UNIT-2

Context-Free Grammars :-

① Context-Free Grammer :-

A context free grammer or CFG is represented by 4-typle (V, T, P, S) where

$V \rightarrow$ set of variables or non-terminals

$T \rightarrow$ set of terminals

$P \rightarrow$ set of productions

$S \rightarrow$ Starting variable

② Regular Grammer :-

A regular grammer is, similar to CFG, a formal grammer that decribes a regular language.

left regular grammer $\Rightarrow$ $S \rightarrow Bw$, $SB \rightarrow a$

Right regular grammer $\Rightarrow$ $S \rightarrow wB$, $B \rightarrow a$

③ Derivation Trees :-

A derivation tree (also called a parse tree) for a CFG $G = (V, T, P, S)$ is a tree satisfying the following conditions:

(i) Every vertex has a label which is a variable or terminal or Λ.

(ii) The root has label S.

(iii) The label of internal vertex is a variable.

(iv) If the vertices $n_1, n_2, \ldots, n_k$ written with labels $X_1, X_2, \ldots, X_k$ are the sons of vertex n with label A, then $A \rightarrow X_1 X_2 \ldots X_k$ is a production in P.

(v) A vertex n is a leaf if its label is $a \in \Sigma$ or Λ; n is the only son of its father if its label is Λ.

Leftmost Derivation :- A derivation $A \overset{*}{\Rightarrow} w$ is called a leftmost derivation if we apply a production only to the leftmost variable at every step.

Rightmost Derivation :- A derivation $A \overset{*}{\Rightarrow} w$ is called a rightmost derivation if we apply a production to the rightmost variable at every step.

④ Ambiguity in CFGs :-

A terminal string $w \in L(G)$ is ambiguous if there exist two or more derivation trees for w (or there exists two or more leftmost derivations of w).

⑤ Simplification of CFGs :-

(1) Construction of reduced grammar :-

→ construction of set of variables
→ construction of set of productions
} by removing useless symbol.

(2) Elimination of null productions :-

A variable A in a context free grammar is nullable if $A \overset{*}{\Rightarrow} \Lambda$

(3) Elimination of unit production :-

A unit production in CFGs is a production of the form $A \to B$, where A and B are variables in G

(4) Removal of left recursion :-

(left recursion becomes problem in designing of compiler that's why we remove left recursion not right recursion)

FORMULA :- If $A \to A\alpha_1 / A\alpha_2 / \dots / A\alpha_n / \beta_1 / \beta_2 / \dots / \beta_n$

where A is a variable and $\alpha_1, \alpha_2, \dots \alpha_n$ & $\beta_1, \beta_2, \dots, \beta_n$ are terminals

then $A \to \beta_1 A' / \beta_2 A' / \dots / \beta_n A'$

$A' \to \alpha_1 A' / \alpha_2 A' / \dots / \alpha_n A' / \epsilon$ is a solution.

(5) left factoring :-

Eg :- $A \to aA / a$

⑥ Normal Forms :-

When the productions in G satisfy certain restrictions, then G is said to be in a 'normal form'.

Chomsky Normal Form (CNF) :-

$A \to a$, $A \to BC$ and $S \to \Lambda$

Greibach Normal Form (GNF) :-

$A \to a\alpha^*$ and $S \to \Lambda$ $\quad a \to$ terminal, $\alpha \to$ variable

$A \to a$