CSE331

Date:...../...../.....

CFG "Context Free Grammer" :

$S \rightarrow 1S00$

$S \rightarrow \epsilon$

this is a production Rule.

suppose,

$S \rightarrow 0S1$ (r-1)

$S \rightarrow \epsilon$ (r-2)

So,

$S = 0S1$ (r-1)

$= 00S11$ (r-1)

$= 000S111$ (r-1)

$= 000111$ (r-2)

here,

S is Start symbol

ϵ is a terminal. it is without the L.S Var.

S is also a variable.

Regular Operations:

- $(a|b)$

$S \to A|B$
$A \to a$
$B \to b$

$S \to a|b$

- (ab)

$S \to AB$
$A \to a$
$B \to b$

$S \to ab$

- a^*

$S \to AS|\epsilon$
$A \to a$

$S \to Sa|\epsilon$

Ex:  ,

0 and 1 is equal in the string

Ex: $\{w/w = 0^n1^n, n \geq 0\}$

$S \to 0S1$
$S \to \epsilon$

$S \to 0S1|\epsilon$

$\hookrightarrow (01)^*$

Last 2 Ex ভালো করে বুঝে নিবে

$\{w/w = 0^n 1^{2n}, n \geq 0\}$

Date:...../...../.....

↳ num of 1 is double of num of 0.

$S \rightarrow 0S11 / \epsilon$

↳ (ϵ আসতে cos $n \geq 0$ আছে। মানে $(011)^*$)

$\{w/w = 0^i 1^j 2^k$

*Grammer is also valid for Mathematical operations.

→ Operators : +, −, ×, ÷, ( )

- all numbers are single digit.
- E is the starting,
- num → 0|1|2|3|4|5|6|7|8|9

So,

$E \rightarrow E + E$

$E \rightarrow E \times E$

$E \rightarrow E - E$

$E \rightarrow E \div E$

Let, $4 \times (3-2)$

$E \rightarrow E \times E$

$\rightarrow E \times (E-E)$

$\rightarrow num \times (num - num)$

$\rightarrow 4 \times (3-2)$

* String derivation:

Date:...../...../.....

Let, " num + num * num "

leftmost derivation

E → E+E

→ num + E

→ num + E*E

→ num + num * E

→ num+ num * num

rightmost derivation

E → E + E

→ E + E * E

→ E + E * num

→ E + num * num

→ num + num * num

→ It doesn't matter which operator we are ~~using~~ taking 1st at starting. The derivation and parse tree will be same for both cases for same operator.

leftmost derivation

E → E * E

→ E + E * E

→ num + E * E

→ num + num * E

→ num + num * num

rightmost derivation

~~E → E * E~~

E → E * E

→ E * num

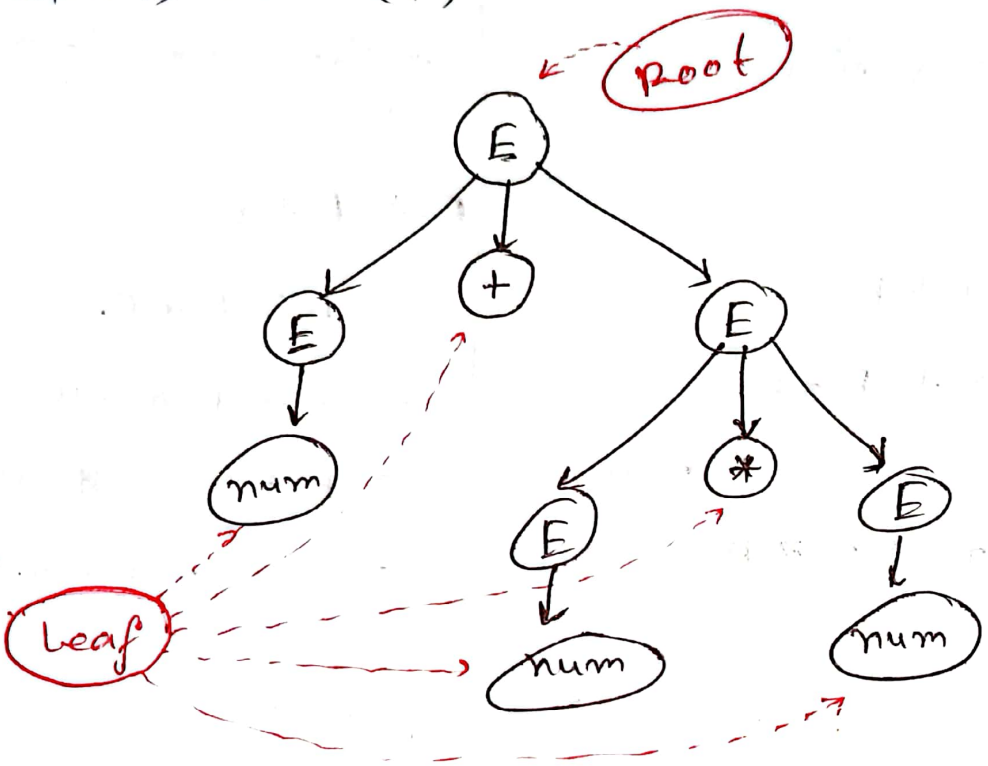
→ E + E * num

→ E + num * num

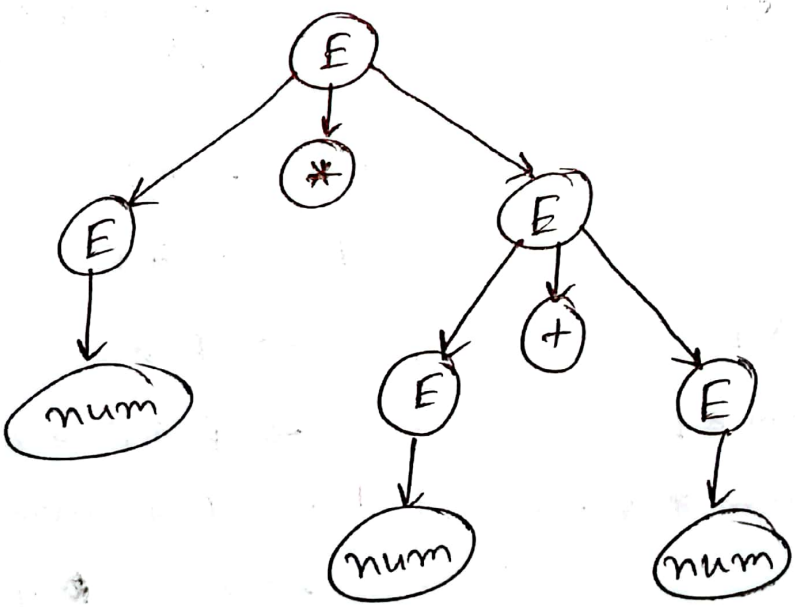
→ num + num * num

The parse Tree:

(i) for left and right most derivation ~~tos~~ ~~Start~~ starts with (+).



(ii) for left and right most derivation starts with (*).



* Ambiguity: A CFG is said to be ambiguous, if for some string 'w' that belongs to the language of the grammer there exists multiple parse trees or multiple leftmost derivations or multiple rightmost derivations.

In the prev two pages, we have seen the derivations and the parse trees. Here we can see that, the string 'num+num*num' has two left/rightmost derivations & or also two different parse trees. That means the string is ambiguous.

• সোজাসুজি, string যদি more than one way তে generate করা যায় & তাহলে সেই grammer এর ambiguity আছে।

(using leftmost but different approach or using rightmost but using diff approach.)

Ex: $A \xrightarrow{(1)} A1 \mid \xrightarrow{(2)} OA1 \mid \xrightarrow{(3)} O1$, lets see is this grammer have ambiguity or not. using "001111".

Now, we have to follow left/rightmost derivation

→ using leftmost,

$A \rightarrow A1 \quad (1)$

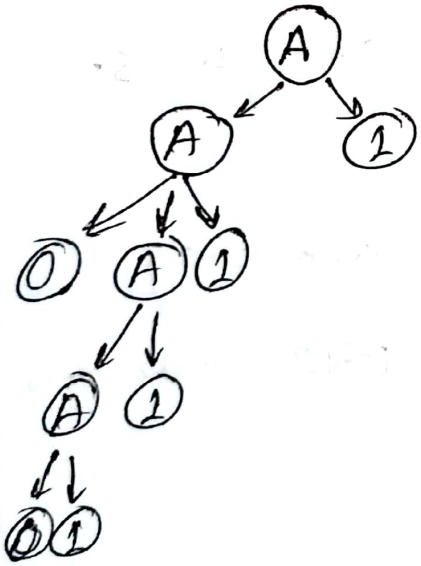
$\rightarrow OA11 \quad (2)$

$\rightarrow OA111 \quad (1)$

$\rightarrow OOA1111 \quad (3)$

$\rightarrow 001111$

parse tree,



→ using left most.

$A \rightarrow OA1 \quad (2)$

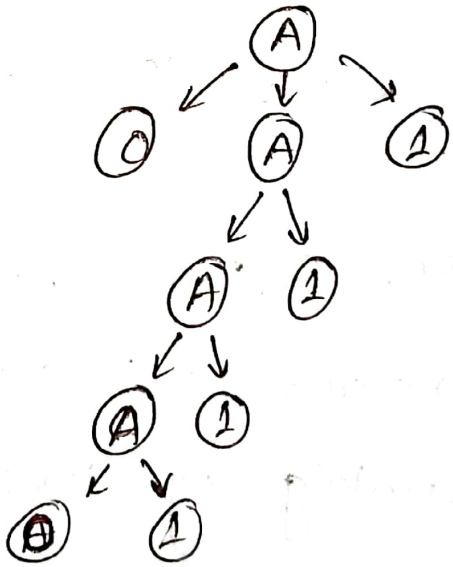
$\rightarrow OA11 \quad (1)$

$\rightarrow OA111 \quad (1)$

$\rightarrow OO1111 \quad (3)$

$\rightarrow 001111$

parse tree,



here, derivation is same but parse tree is different.

So, this grammer is ambiguous.

* Regular Operations :

Date: ...../...../.....

$(0|10)^*$ | 10^*

$A \rightarrow 0$

$B \rightarrow 10$

$C \rightarrow A|B$

$D \rightarrow CD|\epsilon$

$E \rightarrow 1$

$F \rightarrow 0F|\epsilon$

$G \rightarrow 1F$

$S \rightarrow D|G$

again,

$A \rightarrow 0|10$

$B \rightarrow AB|\epsilon$

$C \rightarrow 1$

$D \rightarrow 0D|\epsilon$

$C \rightarrow 0C|\epsilon$

$D \rightarrow 1C$

$S \rightarrow B|CD$

$S \rightarrow B|D$

watch 31 and 32 number video

CNF (Chomsky Normal Form):

- 4 steps for CFG to CNF conversion.
  - 1. Eliminate ϵ-production (nullable)
  - 2. Eliminate unit production (Var → Var)
  - 3. Eliminate the var with no terminal.
  - 4. Eliminate unreachable var where we can not reach from S.

• cleaning up the Grammer using 4 Steps.

→ A CFG is called a CNF if every production is one of these two forms,

1. $A \rightarrow BC$ [two var in RS]
2. $A \rightarrow a$ [one terminal in RS]

- if there is any $C \rightarrow \epsilon$ production, then eliminate C from everywhere in first step.

Ex-1 : $\Sigma = \{a, b\}$

Date:...../...../.....

$S \rightarrow ASB \mid a$

$A \rightarrow aAS \mid a \mid \epsilon$

$B \rightarrow SbS \mid A \mid bb$

S-1 : (eliminate ϵ-pro)

here, nullable var,

$A \rightarrow \epsilon$

$B \rightarrow A \rightarrow \epsilon$

$S \rightarrow ASB \mid a \mid ASE \mid ESB$

$A \rightarrow aAS \mid a \mid aES$

$B \rightarrow SbS \mid A \mid bb$

$S \rightarrow ASB \mid a \mid AS \mid SB$

$A \rightarrow aAS \mid a \mid aS$

$B \rightarrow SbS \mid A \mid bb$

S-2 (eliminat U-pro)

unit var,

$B \rightarrow A$

$S \rightarrow ASB \mid a \mid AS \mid SB$

$A \rightarrow aAS \mid a \mid aS$

$B \rightarrow SbS \mid aAS \mid a \mid aS \mid bb$

A

S-3 → all var has terminals

S-4 → from S, we can reach all var (A, B)

finally, making CNF form

$S \rightarrow XB \mid a \mid AS \mid SB$

$A \rightarrow YX \mid a \mid YS$

$B \rightarrow PS \mid YX \mid a \mid YS \mid ZZ$

$X \rightarrow AS$

$Y \rightarrow a$

$Z \rightarrow b$

$P \rightarrow SZ$

Ex-2:

$S \rightarrow aAa \mid bBb \mid \epsilon$

$A \rightarrow C \mid a$

$B \rightarrow C \mid b$

$C \rightarrow CDE \mid \epsilon$

$D \rightarrow A \mid B \mid ab$

(S-1) eliminate ϵ production

here,

nullable var,

$C \rightarrow \epsilon$

$A \rightarrow \epsilon$

$B \rightarrow \epsilon$

$S \rightarrow \epsilon$

$D \rightarrow A \rightarrow \epsilon$

$S \rightarrow aAa \mid bBb \mid aa \mid bb$

Date:...../...../.....

$A \rightarrow C \mid a$

$B \rightarrow C \mid b$

$C \rightarrow CDE \mid CE \mid DE \mid E$

$D \rightarrow A \mid B \mid ab$

(S-2) eliminate Unit-Production [A, B, C]

$S \rightarrow aAa \mid bBb \mid aa \mid bb$

$A \rightarrow CDE \mid CE \mid DE \mid E \mid a$

$B \rightarrow CDE \mid CE \mid DE \mid E \mid b$

$C \rightarrow CDE \mid CE \mid DE \mid E$

$D \rightarrow CDE \mid CE \mid DE \mid a \mid b \mid ab$

(S-3) eliminate no-~~non~~ terminal ~~variable~~. production rule. [C]

~~here, as C~~

here, as C has no terminal, we will remove all C from everywhere. Also, E doesn't have any production rule we have to remove E also.

$S \rightarrow aAa \mid bBb \mid aa \mid bb$

$A \rightarrow CDE \mid CE \mid DE \mid E \mid a$

$B \rightarrow CDE \mid CE \mid DE \mid E \mid b$

$D \rightarrow CDE \mid CE \mid DE \mid a \mid b \mid ab$

~~S-A unreach~~

(S-4) eliminate unreachable variable from start(S). [D]

$S \rightarrow aAa \mid bBb \mid aa \mid bb$

$A \rightarrow a$

$B \rightarrow b$

finally in CNF, [two var | one terminal]

$S \rightarrow AAA \mid BBB \mid AA \mid BB$

$A \rightarrow a$

$B \rightarrow b$

again,

$S \rightarrow AX \mid BY \mid AA \mid BB$

$A \rightarrow a$

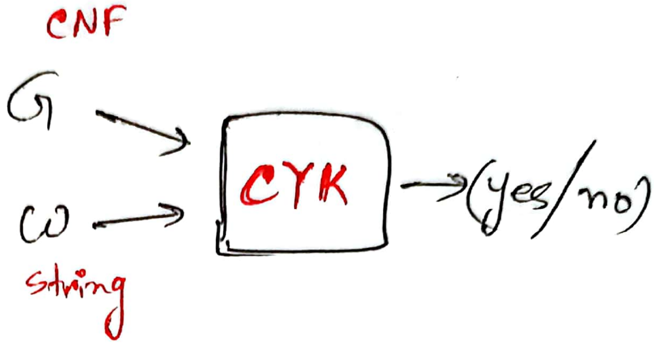
$B \rightarrow b$

$X \rightarrow AA$

$Y \rightarrow BB$

CYK :

Date:....../......./.......



- make a box according to the length of the string.
- use 'up and slide' rule.
- first row will be filled up by seeing the terminal generated by ~~the~~ each variable.
- second row will be filled up by 'up and slide ~~rule~~. rule.
- Then sequence ~~ঠি~~ thik থাকবে

Ex: "baaba"

Date:...../...../.....

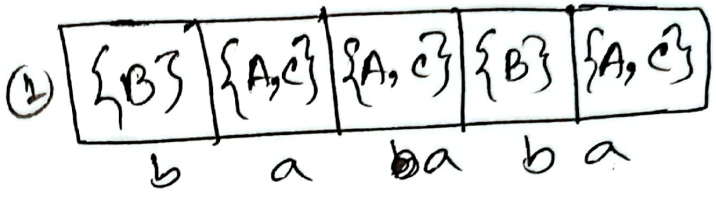
$S \rightarrow AB/BC$

$A \rightarrow BA/a$

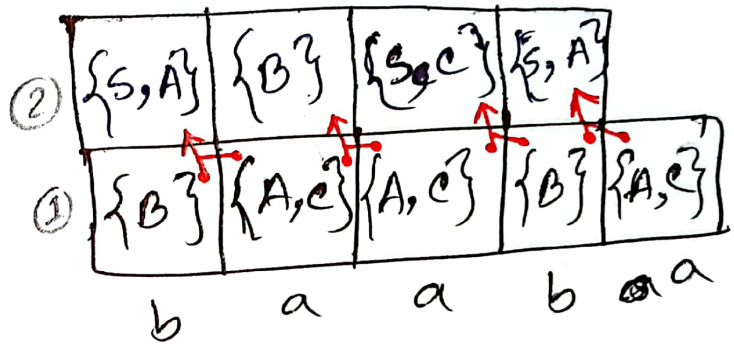
$B \rightarrow CC/b$

$C \rightarrow AB/a$

(I)



(II)



$\rightarrow \{BA\}\{BC\}$

$A \quad S \Rightarrow \{SA\}$

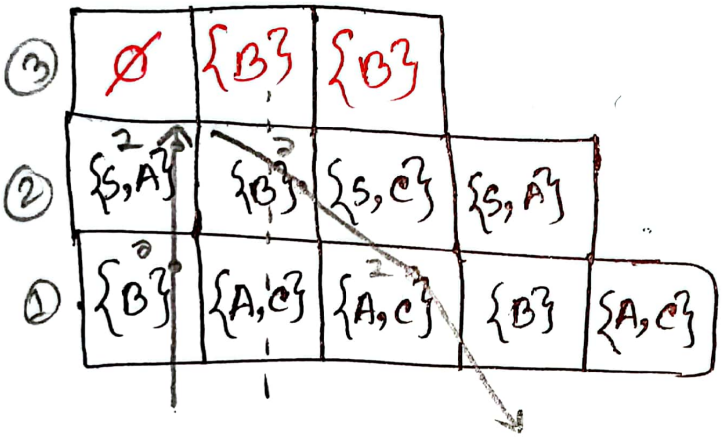
$\rightarrow \{AC\}\{AC\}$

$\rightarrow \{AA\}\{AC\}\{CA\}\{CC\}$

$\times \quad \times \quad \times$

$B \Rightarrow \{B\}$

(III)



Now,

$2 \times 2 \rightarrow \{B\}\{B\} \rightarrow \{BB\} \rightarrow \times$

$2 \times 2 \rightarrow \{S,A\}\{A,C\} \rightarrow$

$\hookrightarrow \{SA\}\{SC\}\{AA\}\{AC\}$

$\times \quad \times \quad \times \quad \times$

$\{\emptyset\}$

COMPLETE!