aSa$

$S \rightarrow bSb$

$S \rightarrow a$

$S \rightarrow b$

Here, the productions which are already in CNF is

$S \rightarrow a$

$S \rightarrow b$

Apply CNF rule to other productions,

$S \rightarrow aSa$ $\quad | \; C_a \rightarrow a$

$S \rightarrow C_a S C_a$

$$\boxed{\begin{aligned} S &\rightarrow C_a A \\ A &\rightarrow S C_a \end{aligned}}$$

$S \rightarrow bSb$ $\quad | \; C_b \rightarrow b$

$S \rightarrow C_b S C_b$

$$\boxed{\begin{aligned} S &\rightarrow C_b B \\ B &\rightarrow S C_b \end{aligned}}$$

The resultant productions are,

$S \rightarrow C_a A \mid C_b B \mid a \mid b$

$A \rightarrow S C_a$

$B \rightarrow S C_b$

$C_a \rightarrow a$

$C_b \rightarrow b$

2) Reduce the following grammar to Chomsky normal form.

$S \rightarrow a \mid AAB$

$A \rightarrow ab \mid aB \mid \epsilon$

$B \rightarrow aba \mid \epsilon$.

Solution:

Productions are,

$S \rightarrow a$

$S \rightarrow AAB$

$A \rightarrow ab$

$A \rightarrow aB$

$A \rightarrow \epsilon$

$B \rightarrow aba$

$B \rightarrow \epsilon$

Here, Eliminate ϵ production.

$A \rightarrow \epsilon$ and $B \rightarrow \epsilon$

$\Rightarrow$ After elimination, productions are

$S \rightarrow a \mid AAB \mid AA \mid AB \mid B$

$A \rightarrow ab \mid aB \mid a$

$B \rightarrow aba$

$\Rightarrow$ Eliminate unit production.

$S \rightarrow a \mid AAB \mid AA \mid AB \mid aba$

$A \rightarrow ab \mid aB \mid a$

$B \rightarrow aba$

Productions already in CNF is

$S \rightarrow a$

$S \rightarrow AA$

$S \rightarrow AB$

$A \rightarrow a$

Apply CNF rules to other productions,

$S \rightarrow \underbrace{AA}_{S}B$

$\boxed{S \rightarrow SB}$

$S \rightarrow aba$

$S \rightarrow C_a \underbrace{C_b C_a}_{C_1}$

$C_a \rightarrow a$

$C_b \rightarrow b$

$\boxed{\begin{array}{l} S \rightarrow C_a C_1 \\ C_1 \rightarrow C_b C_a \end{array}}$

$\boxed{\begin{array}{l} A \rightarrow C_a C_b \\ A \rightarrow C_a B \end{array}}$

$B \rightarrow C_a \underbrace{C_b C_a}_{C_1}$

$\boxed{B \rightarrow C_a C_1}$

$\therefore$ The resultant productions are,

$S \rightarrow SB \mid C_a C_1 \mid AA \mid AB \mid a$

$C_1 \rightarrow C_b C_a$

$A \rightarrow C_a C_b \mid a \mid C_a B$

$B \rightarrow C_a C_1$

3) ~~Ex~~. Convert the given CFG to CNF.

$S \to aB$ $\quad$ $A \to bAA$
$S \to bA$ $\quad$ $B \to b$
$A \to a$ $\quad$ $B \to bS$
$A \to aS$ $\quad$ $B \to aBB$

Solution:

Productions already in CNF is,
$A \to a$
$B \to b$

Apply CNF rules to other productions.
$C_a \to a$, $C_b \to b$

$S \to C_a B$
$S \to C_b A$
$A \to C_a S$
$A \to C_b \underbrace{AA}_{C_1}$
$A \to C_b C_1$
$C_1 \to AA$
$B \to C_b S$
$B \to C_a \underbrace{BB}_{C_2}$
$B \to C_a C_2$
$C_2 \to BB$

The resultant productions are,

$S \to C_a B$
$S \to C_b A$
$A \to C_a S$
$A \to a$
$A \to C_b C_1$
$C_1 \to AA$
$B \to b$
$B \to C_b S$
$B \to C_a C_2$
$C_2 \to BB$

4) Convert the grammar with productions into CNF
$A \to bAB \mid \lambda$, $B \to BAa \mid \lambda$.

5) Convert the grammar $S \to AB \mid aB$, $A \to aab \mid \epsilon$, $B \to bbA$ into CNF.

6) Convert to chomsky Normal Form.

$S \to A \mid CB$ $\quad$ $B \to 1B \mid 1$ $\quad$ $D \to 2D \mid 2$
$A \to C \mid D$ $\quad$ $C \to 0C \mid 0$

Greibach Normal Form (GNF)

A grammar $G=(V,T,P,S)$ is said to be in GNF if every production rule is of the form.

$$X \rightarrow a\alpha$$

where $a \in T$, $X \in V$

$\alpha \in V^*$

Right hand side of every productions starts with a terminal, followed by a string of variables of zero/more length.

Problems:

1) Convert the given CFG to GNF

$S \rightarrow ABA$

$A \rightarrow aA \mid \epsilon$

$B \rightarrow bB \mid \epsilon$.

Solution:

Simplify the CFG, Eliminate ϵ production $A \rightarrow \epsilon$, $B \rightarrow \epsilon$.

$S \rightarrow ABA \mid AB \mid BA \mid AA \mid A \mid B$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

Eliminate unit productions,

$S \rightarrow ABA \mid AB \mid BA \mid AA \mid aA \mid a \mid bB \mid b$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$.

Apply GNF rules,

$S \rightarrow \underline{A}BA$

$S \rightarrow aABA \mid aBA$

$S \rightarrow \underline{A}B$

$S \rightarrow aAB \mid aB$

$S \rightarrow BA$

$S \rightarrow bBA \mid bA$

$S \rightarrow AA$

$S \rightarrow aAA \mid aA$

∴ The resultant productions are,

$S \rightarrow aABA \mid aBA \mid aAB \mid aB \mid bBA \mid bA$

$S \rightarrow aAA \mid aA \mid a \mid b$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

2) Convert given CFG to GNF where $V = \{S, A\}$, $T = \{0, 1\}$ and P is,

$$S \to AA \mid 0$$
$$A \to SS \mid 1$$

Solution:

Replace S as A_1 and A as A_2

CFG becomes, $A_1 \to A_2 A_2 \mid 0$

$A_2 \to A_1 A_1 \mid 1$

Start with $A_2 \to A_1 A_1 \mid 1$

$$\underbrace{A_2 \to A_2}_{\text{left recursion.}} A_2 A_1 \mid 0 A_1 \mid 1$$

Introduce B_2, to eliminate left recursion $\Big\} \Rightarrow$ $A_2 \to A_2 A_2 A_1$, $A_2 \to 0 A_1 \mid 1$

$B_2 \to A_2 A_1 \mid A_2 A_1 B_2$, $A_2 \to 0 A_1 B_2 \mid 1 B_2$

The productions are

$A_2 \to 0 A_1 \mid 1$

$A_2 \to 0 A_1 B_2 \mid 1 B_2$ $\Rightarrow$ $A_1 \to A_2 A_2 \mid 0$

$A_1 \to 0 A_1 A_2 \mid 1 A_2 \mid 0 A_1 B_2 A_2 \mid 1 B_2 A_2$ (GNF)

$B_2 \to A_2 A_1$

$B_2 \to 0 A_1 \mid 1 A_1 \mid 0 A_1 B_2 A_1 \mid 1 B_2 A_1$ (GNF)

$B_2 \to A_2 A_1 B_2$

$B_2 \to 0 A_1 A_1 B_2 \mid 1 A_1 B_2 \mid 0 A_1 B_2 A_1 B_2 \mid 1 B_2 A_1 B_2$ (GNF).

Converting Back $A_1 = S$ and $A_2 = A$

$S \to 0SA \mid 1A \mid 0SB_2A \mid 1B_2A \mid 0$

$A \to 0S \mid 1 \mid 0SB_2 \mid 1B_2$

$B_2 \to 0SS \mid 1S \mid 0SB_2S \mid 1B_2S$

$B_2 \to 0SSB_2 \mid 1SB_2 \mid 0SB_2SB_2 \mid 1B_2SB_2$

3) Convert the grammar $S \to AB$, $A \to BS | b$, $B \to SA | a$ into Greibach Normal form. (9)

Solution:

Consider the grammar,

$$S \to AB, \quad A \to BS | b, \quad B \to SA | a$$

Assume $S = A_1$, $A = A_2$, and $B = A_3$.

$$A_1 \to A_2 A_3$$

$$A_2 \to A_3 A_1 | b$$

$$A_3 \to A_1 A_2 | a$$

Consider, $A_3 \to A_1 A_2 | a$

$$A_3 \to A_2 A_3 A_2 | a$$

$$A_3 \to A_3 A_1 A_3 A_2 | b A_3 A_2 | a$$

Now,

$$\underbrace{A_3 \to A_3}\, A_1 A_3 A_2, \quad A_3 \to b A_3 A_2 | a$$

Introduce B_3 to eliminate left recursion $\Rightarrow$ $B_3 \to A_1 A_3 A_2 | A_1 A_3 A_2 B_3$

$$A_3 \to b A_3 A_2 | a | b A_3 A_2 B_3 | a B_3 . \text{ (GNF)}$$

$$A_2 \to A_3 A_1 | b$$

$$A_2 \to b A_3 A_2 A_1 | a A_1 | b A_3 A_2 B_3 A_1 | a B_3 A_1 | b \text{ (GNF)}$$

$$A_1 \to A_2 A_3$$

$$A_1 \to b A_3 A_2 A_1 A_3 | a A_1 A_3 | b A_3 A_2 B_3 A_1 A_3 | a B_3 A_1 A_3 | b A_3$$

$$B_3 \to A_1 A_3 A_2$$

$$\to b A_3 A_2 A_1 A_3 A_3 A_2 | a A_1 A_3 A_3 A_2 | b A_3 A_2 B_3 A_1 A_3 A_3 A_2 |$$

$$a B_3 A_1 A_3 A_3 A_2 | b A_3 A_3 A_2 .$$

$B_3 \rightarrow A_1 A_3 A_2 B_3$

$\rightarrow bA_3A_2A_1A_3A_3A_2B_3 \mid aA_1A_3A_3A_2B_3 \mid bA_3A_2B_3A_1A_3A_3A_2B_3 \mid$

$aB_3A_1A_3A_2B_3 \mid bA_3A_3A_2B_3.$

$\therefore$ The resultant productions are,

$A_1 \rightarrow bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3A_2B_3A_1A_3 \mid bA_3$

$A_2 \rightarrow bA_3A_2A_1 \mid aA_1 \mid bA_3A_2B_3A_1 \mid aB_3A_1 \mid b$

$A_3 \rightarrow bA_3A_2 \mid a \mid bA_3A_2B_3 \mid aB_3$

$B_3 \rightarrow bA_3A_3A_2bA_3A_2A_1A_3A_2 \mid aA_1A_3A_3A_2 \mid$

$bA_3A_2B_3A_1A_3A_3A_2 \mid aB_3A_1A_3A_3A_2 \mid$

$bA_3A_2A_1A_3A_3A_2B_3 \mid aA_1A_3A_3A_2B_3 \mid$

$bA_3A_2B_3A_1A_3A_3A_2B_3 \mid aB_3A_1A_3A_2B_3 \mid bA_3A_3A_2B_3.$