Lecture Notes
on
Theory of Automata and Formal Languages

Unit-3

Dharmendra Kumar
(Associate Professor)
United College of Engineering and Research, Prayagraj

May 10, 2023

Contents

1 Context Free Grammar (CFG) 5
1.1 Definition . . . 5
1.2 Derivation Tree . . . 5
1.3 Left Most Derivation . . . 5
1.4 Right Most Derivation . . . 5
1.5 Some Examples . . . 6
1.6 Ambiguity in Grammar and Language . . . 6
1.7 Inherent Ambiguity . . . 7
1.8 Simplification of Context Free Grammar . . . 7
1.8.1 Active Variable . . . 7
1.8.2 Reachable symbols . . . 7
1.8.3 Useful Variable . . . 7
1.8.4 Construction a Grammar in which all the variables are active (Elimination of Non-Active Variables from a Grammar) . . . 7
1.8.5 Construction a Grammar in which all the symbols are reachable (Elimination of Non-Reachable Symbols from a Grammar) . . . 8
1.8.6 Construction of Reduced Grammar . . . 9
1.9 Elimination of null productions . . . 9
1.10 Elimination of Unit Productions . . . 10
1.11 Chomosky Normal Form (CNF) . . . 10
1.11.1 Definition . . . 10
1.11.2 Reduction into Chomosky Normal Form . . . 10
1.12 Greibach Normal Form (GNF) . . . 11
1.12.1 Definition . . . 11
1.12.2 Lemma-1 . . . 11
1.12.3 Lemma-2 . . . 11
1.12.4 Reduction to Greibach Normal Form . . . 12
1.13 Exercise . . . 12
1.14 Closure Properties of Context Free Languages . . . 13
1.15 Pumping Lemma for Context Free Languages . . . 14
1.16 Exercise . . . 15
1.17 Decision Properties of Regular and Context Free Languages . . . 15
1.18 AKTU Examination Questions . . . 16

Chapter 1

Context Free Grammar (CFG)

1.1 Definition

A grammar is said to be context free grammar if all the production rules of the grammar are of the following form:-
$A \rightarrow \alpha$ where $\alpha \in (V \cup \Sigma)^*$ and $A \in V$

1.2 Derivation Tree

A tree is said to be derivation tree if it satisfies the following properties:-

1. All the nodes of the tree are labeled by variable, terminal or ϵ symbol.

2. The root node of the tree has labeled S (Starting symbol of the grammar).

3. All the internal nodes have labeled variable symbol.

4. All the leaf nodes have labeled terminal symbol or ϵ symbol.

5. If A $\rightarrow$ X_1X_2..........X_n be a production rule used in the derivation of the string, then in the tree, A will be at the parent node and $X_1, X_2,$ ...., X_n will be at the children of this node A.

1.3 Left Most Derivation

A derivation A$\overset{*}{\Rightarrow}$w is said to be left most derivation if we apply the production rule in the derivation at the left most variable in every step.

1.4 Right Most Derivation

A derivation A$\overset{*}{\Rightarrow}$w is said to be right most derivation if we apply the production rule in the derivation at the right most variable in every step.

1.5 Some Examples

Example: Consider the following grammar

S→ 0B/1A
A→ 0/0S/1AA
B→ 1/1S/0BB

For the string 00110101, find the left most derivation, right most derivation and derivation tree.

Example: Consider the following grammar

S→ AA
A→ a/bA/Ab/AAA

Find parse tree for the string bbaaaab.

Example: Consider the following grammar

S→ aAS/a
A→ SbA/SS/ba

Find derivation tree for the string aabbaa.

1.6 Ambiguity in Grammar and Language

Ambiguous String
A string w $\in$ L(G) is said to be ambiguous string if there exists more than one derivation for the string.

Ambiguous Grammar
A grammar G is said to be ambiguous if there exists some string w $\in$ L(G) for which more than one derivation tree are possible.

Example: Consider the following grammar:-

S→S+S/S*S/a/b

Is this grammar ambiguous?
Solution:

Example: Consider the grammar G,

S→SbS/a.

Show that grammar G is ambiguous.
Solution:

Example: Consider the following grammar:-

S→ a/abSb/aAb
A→ bS/aAAb

Is this grammar ambiguous?
Solution:

Example: Consider the following grammar:-

S→ aB/ab
A→ aAB/a
B→ ABb/b

Is this grammar ambiguous?
Solution:

1.7 Inherent Ambiguity

- If L is a context free language for which there exists an unambiguous grammar, then L is said to be unambiguous.
- If every grammar that generates L is ambiguous, then the language is said to be inherently ambiguous.

Example: Following language is inherent ambiguous
L = $\{a^n b^n c^m d^m !n \geq 1, m \geq 1\} cup\ \{a^n b^m c^m d^n !ngeq1, m \geq 1\}$

1.8 Simplification of Context Free Grammar

Simplification of grammar means to remove the useless symbols (variables and terminals) and production rules from the grammar.

1.8.1 Active Variable

A variable A is said to be active if it derives a terminal string i.e
A→ x , where x∈ Σ^*

1.8.2 Reachable symbols

A symbol is said to be reachable if it appears in a string derives from starting symbol of the grammar i.e.
If S ⇒ X, then every symbols in X are said to be reachable.

1.8.3 Useful Variable

A variable is said to be useful if it is active and reachable both.

1.8.4 Construction a Grammar in which all the variables are active (Elimination of Non-Active Variables from a Grammar)

Suppose the given context free grammar is
G = (V,Σ,S,P)
Now we construct a context free grammar G' in which all the variables are active.

G' = (V',Σ, S, P')
Step-1: Determination of V'
Let A_i denote the set of active variables. A_i is determined recursively as following:-
A_1 = {A ∈ V ! if A→ $x \in P$, where $x \in \Sigma^*$ }
$A_2 = A_1 \cup$ {A ∈ V ! if A→ $\alpha \in P$, where $\alpha \in (A_1|cup\Sigma)^*$ }
....
....
$A_{i+1} = A_i \cup$ {A ∈ V ! if A→ $\alpha \in P$, where $\alpha \in (A_i|cup\Sigma)^*$ }
Repeat this process until $A_{i+1} = A_i$
Now, we terminate this process.
Now, V' = A_i
Step-2: Determination of P'
P' is obtained from P by removing those production rules in which non-active variables belong.

Example: Consider the grammar
G = ({S,A,B,C,E}, {a,b,c}, P, S)
where P = { S→AB, A→a, B→b, B→C, E→c/ϵ
Eliminate the non-active variables from this grammar.

1.8.5 Construction a Grammar in which all the symbols are reachable (Elimination of Non-Reachable Symbols from a Grammar)

Suppose the given context free grammar is
G = (V,Σ,S,P)
Now we construct a context free grammar G' in which all the symbols are reachable.
G' = (V',Σ', S, P')
Step-1: Determination of V' and ,Σ'
Let R_i denote the set of reachable symbols. R_i is determined recursively as following:-
R_1 = {S }
$R_2 = R_1 \cup$ {x ! A→ $\alpha \in P$, where $A \in R_1$ and $\alpha \in (V \cup \Sigma)^*$ and α contains x}
....
....
$R_{i+1} = R_i \cup$ {x ! A→ $\alpha \in P$, where $A \in R_i$ and $\alpha \in (V \cup \Sigma)^*$ and α contains x}
Repeat this process until $R_{i+1} = R_i$
Now, we terminate this process.
Now, V' = $R_i \cap V$
Now, Σ' = $R_i \cap \Sigma$
Step-2: Determination of P'
P' is obtained from P by removing those production rules in which non-reachable symbols belong.

Example: Consider the grammar
G = ({S,A,B,C,E}, {a,b,c}, P, S)
where P = { S→AB, A→a, B→b, B→C, E→c/ϵ
Eliminate the non-reachable symbols from this grammar.

1.8.6 Construction of Reduced Grammar

Reduced Grammar: A grammar is said to be reduced grammar if all the symbols and production rules are useful.

Procedure:
Suppose the given grammar is G.
This methods consists of two steps. They are:-
Step-1: Find the grammar G' equivalent to G in which all the variables are active.
Step-2: Find the grammar G" equivalent to G' which includes only reachable symbols.
This grammar G" is a reduced grammar equivalent to G.

Example: Find the reduced grammar equivalent to the following grammar
S $\rightarrow$ AB/CA
B $\rightarrow$ BC/AB
A $\rightarrow$ a
C $\rightarrow$ aB/b

Example: Reduce the following grammar
S $\rightarrow$ aAa
A $\rightarrow$ Sb/bCC/DaA
C $\rightarrow$ abb/DD
E $\rightarrow$ aC
D $\rightarrow$ aDA

1.9 Elimination of null productions

Null Production: A production rule is said to be null production if it is of the following form:-

$A \rightarrow \epsilon$

Nullable Variable: A variable A is said to be nullable if it derives empty string i.e.

A$\Rightarrow \epsilon,\ \forall A \in V$

Procedure:
Consider a grammar G = (V, Σ, S, P). Let G' is a grammar having no null productions such that L(G') = L(G)-{ϵ}.

G' is constructed as following:-

G' = (V, Σ, S, P')

Step-1: Determination of the set of nullable variables
Let W_i is the set of nullable variables. W_i is calculated as following:-
W_1 = {A ! A$\rightarrow \epsilon \ \in P$}
W_2 = $W_1\cup${A ! $\exists$ a production $A \rightarrow \alpha \ \in P$and $\alpha \in W_1^*$}
....
....

$W_{i+1} = W_i \cup \{$A ! $\exists$ a production $A \rightarrow \alpha\ \in P$and $\alpha \in W_i^*\}$
Repeat this process until $W_{i+1} = W_i$
Now, we terminate this process.
Step-2: Determination of P'
(i) We add the production rules of P into P' whose RHS does not include any nullable variable.
(ii) Consider the remaining production rules of P.
If A$\rightarrow$ X_1X_2..............$X_n \in P$ then we add A$\rightarrow$ $\alpha_1\alpha_2$..............α_n into P', where $\alpha_i = X_i$ or ϵ but not all α_i equal to ϵ if X_i is nullable variable otherwise put $\alpha_i = X_i$.

Example: Consider the following grammar
S$\rightarrow$ aS/AB
A$\rightarrow$ ϵ
B$\rightarrow$ ϵ
D$\rightarrow$ b
Eliminate the null productions.

1.10 Elimination of Unit Productions

Unit Production: A production rule is said to be unit production if it is of the following form:-
A$\rightarrow$B, where A, B $\in$ V.

Example: Consider the following grammar
S$\rightarrow$ AB
A$\rightarrow$a
B$\rightarrow$C/b
C$\rightarrow$ D
D$\rightarrow$ E
E$\rightarrow$ a
Eliminate the unit productions.

1.11 Chomosky Normal Form (CNF)

1.11.1 Definition

A grammar G is said to be in Chomosky nomal form if every prodution rules are of the following form:-

A$\rightarrow$BC or A$\rightarrow$ a,

where A,B,C $\in$V and a $\in \Sigma$.

1.11.2 Reduction into Chomosky Normal Form

Step-1: Elimination of null productions and unit productions
In this step, we eliminate the null production and unit productions from the grammar. Let the resultant grammar is G=(V,Σ,S,P).
Step-2: Elimination of terminals from RHS if rule is not in CNF

Let $G_1 = (V_1, \Sigma, S, P_1)$ be the grammar obtained in this step.
All the productions of P which are in CNF, must also belong into P_1 and all the variables of V must also belong into V_1.
Consider the remaining production rules.
Add the new variables into V_1 equal to the number of terminals on the right hand side of these production rules. And if the terminals are $a_1, a_2, ..., a_n$, then add variables $X_1, X_2, ..., X_n$ into V_1 and add $X_1 \to a_1$, $X_2 \to a_2$,................,$X_n \to a_n$ rules into P_1.
Step-3: Restricting the number of variables on the RHS
Let $G_2 = (V_2, \Sigma, S, P_2)$ be the grammar obtained in this step.
Add all the elements of V_1 into V_2. Add the production rules of P_1 into P_2 which are in CNF.
Consider those production rules of P_1 which are not in CNF. These production rules contains at least three elements on the right hand side.
Consider production rules $A \to X_1X_2$...........X_n, where n≥3. Then we add n-2 variable into V_2. Let these are $Y_1, Y_2, ..., Y_{n-2}$. Add production rules $A \to X_1Y_1$, $Y_{\to}X_2Y_2$,...........,$Y_{n-2} \to X_{n-1}X_n$ to P_2.
Now, the grammar G_2 is the resultant grammar which is in CNF.

Example: Reduce the following grammar into CNF:-
(1) S→aAD, A→aB/bAB, B→b, D→d.
(2) S→aAbB, A→a/aA, B→b/bB.
(3) S→˜S/[S⊃)S]/p/q.

1.12 Greibach Normal Form (GNF)

1.12.1 Definition

A CFG is said to be in Greibach normal form if all the production rules are of the following form:

A→ $a\gamma$, where $\gamma \in V^*$, a $\in \Sigma$ and A∈V.

1.12.2 Lemma-1

Let A→ $B\gamma$ be an A-production and let B-productions are B→ $\beta_1|\beta_2|$...........$|\beta_n$. Then A→ $B\gamma$ production is replaced by the following rules:-
A→ $\beta_1\gamma|\beta_2\gamma|$..........$|\beta_n\gamma$.

1.12.3 Lemma-2

Let the set of A-productions be A→ $A\alpha_1|A\alpha_2|$............$|A\alpha_m|\beta_1|\beta_2|$.......$|\beta_n$ (β_i's do not start with A). Then we replace the rules A→ $A\alpha_1|A\alpha_2|$............$A\alpha_m$ by the following procedure:-
Add new variable Z to the grammar. And the set of A-production are
A→ $\beta_1|\beta_2|$.......$|\beta_n$
A→ $\beta_1Z|\beta_2Z|$.......$|\beta_nZ$
The set of Z-productions are
Z→ $\alpha_1|\alpha_2|$.......$|\alpha_m$

Z→ $\alpha_1 Z|\alpha_2 Z|.......|\alpha_m Z$

1.12.4 Reduction to Greibach Normal Form

Step-1: Construct the given grammar into CNF. Next, we rename the variables as $A_1, A_2, ..., A_n$ with S = A_1.
Step-2: Consider the production rules which are of the following form
$A_i \rightarrow A_j\gamma$, where j¡i and $\gamma \in V^*$
Apply the lemma-1 in these rules until j≥i.
Step-3: Consider the production rules which are of the following form
$A_i \rightarrow A_j\gamma$, where j = i and $\gamma \in V^*$
Apply the lemma-2 in these rules until j≥i.
Step-4: Consider the production rules which are not in GNF. Apply lemma-1 in all these rules. After this, the resultant grammar will be in GNF.

Example: Convert the following grammar into GNF.

1. S→AB, A→BS/b, B→SA/a.
2. S→AA/a, A→SS/b.
3. E→E+T/T, T→T*F/F, F→(E)/a,
4. S→ABb/a, A→aaA, B→bAb.
5. S→SS, S→0S1/01, B→SA/a.

1.13 Exercise

1. Eliminate the useless production from the following grammar
   S→a/aA/B/C, A→aB/ϵ, B→aA, C→cCD, D→ddd,.
2. Eliminate all the ϵ-productions from the following grammar:-
   S→AaB/aaB, A→ ϵ, B→bbA/ϵ
3. Remove all unit productions, all useless productions and all ϵ-productions from the grammar
   S→aA/aBB, A→aaA/ϵ, B→bB/bbC, C→B.
4. Transform the following grammar into CNF
   S→abAB, A→bAB/ϵ, B→BAa/A/ϵ .
5. Transform the following grammar into CNF
   S→AB/aB, A→aab/ϵ, B→bbA .

1.14 Closure Properties of Context Free Languages

Theorem: Show that the family of context free languages is closed under union operation.
Proof: Let L_1 and L_2 be two context free languages generated by context free grammar $G_1 = (V_1, \Sigma_1, S_1, P_1)$ and $G_1 = (V_2, \Sigma_2, S_2, P_2)$ respectively.
Now, we construct the grammar G as the following;-
$G = (V, \Sigma, S, P)$
Where, V = $V_1 \cup V_2 \cup \{S\}$
$\Sigma = \Sigma_1 \cup \Sigma$
and P = $P_1 \cup P_2 \cup \{S \to S_1 | S_2\}$
Now, we have to show that
L(G) = $L(G_1) \cup L(G_2)$
Let x $\in$ L(G) $\Leftrightarrow$ S $\overset{*}{\Rightarrow}$ x
$\Leftrightarrow$ S $\Rightarrow S_1 \overset{*}{\Rightarrow}$ x or
S $\Rightarrow S_2 \overset{*}{\Rightarrow}$ x or
$\Leftrightarrow S_1 \overset{*}{\Rightarrow}$ x or $S_2 \overset{*}{\Rightarrow}$ x
$\Leftrightarrow$ x$\in L(G_1)$ or x$\in L(G_2)$
$\Leftrightarrow$ x$\in L(G_1) \cup L(G_2)$
Therefore, L(G) = $L(G_1) \cup L(G_2)$
Clearly, G is a CFG for $L_1 \cup L_2$, therefore $L_1 \cup L_2$ is also a context free language.

Theorem: Show that the family of context free languages is closed under concatenation operation.
Proof: Let L_1 and L_2 be two context free languages generated by context free grammar $G_1 = (V_1, \Sigma_1, S_1, P_1)$ and $G_1 = (V_2, \Sigma_2, S_2, P_2)$ respectively.
Now, we construct the grammar G as the following;-
$G = (V, \Sigma, S, P)$
Where, V = $V_1 \cup V_2 \cup \{S\}$
$\Sigma = \Sigma_1 \cup \Sigma$
and P = $P_1 \cup P_2 \cup \{S \to S_1.S_2\}$
Now, we have to show that
L(G) = $L(G_1).L(G_2)$
Let x $\in$ L(G) $\Leftrightarrow$ S $\overset{*}{\Rightarrow}$ x
$\Leftrightarrow$ S $\Rightarrow S_1.S_2 \overset{*}{\Rightarrow}$ x
$\Leftrightarrow S_1 \overset{*}{\Rightarrow} x_1$ and $S_2 \overset{*}{\Rightarrow} x_2$ (Let x= x_1x_2)
$\Leftrightarrow x_1.x_2 \in L(G_1).L(G_2)$
$\Leftrightarrow$ x$\in L(G_1).L(G_2)$
Therefore, L(G) = $L(G_1).L(G_2)$
Clearly, G is a CFG for $L_1.L_2$, therefore $L_1.L_2$ is also a context free language.

Theorem: Show that the family of context free languages is closed under kleene closure operation.
Proof: Let L is a context free languages and G is a context free grammar generating L.
$G = (V, \Sigma, S, P)$
Now, we construct a grammar G' as the following:-
$G' = (V', \Sigma, S', P')$
Where, V' = V$\cup${S'}

P' = P$\cup\{S' \to SS'|\epsilon\}$
Now, we have to show that L(G') = (L(G))*.
Let x $\in$ L(G') $\Leftrightarrow$ S' $\overset{*}{\Rightarrow}$ x
$\Leftrightarrow$ S' $\Rightarrow S.S.S.........S(n - times) \overset{*}{\Rightarrow}$ x
$\Leftrightarrow S \overset{*}{\Rightarrow} x_1,\ S \overset{*}{\Rightarrow} x_2\ ....S \overset{*}{\Rightarrow} x_n$ (Let x= $x_1x_2....................x_n$)

$\Leftrightarrow\ x_1 \in L(G),\ \ x_2 \in L(G),\ ....,\ x_n \in L(G)$

$\Leftrightarrow\ x_1.x_2...........x_n \in (L(G))^n$
$\Leftrightarrow\ x \in (L(G))^*\ \ (L(G))^n \subseteq (L(G))^*$

Therefore, L(G') = (L(G))*.
Therefore, G' is a grammar generating the language L*. Hence L* is a context free language.

Theorem: Show that the family of context free languages is not closed under intersection operation.
Proof: Consider the two context free languages:-
$L_1 = \{a^n b^n c^m\ !\ n \geq 0, m \geq 0\}$
$L_2 = \{a^n b^m c^m\ !\ n \geq 0, m \geq 0\}$
The intersection of these languages is
$L_1 \cap L_2 = \{a^n b^n c^n\ !\ n \geq 0\}$
This language is not context free.
Therefore, the family of context free languages is not closed under intersection operation.

Theorem: Show that the family of context free languages is not closed under complement operation.
Proof: Consider the following formula

$L_1 \cap L_2 = \ \overline{(\bar{L_1} \cup \bar{L_2})}$ ....(1)
Suppose the context free languages are closed under complement operation.
From (1), If L_1 and L_2 are context free languages, then $\bar{L_1}$ and $\bar{L_2}$ are also context free language.
Since $\bar{L_1}$ and $\bar{L_2}$ are context free, therefore $\bar{L_1} \cup \bar{L_2}$ will also be context free language.
Therefore $\overline{(\bar{L_1} \cup \bar{L_2})}$ ias also context free.
Since the R.H.S. of equation (1) is context free, therefore L.H.S. is also context free. But, by previous theorem $L_1 \cap L_2$ is not context free, therefore context free languages is not closed under complement operation.

1.15 Pumping Lemma for Context Free Languages

Let L be an infinite context free language. Then there exists some positive integer n such that any x$\in$ L with x$\geq$n, can be decomposed as
x=uvwxy
with $\mid vxy \mid \leq n$ and $\mid vx \mid \geq 1$,

such that
$uv^iwx^iy \in$ L, $\forall$ i = 0,1,2,3,.......................

Application: This lemma is used to show a given language is not context free.

Example: Show that the language
L = { $a^nb^nc^n$ | $n \geq 0$}
is not context free.
Solution:

1.16 Exercise

1. L = {ww ! w∈{a,b}*}
2. L = {$a^{n!}$! $n \geq 0$}
3. L = {a^nb^j ! $n = j^2$}
4. L = {a^{n^2} ! $n \geq 0$}
5. L = {a^p ! p is a prime number}
6. L = {$a^nb^jc^k$! $k > n,\ k > j$}

1.17 Decision Properties of Regular and Context Free Languages

Theorem: Given a context free grammar G=(V,Σ,S,P), there exists an algorithm for deciding whether or not L(G) is empty.
Proof: For the simplicity, we assume that $\epsilon \notin L(G)$. We use the algorithm for removing useless symbols and productions. If S is found to be useless, then L(G) is empty otherwise L(G) contains at least one element.

Theorem: Given a context free grammar G=(V,Σ,S,P), there exists an algorithm for deciding whether or not L(G) is infinite.
Proof: We assume that G contains no $\epsilon - poductions$, no unit-productions, and no useless symbols.
Convert the grammar into CNF.
We draw a directed graph whose vertices are variables in G. If A→BC is a production, then there are directed edges from A to B and A to C.
L is finite iff the directed graph has no cycles.

Theorem: Show that there exists an algorithm for deciding whether a regular language, L is empty.
Proof: Construct a deterministic finite automata M accepting L. Determine the set of all the states reachable from q_0. If this set contains a final state, then L is non-empty otherwise L is empty.

Theorem: Show that there exists an algorithm for deciding whether a regular language, L is infinite.
Proof: Construct a deterministic finite automata M accepting L. L is infinite iff M has a cycle.

1.18 AKTU Examination Questions

1. Convert the following CFG to its equivalent GNF:
   S → AA a, A → SS b.

2. Prove that the following Language L = $\{a^n b^n c^n ! \ n \geq 1\}$ is not Context Free.

3. Is context free language closed under union? If yes, give an example.

4. Remove useless productions from the following grammar
   S→AB/ab, A→a/aA/B, B→D/E

5. Reduce the given Grammar G = ({S,A,B},{a,b},S,P) to Chomosky Norma Form, where P is the following
   S→bA/aB, A→a/aS/bAA, B→b/bS/aBB

6. Discuss the inherent ambiguity of context free languages with suitable example. Construct the context free grammar that accept the following language
   L = $\{a^i b^j c^k ! \ i = jorj = k\}$

7. Define the parse tree. Construct the parse tree for the string abbcde considering the productions
   S→aAcBe, A→b/Ab, B→d
   Is this ambiguous? Justify.

8. Prove or disprove that union and concatenation of two context free languages is also context free.

9. Prove that the language L = $\{a^n b^n c^n ! \ n \geq 1\}$ is neither regular nor context free.

10. Determine the language generated by grammar $S \rightarrow Sab|aSb|abS|baS|bSa|Sba|aS|a$

11. What is inherent ambiguity? Explain with the help of suitable example.

12. Remove the Unit productions from the following grammar: $S \rightarrow aSb|A$, $A \rightarrow cAd|cd$

13. Write the procedure to convert a given CFG into equivalent grammar in CNF. Apply the procedure and convert the grammar with following production into CNF: $S \rightarrow bA|aB$, $A \rightarrow bAA|aS|a$, $B \rightarrow aBB|bS|b$

14. Define Greibach normal form for a CFG. Reduce the following CFG into GNF: S → AB, $A \rightarrow BS|a$, $B \rightarrow A|b$

15. Let G be the grammar $S \to 0B|1A$, $A \to 0|0S|1AA$, $B \to 1|1S|0BB$ For the string 00110101, find: (i) The leftmost derivation. (ii) The rightmost derivation. (iii) The derivation tree.

16. Check whether the grammar is ambiguous or not. R→ R+R/ RR/ R*/ a / b / c. Obtain the string w = a+b*c

17. Eliminate unit productions in the grammar. S→A/bb A→B/b B→S/a

18. Find out whether the language L = $\{x^n y^n z^n | n \geq 1\}$ is context free or not.

19. Convert the following CFG into CNF
    S → XY / Xn / p
    X → mX / m
    Y → Xn / o

20. Convert the following CFG into CNF S → ASA / aB, A → B / S, B → b / ϵ

21. Write CFG for language L = $\{a^n b^n \ !n \geq 0\}$. Also convert it into CNF.

22. Define ambiguity. Show that the grammar G with following production is ambiguous.
    S → a / aAb / abSb, A → aAAb / bS

23. Convert the following grammar in GNF: S → AB , A→ BS / a , B→SA / b

24. Define derivation Tree. Show the derivation tree for string 'aabbbb' with the following grammar S→AB/ϵ, A→aB, B→Sb.