

# CONTEXT FREE GRAMMAR (CFG)

Grammar  $\rightarrow$

$$G = (V, T, P, S) \quad P = \text{set of productions}$$

$V$  = Variable symbols

$N$  = Non-terminal symbols.

$T$  = Terminal symbols.

\* Any symbol which can be expanded or be replaced by any other symbol/string is a variable.

$P$  = Rules governing the rules of variable replacement.

$S$  = Starting state.

Notations :

- i) lowercase English alphabet (a, b, c)  $\rightarrow$  Terminal symbols.
- ii) uppercase alphabet (A, B, C)  $\rightarrow$  variable (S is reserved)
- iii)  $x, y, z$   $\rightarrow$  String of terminal symbols.
- iv)  $X, Y, Z$   $\rightarrow$  Single symbol (variable / Terminal).  
when we do not know what will be the next variable.

v)  $\alpha, \beta, \gamma$   $\rightarrow$  Strings of grammar symbols.  
 $\alpha$  is in  $(V \cup T)^*$  (may contain variable or terminal)

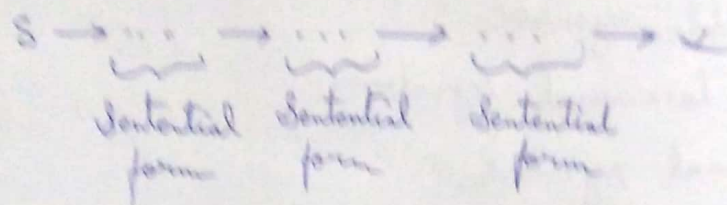
vi)  $\Gamma, \Delta$   $\rightarrow$  String of variables only.  
(uppercase Greek letter)



→ Derives

$$A \rightarrow \alpha \quad (\text{production})$$

read as "Variable A derives the string  $\alpha$ "



General form of production →

$$\alpha \rightarrow \beta$$

(String derives string)

If all  $\alpha$ 's are single variables i.e. all productions are of the form  $A \rightarrow \beta$ , then

$G$  is either regular or CFG.

- $G$  is regular if  $\beta$  consists of single or zero terminal symbols i.e.  $\beta$  is of the form  $w$  or  $wB$ , a string of only terminals or a string of terminals followed by a variable.

$$\begin{cases} \beta = w \text{ or } wB & (\text{right linear}) \\ \beta = w \text{ or } Bw & (\text{left linear}) \end{cases} \rightarrow \text{Linear grammar.}$$

\* CFG can be anything. \* \*

if  $\alpha \rightarrow \beta$ , length of  $\alpha$  in every production  $<$  length of  $\beta$   
is called as context free grammar.

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

$$P: \begin{aligned} S &\rightarrow 0S1 \\ S &\rightarrow \epsilon \end{aligned}$$

$\therefore$  The language accepted by this grammar =  $0^n 1^n$ .

instead of writing;  $A \rightarrow \beta_1; A \rightarrow \beta_2; A \rightarrow \beta_3$

we can write;  $A \rightarrow \beta_1 | \beta_2 | \beta_3$ .

$$S \rightarrow aSa | bSb | a | b | \epsilon$$

Language accepted by it,  $L = \{w \mid w = w^R\}$   
// all palindromes.

(C- is not nested; we can't define a function inside another function).

$$E \rightarrow E + E \mid E * E \mid (E) \mid a$$

(Generates all binary expressions with '+' and '\*').

Derivation sequence of  $id + id * id$

$$\begin{aligned} E &\rightarrow E + E \\ &\rightarrow E + E * E \\ &\rightarrow id + E * E \\ &\rightarrow id + id * E \\ &\rightarrow id + id * id \end{aligned}$$

\* If always the leftmost variable is replaced then it leftmost derivation. Otherwise, rightmost derivation. Always maintain the consistency.



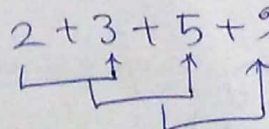
Leftmost derivation  $\rightarrow$   $E \rightarrow E + E$  ( $id + id$ )<sub>4</sub>  
 $\rightarrow id + E$   
 $\rightarrow id + E * E$   
 $\rightarrow id + id * E$   
 $\rightarrow id + id * id$   
 (\* is given precedence)

Another leftmost derivation  $\rightarrow$   $E \rightarrow E * E$   
 $\rightarrow E + E * E$   
 $\rightarrow id + E * E$   
 $\rightarrow id + id * E$   
 $\rightarrow id + id * id$   
 (+ is given precedence)

If  $G$  has two or more leftmost derivation or two or more rightmost derivations, then the Grammar is ambiguous.

Grammar  $\Rightarrow$   $E \rightarrow E + T \mid T$   
 $T \rightarrow T * F \mid F$   
 $F \rightarrow (E) \mid id$  } In this case, multiplication will always have higher priority than addition

$2 + 3 + 5 + 9$  } '+' is left associative



\* exponentiation is right associative.

Now, if '^' is to be given higher exponentiations  $\rightarrow$

$E \rightarrow E + T \mid T$   
 $T \rightarrow T * F \mid F$   
 $F \rightarrow G^F \mid G$  // This will make sure  
 $G \rightarrow (E) \mid id$  // that exponentiation  
 // is right associative



Context Free Grammar  $\rightarrow$

12/2/18

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

Every production is of the form ;  $A \rightarrow \alpha$

\* In case of sentential form, the general form is =  $\alpha A \beta$

The derivation process =

$$S \rightarrow \dots \xrightarrow{A} \dots \xrightarrow{\beta} \dots$$

For indicating a set of productions to get  $w$

$$S \Rightarrow w$$

for denoting the Grammar  $\rightarrow$

$$S \xRightarrow{G} w$$

for one or more times  $\rightarrow S \xRightarrow{*G} w$

$$\{x \mid n_a(x) = n_b(x)\} \quad // \text{ number of } a = \text{ number of } b.$$

$$S \Rightarrow asb \mid bsa \mid \cancel{ab} \mid \cancel{ba} \mid \epsilon \quad \left. \vphantom{S \Rightarrow asb \mid bsa \mid \cancel{ab} \mid \cancel{ba} \mid \epsilon} \right\} \rightarrow \text{limited}$$

$$\cancel{S \Rightarrow SS \mid ab \mid ba}$$

Correct one  $\rightarrow$

$$S \rightarrow asbs \mid bsas \mid \epsilon$$

abba
aabb
abab

\* In some languages, all the grammars considered are ambiguous. These are called inherently ambiguous languages.



\* The above grammar is ambiguous.

$$[S \rightarrow bsas | asbs | \epsilon]$$

for the string abab; there are two left linear derivations  $\Rightarrow$

$$S \rightarrow asbs \rightarrow a bs as bs \rightarrow ab as bs \rightarrow ab ab s \rightarrow ab ab$$

$$S \rightarrow asbs \rightarrow abs \rightarrow aba$$

For making an unambiguous grammar  $\rightarrow$

$$S \rightarrow AB | BA \quad \begin{cases} B \rightarrow \text{string with 1 b more than a's} \\ A \rightarrow \end{cases}$$

$$\Sigma = \{a^i b^j c^k \mid i, j, k \geq 0\}$$

$$S \rightarrow \{ bsas | asbs | \epsilon | \epsilon \}$$

$$\begin{array}{l} S \rightarrow \cancel{ab\epsilon} DC \\ D \rightarrow aDb | \epsilon \\ C \rightarrow cC | \epsilon \end{array}$$

Modified version  $\rightarrow$

$$\Sigma = \{a^i b^j c^k \mid i, j, k \geq 0 \text{ and } i \neq j\}$$

$$\begin{array}{l} S \rightarrow AB \\ A \rightarrow aA_1 | aC_1 | bC_2 \\ C_1 \rightarrow aC_1 | \epsilon \\ C_2 \rightarrow bC_2 | \epsilon \end{array} \quad B \rightarrow cB | \epsilon$$



Definition of identifier  $\rightarrow$

$$\boxed{\text{alpha} (\text{alpha} | \text{digit})^*}$$

$$E \rightarrow E + E \mid E * E \mid (E) \mid \epsilon$$

$$I \rightarrow I a \mid I b \mid I 0 \mid I 1 \mid a \mid b$$

Last one!!  $\rightarrow$

$$L_1: \{ a^n b^n c^m d^m ; n, m \geq 1 \}$$

$$\left[ \begin{array}{l} S \rightarrow ABCD \\ A \rightarrow aA \mid \epsilon \\ B \rightarrow bB \mid \epsilon \\ C \rightarrow cC \mid \epsilon \\ D \rightarrow d \end{array} \right]$$

This is not correct

$$S_1 \rightarrow A_1 B_1$$

$$A_1 \rightarrow aA_1 \mid b \mid \epsilon$$

$$B_1 \rightarrow cB_1 \mid d \mid \epsilon$$

This is correct

Consider another language  $\rightarrow L_2: \{ a^n b^m c^m d^n \mid n, m \geq 1 \}$

$$S_2 \rightarrow a S_2 b \mid a c_1 d$$

$$C_1 \rightarrow b C_1 c \mid bc$$

$$L_1 \cup L_2 = a^+ b^+ c^+ d^+$$

$$L_1 \cap L_2 = a^n b^n c^n d^n$$

\* Context free languages are closed under 'U' (union)

In all the grammars in  $L_1 \cup L_2$  are ambiguous.

$\therefore L_1 \cup L_2$  is inherently ambiguous.

a b c