6. Context-Free Grammars

The context 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 respectively . P is set of production rules, where each production rule is in the form of

Non terminal $\longrightarrow$ **non terminals**
Or Non terminal $\longrightarrow$ **terminals**

A context-free grammar (grammar for short) consists of terminals, Non terminals, S a start symbol, and productions.

1. Terminals are the basic symbols from which strings are formed. The term "token name" is a synonym for "'terminal" and frequently we will use the word "token" for terminal when it is clear that we are talking about just the token name. We assume that the terminals are the first components of the tokens output by the lexical analyzer.

2. Non terminals are syntactic variables that denote sets of strings. The sets of strings denoted by non terminals help define the language generated by the grammar. Non terminals impose a hierarchical structure on the language that is key to syntax analysis and translation.

3. In a grammar, one nonterminal is distinguished as the start symbol, and the set of strings it denotes is the language generated by the grammar. Conventionally, the productions for the start symbol are listed first.

4. The productions of a grammar specify the manner in which the terminals and non terminals can be combined to form strings. Each production consists of:

(a) A nonterminal called the head or left side of the production; this production defines some of the strings denoted by the head.

(b) The symbol +. Sometimes : : = has been used in place of the arrow.

(c) A body or right side consisting of zero or more terminals and non terminals. The components of the body describe one way in which strings of the nonterminal at the head can be constructed.

6.1.Notational Conventions

To avoid always having to state that "these are the terminals," "these are the non terminals ," and so on, the following notational conventions for grammars will be used throughout the remainder of this book.

1. These symbols are terminals:

(a) Lowercase letters early in the alphabet, such as a, b, e.

(b) Operator symbols such as +, **r,** and so on.

(c) Punctuation symbols such as parentheses, comma, and so on.

(d) The digits 0,1,. . . ,9.

(e) Boldface strings such as id or if, each of which represents a single terminal symbol.

2. These symbols are non terminals:

(a) Uppercase letters early in the alphabet, such as A, B, *C*.

(b) The letter S, which, when it appears, is usually the start symbol.

(c) Lowercase, italic names such as *expr* or *stmt.*

(**d)** When discussing programming constructs, uppercase letters may be used to represent non terminals for the constructs. For example, non terminals for expressions, terms, and factors are often represented by E, T, and F, respectively.

Example 1 : The grammar in Fig. 6.1 defines simple arithmetic expressions. In this grammar, the terminal symbols are The nonterminal symbols are expression, term and factor, and expression is the start symbol

Expression ⟶ expression + term
Expression ⟶ expression - term
expression ⟶ term
term ⟶ term * factor
term ⟶ term / factor
term ⟶ factor
factor ⟶ (expression)
factor ⟶ **id**

Figure 6.1: Grammar for simple arithmetic expressions

Example 2: Let

$$P=\left\{\begin{array}{l} S \longrightarrow S+S \\ S \longrightarrow S*S \\ S \longrightarrow (S) \\ S \longrightarrow 4 \end{array}\right\}$$

If the language is 4+4*4 then we can us the production rules given by P. The start symbol is S . The number of non terminals in the rules P is one and the only non terminal. The terminals are +,-,( ) and 4.

Example 3: the formation of production rules for checking syntax of any English statement is

SENTENCE ⟶ NOUN VERB

NOUN ⟶ RAMA / SEETA/ GOPAL

VERB ⟶ goes / writes / sings

Example 4: Construct the CFG for the regular expression (0+1)*

Sol: The CFG can be given by

$$P = \left\{ \begin{array}{l} S \longrightarrow 0S \ / \ 1S \\ S \longrightarrow \varepsilon \end{array} \right\}$$

The rules are in combination of 0'S and 1'S with start state symbol since indicates { Ɛ, 0 ,1, 01, 00 ,11, …..} in this set Ɛ is a sting. So in the rules we can set the rule

Example 5: Construct a grammar for the language containing strings of at least two a'S

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

V= {S, A}

T= {a,b}

$$P = \left\{ \begin{array}{l} S \longrightarrow AaAaA \\ A \longrightarrow aA \backslash\ bA \backslash\ \varepsilon \end{array} \right\}$$

6.2. Derivation and Languages

The production rules are used to drive certain strings. We will now formally define the language generated by grammar G=(V, T, P, S). The generation of language using specific rules is called derivation.

Definitions :

Let G=(V, T, P, S) be the context free grammar. If A B is a production of P and a and y are any strings from non terminals or terminals. Each nonterminal is replaced by the same body in the two derivations, but the order of replacements is different. To understand how parsers work, we shall consider derivations in which the nonterminal to be replaced at each step is chosen as follows:

1. Leftmost derivations:- Is a derivation in which the leftmost non terminal to replaced first from the sentential form.

2. Rightmost derivations Is a derivation in which the rightmost non terminal is replaced first from the sentential form**.**

Example 1: Derive the string(aba) for leftmost derivation and rightmost derivation using a CFG given by

S ⟶ XYX

X ⟶ a

Y ⟶ b

1- Leftmost derivation first	2- rightmost derivation
S ⟶ XYX	S ⟶ XYX
S ⟶ aYX	S ⟶ XYa
S ⟶ abX	S ⟶ Xba
S ⟶ aba	S ⟶ aba

Example2: Derive the string(aabbabba) for leftmost derivation and rightmost derivation using a CFG given by

Let P= S ⟶ aB | bA

A ⟶ a| aS | bAA

B ⟶ b| bS| aBB

1- Let us see the Leftmost derivation first

Start	User-rules
S	
aB	S $\longrightarrow$ aB
aaBB	B $\longrightarrow$ aBB
aabB	B $\longrightarrow$ b
aabbS	B $\longrightarrow$ bS
aabbaB	S $\longrightarrow$ aB
aabbabS	B $\longrightarrow$ bS
aabbabbA	S $\longrightarrow$ bA
aabbabba	A $\longrightarrow$ a

Is derived

2- Let us see the Rightmost derivation first

Start	User-rules
S	
aB	S $\longrightarrow$ aB
aaBB	B $\longrightarrow$ aBB
aaBbS	B $\longrightarrow$ bS
aaBbbA	S $\longrightarrow$ bA
aaBbba	A $\longrightarrow$ a
aabSbba	B $\longrightarrow$ bS
aabbAbba	S $\longrightarrow$ bA
aabbabba	A $\longrightarrow$ a

Is a derivation