UNIT - III

Context Free Grammer

classmate
Date
Page

Context Free Grammer

A grammer is said to be context free grammer if it is in the form $\alpha \rightarrow \beta$ where α & β are arbitary strings of grammer symbols it may be Terminal (T) or non Terminal (NT) or both.
And $|\alpha| \leq |\beta|$ and $\alpha \in V$.

The CFG is denoted by $G = (V, \Sigma, P, S)$
where,

V = finite set of variable or non-terminal
Σ = finite set of input symbols.
P = finite set of productions
S = Start symbol of the grammer.

Q. Construct a CFG for generating a language
$L = \{a^n b a^n : n \geq 1\}$

→ Given language

$$L = \{a^n b a^n : n \geq 1\}$$

Set of strings generated by the given language will be

$$L^* = \{aba, aabaa, aaabaaa, \ldots\ldots\}$$

The required grammer is $G = \langle V, \Sigma, P, S \rangle$
where,

Productions are

$$S \rightarrow aAa$$
$$A \rightarrow b \mid aAa$$

UNIT - III.

Context Free Grammer

Context Free Grammer

A grammer is said to be context free grammer if it is in the form $\alpha \to \beta$ where α & β are arbitarey strings of grammer symbols it may be Terminal (T) or non Terminal (NT) or both.

And $|\alpha| \leq |\beta|$ and $\alpha \in V$.

The CFG is denoted by $G = (V, \Sigma, P, S)$

where,

V = finite set of variable or non-terminal

Σ = finite set of input symbols.

P = finite set of productions

S = Start symbol of the grammer.

Q. Construct a CFG for generating a language $L = \{a^n b a^n : n \geq 1\}$

→ Given language

$$L = \{a^n b a^n : n \geq 1\}$$

Set of strings generated by the given language will be

$$L^* = \{aba, aabaa, aaabaaa, \ldots\ldots\}$$

The required grammer is $G = \langle V, \Sigma, P, S \rangle$

where,

Productions are

$$S \to aAa$$

$$A \to b \mid aAa$$

where,

$V = \{S, A\}$
$\Sigma = \{a, b\}$
S = start symbol

Verification for string 'aabaa'

$S \rightarrow aAa$
$S \rightarrow aaAaa$
$S \rightarrow aabaa$

Q. $L = \{a^n b a^n \mid n \geq 0\}$

→ Set of steings generated by the given language will be,

$L^* = \{b, aba, aabaa, \ldots\ldots\}$

The required grammer, $G = \langle V, \Sigma, P, S \rangle$
where,

Productions aree,

$S \rightarrow b \mid aSa$

where,

$V = \{S\}$
$\Sigma = \{a, b\}$
S = Start symbol

Verification for string 'aba'

$S \rightarrow aSa$
$S \rightarrow aba$

Q. $L = \{ ab^n ab^n a \mid n \geq 0 \}$

→ Set of strings generated by the given language will be,

$L^* = \{ aaa, ababa, abbabba, \ldots \}$

For $n=0$, $ab^0ab^0a = aaa$
$n=1$, $ab^1ab^1a = ababa$
$n=2$, $ab^2ab^2a = abbabba$
⋮

The required grammer is, $G = \langle V, \Sigma, P, S \rangle$ where,

productions are,

$S \rightarrow aAa$
$A \rightarrow a \mid bAb$

where,

$V = \{S, A\}$
$\Sigma = \{a, b\}$
S = start symbol

Verification for string 'abbabba'

$S \rightarrow aAa$
$S \rightarrow abAba$ $\quad A \rightarrow bAb$
$S \rightarrow abbAbba$ $\quad A \rightarrow bAb$
$S \rightarrow abbabba$ $\quad A \rightarrow a$

Q. $L = \{ ab^n ab^n a \mid n \geq 1 \}$

→ The set of steings generated by given language will be,

for $n=1$ ababa
$n=2$ abbabba
$n=3$ abbbabbba
:

$L^* = \{$ ababa, abbabba, abbbabbba, ... $\}$

The required grammer is $G = (V, \Sigma, P, S)$ where,

productions are,

$S \rightarrow abAba$
$A \rightarrow a \mid bAb$

where,

$V = \{S, A\}$
$\Sigma = \{a, b\}$
S = Start symbol

Verification for string abbabba,

$S \rightarrow abAba$
$S \rightarrow abbAbba$ $\quad A \rightarrow bAb$
$S \rightarrow abbabba$ $\quad A \rightarrow a$

$L = \{0^n 1^{2n} \mid n \geq 0\}$

The set of strings generated by given language will be,

For $n = 0$, ϵ

$n = 1$, $0^1 1^{2 \cdot 1} = 011$

$n = 2$, $0^2 1^{2 \cdot 2} = 0000\,001111$

$n = 3$, $0^3 1^{2 \cdot 3} = 000111111$

$\vdots$

$L^* = \{\epsilon, 011, 001111, 000111111, \ldots\ldots\}$

The required grammer is $G = (V, \Sigma, P, S)$ where,

$V = \{S\}$

$\Sigma = \{0, 1\}$

P = production

S = start symbol

Productions are,

$S \rightarrow 0S11 \mid \epsilon$

Verification for String '001111'

$S \rightarrow 0S11$

$S \rightarrow 00S1111$ $\quad S \rightarrow 0S11$

$S \rightarrow 001111$ $\quad S \rightarrow \epsilon$

Q. $L = \{0^n 1^{2n} \mid n \geq 1\}$

→ The set string generated by given language will be,

for $n=1$: 011
$n=2$: 001111
$n=3$: 000111111
:

$L^* = \{011, 001111, 000111111, \ldots\}$

The required grammer is $G = (V, \Sigma, P, S)$ where,

productions are

$S \rightarrow 011 \mid 0S11$

where

$V = \{S\}$
$\Sigma = \{0, 1\}$
S = Start Symbol

Verification for string 000111111

$S \rightarrow 0S11$
$S \rightarrow 00S11\,11$ $\quad S \rightarrow 0S11$
$S \rightarrow 000111111$ $\quad S \rightarrow 011$

* Derivation Tree.

The derivation in a CFG can be represented using trees such trees representing derivation are called derivation tree.

The process of deriving a string we form S using production of grammer is known as derivation. symbolically this is written as $S \rightarrow \omega$

If we use graphical representation of the derivation using tree, then it is known as derivation tree or Parse tree.

Left-most derivation & Right-most derivation.

If in the each step in the derivation, production is applied to the leftmost variable (Non-Terminal), then the derivation is called left most derivation.

If in the each step in the derivation, production is applied to rightmost variable (Non-Terminal), Then the derivation is called right most derivation.

Q. Let G be the grammer

$S \rightarrow bA \mid aB$
$A \rightarrow a \mid aS \mid bAA$
$B \rightarrow b \mid bS \mid aBB$

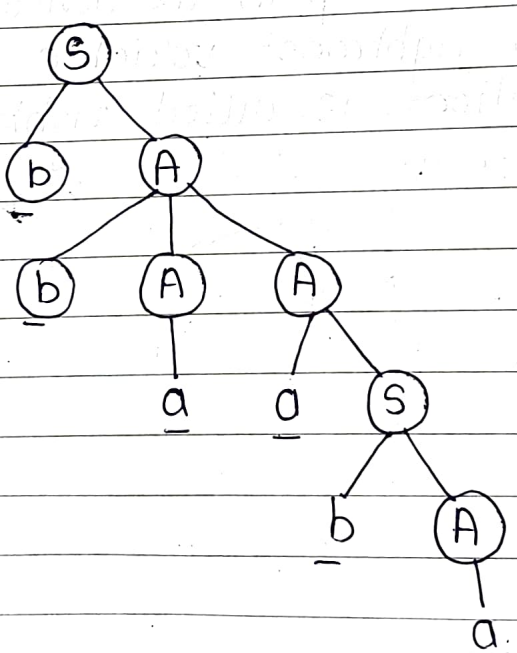
For string bbaaba find :-
(a) Leftmost derivation
(b) Rightmost derivation
(c) Parse Tree.

→ Leftmost Derivation

$S \rightarrow bA$
$S \rightarrow bbAA$
$\rightarrow bbaA$
$\rightarrow bbaaS$
$\rightarrow bbaabA$
$\rightarrow bbaaba$

$S \rightarrow bA$
$A \rightarrow bAA$
$A \rightarrow a$
$A \rightarrow aS$
$S \rightarrow bA$
$A \rightarrow a$

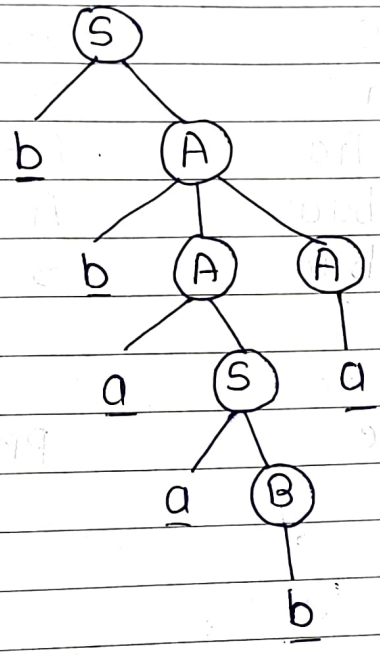
Parse Tree



Rightmost Derivation

$S \rightarrow bA$	$S \rightarrow bA$
$\rightarrow bbAA$	$A \rightarrow bAA$
$\rightarrow bbAa$	$A \rightarrow a$
$\rightarrow bbaSa$	$A \rightarrow aS$
$\rightarrow bbaaBa$	$S \rightarrow aB$
$\rightarrow bbaaba$	$B \rightarrow b$

Parse Tree



Q. Consider Grammer $G = (\{S, A\}, \{a, b\}, P, S)$ where,

P consist of

$S \rightarrow aAS \mid a$

$A \rightarrow SbA \mid SS \mid ba$

For string aabbaa

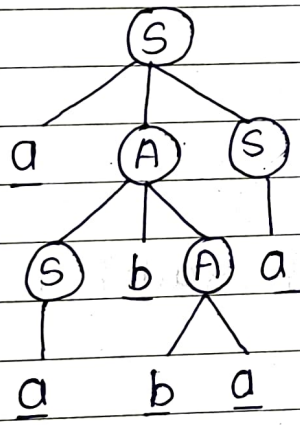
→ Leftmost derivation

$S \rightarrow aAS$ — $S \rightarrow aAS$
$\rightarrow aSbAS$ — $A \rightarrow SbA$
$\rightarrow aabAS$ — $S \rightarrow a$
$\rightarrow aabbaS$ — $A \rightarrow ba$
$\rightarrow aabbaa$ — $S \rightarrow a$

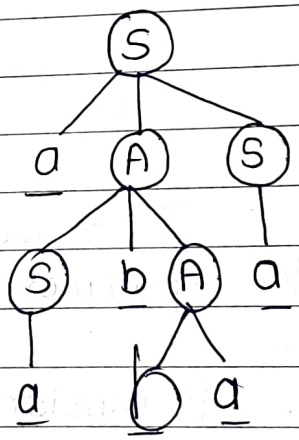
Rightmost derivation

$S \rightarrow aAS$ — $S \rightarrow aAS$
$\rightarrow aAa$ — $S \rightarrow a$
$\rightarrow aSbAa$ — $A \rightarrow SbA$
$\rightarrow aSbbaa$ — $A \rightarrow ba$
$\rightarrow aabbaa$ — $S \rightarrow a$

LMD- Parse tree



RMD- Parse Tree



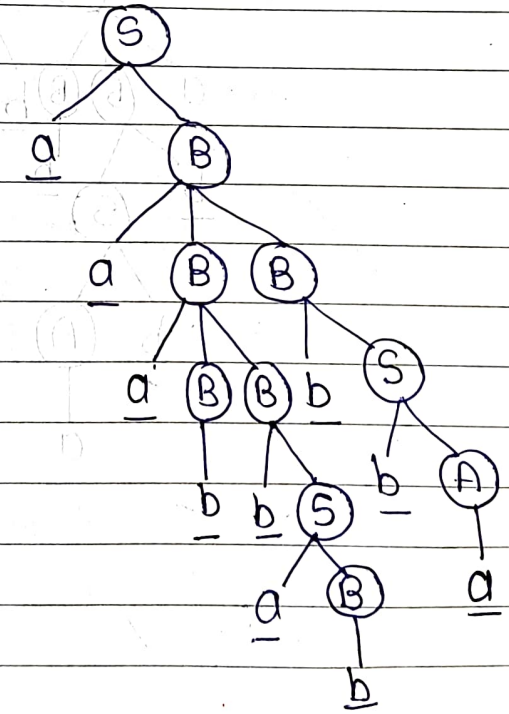
8. $S \rightarrow aB \mid bA$
$A \rightarrow a \mid aS \mid bAA$
$B \rightarrow b \mid bS \mid aBB$

for string 'aaabbabbba'

→ leftmost derivation

$S \rightarrow aB$	$S \rightarrow aB$
$\rightarrow aaBB$	$B \rightarrow aBB$
$\rightarrow aaaBBB$	$B \rightarrow aBB$
$\rightarrow aaabBB$	$B \rightarrow b$
$\rightarrow aaabbSB$	$B \rightarrow bS$
$\rightarrow aaabbaBB$	$S \rightarrow aB$
$\rightarrow aaabbabB$	$B \rightarrow b$
$\rightarrow aaabbabbS$	$B \rightarrow bS$
$\rightarrow aaabbabbbA$	$S \rightarrow bA$
$\rightarrow aaabbabbba$	$A \rightarrow a$

Parse Tree

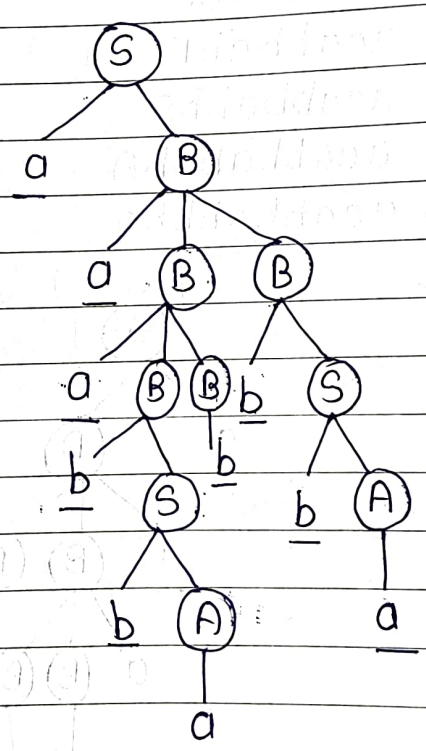


Rightmost Tree

$S \rightarrow aB$
$\rightarrow aaBB$
$\rightarrow aaBbS$
$\rightarrow aaBbbA$
$\rightarrow aaBbba$
$\rightarrow aaaBBbba$
$\rightarrow aaaBbbba$
$\rightarrow aaabSbbba$
$\rightarrow aaabbAbbba$
$\rightarrow aaabbabbba$

$S \rightarrow aB$
$B \rightarrow aBB$
$B \rightarrow bS$
$S \rightarrow bA$
$A \rightarrow a$
$B \rightarrow aBB$
$B \rightarrow b$
$B \rightarrow bS$
$S \rightarrow bA$
$A \rightarrow a$

Parse Tree



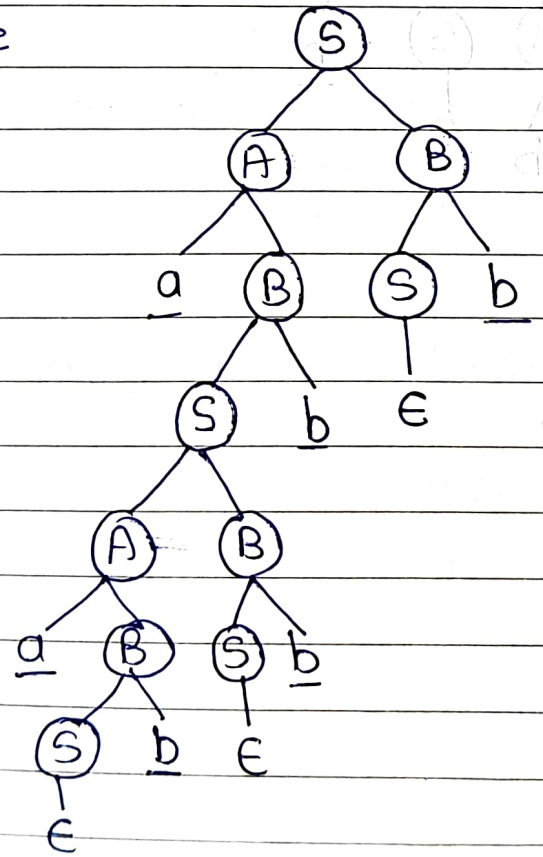
$S \rightarrow AB/\epsilon$
$A \rightarrow aB$
$B \rightarrow Sb$

for string 'aabbbb'

leftmost derivation

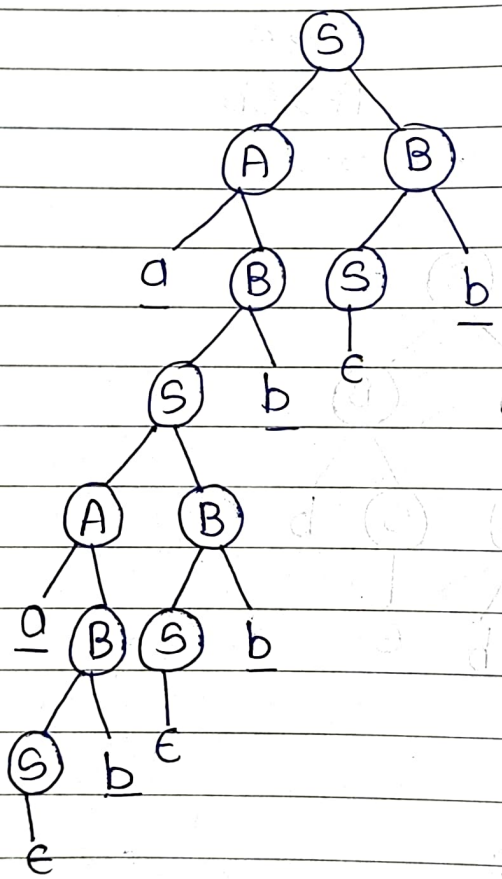
$S \rightarrow AB$	$S \rightarrow AB$
$\rightarrow aBB$	$A \rightarrow aB$
$\rightarrow aSbB$	$B \rightarrow Sb$
$\rightarrow aABbB$	$S \rightarrow AB$
$\rightarrow aaBBbB$	$A \rightarrow aB$
$\rightarrow aaSbBbB$	$B \rightarrow Sb$
$\rightarrow aabBbB$	$S \rightarrow \epsilon$
$\rightarrow aabSbbB$	$B \rightarrow Sb$
$\rightarrow aabbbbB$	$S \rightarrow \epsilon$
$\rightarrow aabbbSb$	$B \rightarrow Sb$
$\rightarrow aabbbb$	$S \rightarrow \epsilon$

Parse Tree



Rightmost derivation

$S \rightarrow AB$	$S \rightarrow AB$
$\rightarrow ASb$	$B \rightarrow Sb$
$\rightarrow Ab$	$S \rightarrow \epsilon$
$\rightarrow aBb$	$A \rightarrow aB$
$\rightarrow aSbb$	$B \rightarrow Sb$
$\rightarrow aABbb$	$S \rightarrow AB$
$\rightarrow aASbbb$	$B \rightarrow Sb$
$\rightarrow aAbbb$	$S \rightarrow \epsilon$
$\rightarrow aaBbbb$	$A \rightarrow aB$
$\rightarrow aaSbbbb$	$B \rightarrow Sb$
$\rightarrow aabbbb$	$S \rightarrow \epsilon$



Q. Construct a CFG for generating a language $L = \{ 0^n 1^m 0^m 1^n , \; m \geq 0 \;\; n \geq 1 \}$

→

	m	n
01	0	1
0011	0	2
001011	1	2
00010111	1	3
00110011	2	2
000111000111	2	3

$L^* = \{ 01, 0011, 001011, 00010111, \ldots \}$

Production are,

$S \rightarrow 0A1$

$A \rightarrow 0A1 / B$

$B \rightarrow 1B0 / \epsilon$

where,

$V = \{ S, A, B \}$

$\epsilon = \{ 0, 1 \}$

S = Start Symbol

Verification for string 001011

$S \rightarrow 0A1$

$S \rightarrow 00A11$

$S \rightarrow 00B11$

$S \rightarrow 001B011$

$S \rightarrow 001011$

Q. $L = \{0^{n}1^{m}0^{m}1^{n} \mid m \geq 0, n \geq 0\}$

→

	m	n
∈	0	0
0 1	0	1
00 11	0	2
0101	1	1
0011	1	2
00110011	2	2
10	1	0
1100	2	0
111000	3	0

$L^{*} = \{\in, 01, 10, 0011, 1100, 111000, 001011, \ldots\}$

Productions are,

$S \rightarrow 0S1 \mid A$

$A \rightarrow 1A0 \mid \in$

Verification for string 001011

$S \rightarrow 0S1$

$\rightarrow 00S11$

$\rightarrow 00A11$

$\rightarrow 001A011$

$\rightarrow 001011$

* Ambigious Grammer

A grammer that produce more than one parse tree for same sentence (leftmost derivation or rightmost der) is said to be ambigious grammer.

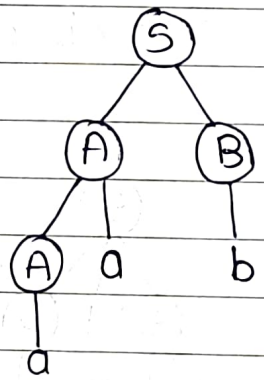
Q. Show that following grammer is ambigious. (aab)

$S \rightarrow AB \mid aaB$
$A \rightarrow a \mid Aa$
$B \rightarrow b$

→ We can generate parse tree for these leftmost derivation as follows:

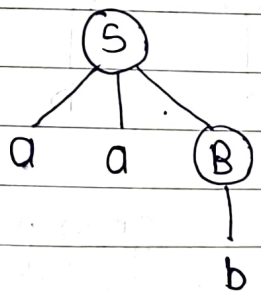
for string 'aab'

$S \rightarrow \underline{AB}$ $\quad S \rightarrow AB$
$S \rightarrow \underline{AaB}$ $\quad A \rightarrow Aa$
$S \rightarrow a\underline{aB}$ $\quad A \rightarrow a$
$S \rightarrow aa\underline{b}$ $\quad B \rightarrow b$



or

$S \rightarrow a\underline{aB}$ $\quad S \rightarrow aaB$
$S \rightarrow aab$ $\quad B \rightarrow b$



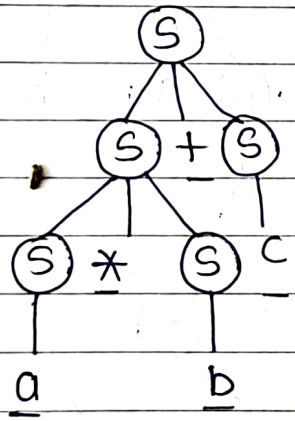
∴ The given production is ambigious.

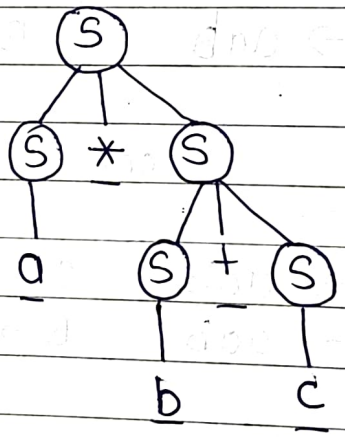
Q. $S \rightarrow S+S \mid S*S \mid a \mid b \mid c$

→ Suppose for string $a*b+c$
Lets use leftmost derivation

$S \rightarrow \underline{S}+S$ $\quad S \rightarrow S+S$
$\rightarrow \underline{S}*S+S$ $\quad S \rightarrow S*S$
$\rightarrow a*\underline{S}+S$ $\quad S \rightarrow a$
$\rightarrow a*b+\underline{S}$ $\quad S \rightarrow b$
$\rightarrow a*b+c$ $\quad S \rightarrow c$

$S \rightarrow \underline{S}*S$ $\quad S \rightarrow S*S$
$\rightarrow a*\underline{S}$ $\quad S \rightarrow a$
$\rightarrow a*\underline{S}+S$ $\quad S \rightarrow S+S$
$\rightarrow a*b+\underline{S}$ $\quad S \rightarrow b$
$\rightarrow a*b+c$ $\quad S \rightarrow c$





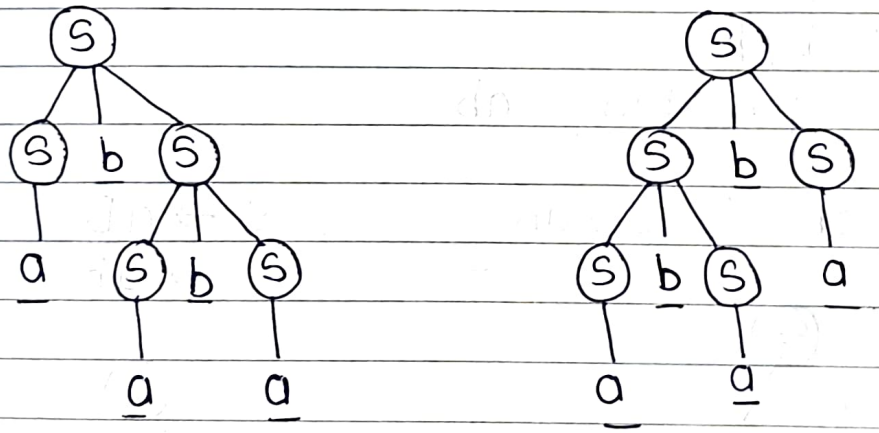
∴ The given production is ambiguous.
As it produce more than one leftmost derivation for same sentence.

Q. $S \rightarrow SbS \mid a$

→ Suppose for string 'ababa'

$S \rightarrow \underline{S}bS \quad S \rightarrow SbS$
$\rightarrow ab\underline{S} \quad S \rightarrow a$
$\rightarrow ab\underline{S}bS \quad S \rightarrow SbS$
$\rightarrow abab\underline{S} \quad S \rightarrow a$
$\rightarrow ababa \quad S \rightarrow a$

$S \rightarrow \underline{S}bS \quad S \rightarrow SbS$
$\rightarrow \underline{S}bSbS \quad S \rightarrow SbS$
$\rightarrow ab\underline{S}bS \quad S \rightarrow a$
$\rightarrow abab\underline{S} \quad S \rightarrow a$
$\rightarrow ababa \quad S \rightarrow a$



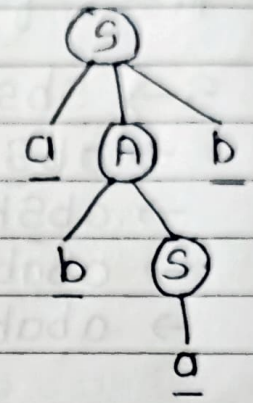
∴ The given production is ambiguous.
As it produce more than one leftmost derivation for same sentence.

Q. $S \to a \mid abSb \mid aAb$
$A \to bS \mid aAAb$

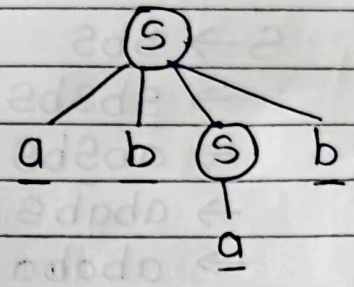
→

Suppose for 'abab'

$S \to aAb \quad S \to aAb$
$\to abSb \quad A \to bS$
$\to abab \quad S \to a$



$S \to abSb \quad S \to absb$
$\to abab \quad S \to a$

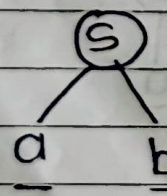


∴ The given production is ambiguous.

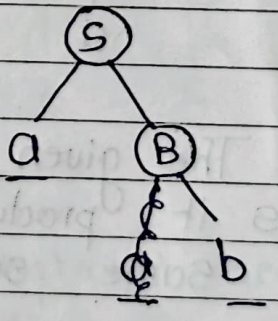
Q. $S \to aB \mid ab$
$B \to ABb \mid b$

→ Suppose for string ab

$S \to ab \quad S \to ab$



$S \to aB \quad S \to aB$
$S \to ab \quad B \to b$



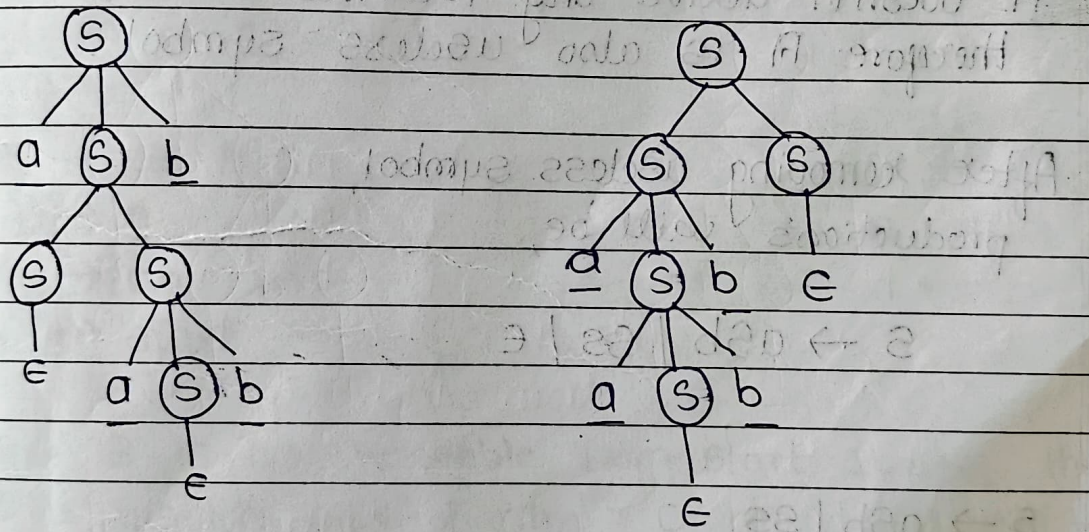
∴ The production is ambiguous.

Q. $S \rightarrow aSb \mid SS \mid \epsilon$

→ Lets consider for string aabb
By using leftmost derivation,

$S \rightarrow \underline{a}Sb$	$S \rightarrow aSb$
$\rightarrow a\underline{S}Sb$	$S \rightarrow SS$
$\rightarrow a\underline{a}Sbb$	$S \rightarrow \epsilon$
$\rightarrow aa\underline{S}bb$	$S \rightarrow aSb$
$\rightarrow aabb$	$S \rightarrow \epsilon$

$S \rightarrow \underline{S}S$	$S \rightarrow SS$
$\rightarrow aSb\underline{S}$	$S \rightarrow aSb$
$\rightarrow aa\underline{S}bbS$	$S \rightarrow aSb$
$\rightarrow aabb\underline{S}$	$S \rightarrow \epsilon$
$\rightarrow aabb$	$S \rightarrow \epsilon$



∴ The given production is ambiguous.
as they produce differe leftmost derivation & parse tree for same sentence.

* Removing Useless Productions

There are two reasons which make any variable useless.

1) Which can not reach to that variable from start symbol.
2) The variable does not derive terminal.

for eg. From ① & ② point

Q. $S \rightarrow aSb \mid SS \mid \epsilon \mid A$
$A \rightarrow aA$
$B \rightarrow ddd$

Here B is not reachable from start symbol S.
Therefore B is useless symbol.
A doesn't derive any terminal
therefore A is also useless symbol.

After removing useless symbol, productions will be,

$S \rightarrow aSb \mid SS \mid \epsilon$

Q. $S \rightarrow aSb \mid SS$

Here S is useless symbol because the variable does not derive terminal string.

Q. $S \rightarrow aSb \mid A \mid \epsilon$
$A \rightarrow aA$

Here production $A \rightarrow aA$ is useless because /A can not be transformed into terminal string. So Remove production A, as A is useless symbol.

$\therefore \quad S \rightarrow aSb \mid \epsilon$

Q. $S \rightarrow A$
$A \rightarrow aA \mid \epsilon$
$B \rightarrow bA$

Here production $B \rightarrow bA$ is useless because we never reach to the variable B from S. therefore B is useless production.

$S \rightarrow A$
$A \rightarrow aA \mid \epsilon$

Q. $S \rightarrow aS \mid A \mid C$
$A \rightarrow a$
$B \rightarrow aa$
$C \rightarrow aCb$

Here B is not reachable from start symbol therefore B is useless symbol and C can not reach to the terminal string therefore C is also useless symbol.

$\therefore \quad S \rightarrow aS \mid A$
$A \rightarrow a$

Q. $S \to a \mid aA \mid B \mid C$
$A \to aB \mid \epsilon$
$B \to Aa$
$C \to CCD$
$D \to ddd$

Here D is not reachable from start symbol. Therefore D is useless symbol and can be removed.

C production doesn't reach to the terminal symbol Therefore C is useless & can be removed.

$\therefore$ $S \to a \mid aA \mid B$
$A \to aB \mid \epsilon$
$B \to Aa$

Note:- $A \to A$, $B \to C$ } that means it is useless because they produce themselves.

Q. $S \to AB$
$A \to a$
$B \to b$
$D \to E$

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

Here D is useless bcoz not reachable from start symbol.

Q. $S \to AB \mid D$
$A \to a$
$B \to b$
$D \to EF$

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

Here symbol D is useless as it does produce any terminal.

classmate
Date
Page

* Removing ϵ (Null) Production :-

Any production of a CFG of the form $A \to \epsilon$ is called ϵ-production. and any variable (non-terminal) for which derivation $A \overset{*}{\to} \epsilon$ is called nullable variable.

Q. Find CFG without ϵ-production equivalent to the grammer defined by.

$S \to ABaC$
$A \to BC$
$B \to b \mid \epsilon$
$C \to D \mid \epsilon$
$D \to d$

nullable
{B, C}

→ from the first step of construction, we find the nullable variable are {A, B, C} Then the required productions using second step is.

$S \to ABaC \mid BaC \mid AaC \mid ABa \mid aC \mid Aa \mid Ba \mid a$
$A \to BC \mid B \mid C$
$B \to b$
$C \to D$
$D \to d$

Q. $S \to aA$
$A \to \epsilon$

→

$S \to aA \mid a$

Q. $S \rightarrow a \mid aA \mid B \mid C$
$A \rightarrow aB \mid \epsilon$
$B \rightarrow Aa$
$C \rightarrow cCD$
$D \rightarrow ddd$

→ Here, C & D are useless symbol because C never gives terminal string and D is not reachable from start symbol S.
So removing useless C & D production

$S \rightarrow a \mid aA \mid B$
$A \rightarrow aB \mid \epsilon$
$B \rightarrow Aa$

Now, removing ϵ productions,

$S \rightarrow a \mid aA \mid B$
$A \rightarrow aB$
$B \rightarrow Aa \mid a$

there is one unit production (i.e $S \rightarrow B$) so can remove it by replacing or substituting.

$S \rightarrow a \mid aA \mid Aa$
$A \rightarrow aB$
$B \rightarrow Aa \mid a$

$S \rightarrow aA \mid aBB$
$A \rightarrow aaA \mid \epsilon$
$B \rightarrow bB \mid bbC$
$C \rightarrow B$

I eliminating useless symbol or production C & B as C is not reachable from S and B doesn't produce any terminal.

∴ By removing useless symbols,

$S \rightarrow aA$
$A \rightarrow aaA \mid \epsilon$

Now, Removing ϵ production.

$S \rightarrow aA \mid a$
$A \rightarrow aaA \mid aa$

teps

① Check useless, if yes Remove.
② Check null, if yes remove.
③ Check useless, if yes remove.
④ Check Unit, if yes remove.
⑤ Check useless, if yes remove.

$S \rightarrow abAB \mid ba$
$A \rightarrow aaa$
$B \rightarrow aA \mid bb$

Q. $S \rightarrow aSa \mid bSb \mid \epsilon$

→ Here, there is no useless production.
Therefore now removing ϵ production,

$S \rightarrow aSa \mid bSb \mid aa \mid bb$.

Q. $S \rightarrow as \mid A \mid c$
$A \rightarrow a$
$B \rightarrow aa$
$C \rightarrow acb$

→ Here, B & C are not reachable from start symbol therefore removing useless symbols.

$S \rightarrow as \mid A \mid c$
$A \rightarrow a$

Removing unit production ($S \rightarrow A$) by substituting value

$S \rightarrow as \mid a \mid c$
$A \rightarrow a$

Here, A is now useless production, as it is not reachable from start symbol so by removing,

$S \rightarrow as \mid a \mid c$

* Normal form of CFG

In a CFG Right hand side of production can be any string of variable and Terminals. When the production in the CFG satisfy certain restriction then CFG is called normal form.

There are two normal form :-

1. Chomsky Normal form (CNF)
2. Greibach Normal form (GNF)

CNF

Any CFG without ϵ-production generated by grammer in which all production are of the form

$$A \rightarrow BC \mid a$$

$$N.T \rightarrow N.T\ N.T \mid T$$

$NT \rightarrow T$

$NT \rightarrow NT\ NT$

Where A, B, C are non-terminal and 'a' is a teminal known as grammer CNF.

Step :- 1. Eliminate ϵ-production from the production.
2. Eliminate unit production from the prod^n.
3. Elimination of teminal on R.H.S.
4. Restricting the no. of variable on R.H.S.

Q. $S \rightarrow AB \mid aB$
$A \rightarrow aab \mid \epsilon$
$B \rightarrow bbA$ into CNF.

→ i) First in the given grammer there is ϵ production so we remove the ϵ production.

$S \rightarrow AB \mid B \mid aB$
$A \rightarrow aab$
$B \rightarrow bbA \mid bb$

ii) Now production has a ~~small~~ unit production so the removing these unit production.

$S \rightarrow AB \mid bbA \mid bb \mid aB$
$A \rightarrow aab$
$B \rightarrow bbA \mid bb.$

iii) Eliminating terminal on R.H.S [We introduce new variables for one terminal present on R.H.S. of production and every terminal on RHS]

$S \rightarrow AB \mid CbCbA \mid CbCb \mid CaB$
$A \rightarrow CaCaCb$
$B \rightarrow CbCbA \mid CbCb$
$Ca \rightarrow a$
$Cb \rightarrow b$

The set of non-terminal in our production.

$\Rightarrow (V, T, P, S)$

$V = \{ S, A, B, Ca, Cb \}$

Now, Restriction in Right hand production to only two variable, by introducing new variable.

$S \rightarrow AB \mid CbD_1 \mid CbCb \mid CaB$
$D_1 \rightarrow CbA$
$A \rightarrow CaD_2$
$D_2 \rightarrow CaCb$
$B \rightarrow CbD_3 \mid CbCb$
$D_3 \rightarrow CbA$
$Ca \rightarrow a$
$Cb \rightarrow b$

Hence, Required VTPS

$V = \{ S, A, B, Ca, Cb, D_1, D_2, D_3 \}$

$T = \{ a, b \}$

- S = Start symbol.

Q. $S \rightarrow bA \mid aB$
$A \rightarrow bAA \mid aS \mid a$
$B \rightarrow aBB \mid bS \mid b$
Find an equivalent grammer in ∈ CNF.

→
- Here no ∈ production.
- Here no unit production.
- So we eliminating terminal on R.H.S.

$S \rightarrow CbA \mid CaB$
$A \rightarrow CbAA \mid CaS \mid a$
$B \rightarrow CaBB \mid CbS \mid b$
$Ca \rightarrow a$
$Cb \rightarrow b$

The set of non-terminal is $\{S, A, B, Ca, Cb\}$

Now, Restriction in R.H.S. production only two variable by introducing new variable.

$S \rightarrow CbA \mid CaB$
$A \rightarrow CbD_1 \mid CaS \mid a$
$D_1 \rightarrow AA$
$B \rightarrow CaD_2 \mid CbS \mid b$
$D_2 \rightarrow BB$
$Cb \rightarrow b$
$Ca \rightarrow a$

where,
$V = \{S, A, D_1, B, D_2, Cb, Ca\}$
$T = \{a, b\}$
S = Start symbol.

Q. $S \rightarrow aS \mid aSbS \mid \epsilon$ construct CNF form.

→ i) Eliminating ϵ production

$S \rightarrow aS \mid aSbS \mid a \mid aSb \mid abS \mid ab$

ii) There is no unit production in above grammer.

iii) Now, eliminating terminal on R.H.S.

$S \rightarrow CaS \mid CaSCbS \mid a \mid CaSCb \mid CaCbS \mid CaCb$

$Ca \rightarrow a$

$Cb \rightarrow b$

iv) Now, restriction on R.H.S production, only two variable are available.

$S \rightarrow CaS \mid CaD_1 \mid a \mid CaD_3 \mid CaD_2 \mid CaCb$

$D_1 \rightarrow (SCbS)$ i.e SD_2

$D_2 \rightarrow CbS$

$D_3 \rightarrow SCb$

$Ca \rightarrow a$

$Cb \rightarrow b$

∴ The set of variable

$V = \{S, D_1, D_2, D_3, Ca, Cb\}$

$T = \{a, b\}$

Q. $S \to \sim S \mid [S \supset S] \mid p \mid q$ construct CNF form.

in the given problem

where,

$V = \{S\}$

$T = \{\sim, [, \supset,], p, q\}$

S = Starting symbol

P = productions

→ Here no ϵ production, no unit production.

So, we eliminating the terminal on R.H.S.

$S \to C_\sim S \mid C_[S C_\supset S C_] \mid p \mid q$

$C_\sim \to \sim$

$C_[\to [$

$C_\supset \to \supset$

$C_] \to]$

Now, arranging R.H.S production

$S \to C_\sim S \mid C_[D_1 \mid p \mid q$

$D_1 \to S D_2$

$D_2 \to C_\supset D_3$

$D_3 \to S C_]$

$C_\sim \to \sim$

$C_[\to [$

$C_\supset \to \supset$

$C_] \to]$

Q. $S \rightarrow aAbB$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$

→ There is no ϵ production & ~~$S \rightarrow B$~~ ~~A→B~~ unit production.
So, we Remove terminal on R.H.S.

$S \rightarrow CaACbB$
$A \rightarrow CaA \mid a$
$B \rightarrow CbB \mid b$
$Ca \rightarrow a$
$Cb \rightarrow b$

Now, restriction on CNF on R.H.S.

$S \rightarrow CaD_1$
$D_1 \rightarrow AD_2$
$D_2 \rightarrow CbB$
$A \rightarrow CaA \mid a$
$B \rightarrow CbB \mid b$
$Ca \rightarrow a$
$Cb \rightarrow b$

Now,
S = Start Symbol
$V = \{S, D_1, D_2, A, B, Ca, Cb\}$
$T = \{a, b\}$
P = productions.

Q. $S \rightarrow ABa$
$A \rightarrow aab$
$B \rightarrow Ac$

→ Here, no useless & no unit- production and no ϵ production

Now,

$S \rightarrow ABCa$
$A \rightarrow CaCaCb$
$B \rightarrow ACc$
$Ca \rightarrow a$
$Cb \rightarrow b$
$Cc \rightarrow c$

Applying restrictions

$S \rightarrow AD_1$
$D_1 \rightarrow BCa$
$A \rightarrow CaD_2$
$D_2 \rightarrow CaCb$
$B \rightarrow ACc$
$Ca \rightarrow a$
$Cb \rightarrow b$
$Cc \rightarrow c$

Now,

S = Start Symbol
P = productions
V = { S, D_1, A, D_2, B, Ca, Cb, Cc }
T = { a, b, c }

* Greibach Normal Form (GNF)

- Every context free language L without ε can be generated by a grammer for which every production is of the form.

$$A \rightarrow a\alpha$$

Where, N.T T

NT → T
NT → T NT NT ..... NT

A is non terminal
a is terminal
α is string of non-terminal or it may be empty
Then this normal form is known as Greibach Normal form (GNF).

- Thus in this normal form right hand side of any production will contain one terminal followed by zero or more non terminal.

Q. Convert into GNF, $S \rightarrow ab / aS / aaS$

→

$S \rightarrow ab$ & $S \rightarrow aaS$ are not GNF.
Now, we can introduce new variable for terminal a & b and substitute that variable at the appropriate place.

$S \rightarrow aCb / aS / aCaS$
$Ca \rightarrow a$
$Cb \rightarrow b$

Now all productions are in GNF
Hence, required GNF tuples are -

$V = \{S, Ca, Cb\}$
$T = \{a, b\}$
$P = \{S \to aCb \mid as \mid aCaS\}$
$Ca \to a$
$Cb \to b$
S = Start symbol

3. Convert following grammer into GNF.

$S \to AB$
$A \to aA \mid bB \mid b$
$B \to b$

→ Substitute production A in production S, so we get,

$S \to aAB \mid bBB \mid bB$
$A \to aA \mid bB \mid b$
$B \to b$

∴ It is in the GNF form

Q. $S \to abSb \mid aa$

→ The given production S is not a form of GNF
so,

$S \to aCbSCb \mid aCa$
$Ca \to a$
$Cb \to b$

Q. $S \rightarrow aSb \mid bSa \mid a \mid b$

→ $S \rightarrow CaSCb \mid bSCa \mid a \mid b$

$Ca \rightarrow a$

$Cb \rightarrow b$

Q. $S \rightarrow AA \mid 0$

$A \rightarrow SS \mid 1$

→

* Lemma Rule

Production A_2 begins with the same numbered variable
Hence introduced B_2 for A_2

We apply lemma rule when $A \to B$, $B \to A$ form exist.

Lemma Rule

$$A \to A\alpha \mid \beta$$

then

$$A \to \beta \mid \beta B$$
$$B \to \alpha \mid \alpha B$$

Q. ~~B2~~ $S \to AA \mid 0$
$A \to SS \mid 1$

→ Substituting S in A

$$S \to AA \mid 0$$
$$\underset{A}{\underline{A}} \to \underset{A}{\underline{A}}\underset{\alpha}{AS} \mid \underset{\beta}{0S} \mid \underset{\beta}{1}$$

By using lemma rule,

$$A \to A\alpha \mid \beta$$

then

$$A \to \beta B$$
$$B \to \alpha \mid \alpha B$$

Therefore,

$S \rightarrow AA|0$

$A \rightarrow 0S|1|0SB|1B$

$B \rightarrow AS|ASB$

Now, production A is in the form of GNF. So putting in B

$S \rightarrow AA|0$

$A \rightarrow 0S|1|0SB|1B$

$B \rightarrow 0SS|1S|0SBS|1BS|0SSB|1SB|0SBSB|1BSB$

Now, S is not GNF so put A in S.

$S \rightarrow 0SA|1A|0SBA|1BA|0$

$A \rightarrow 0S|1|0SB|1B$

$B \rightarrow 0SS|1S|0SBS|1BS|0SSB|1SB|0SBSB|1BSB$

Now, all the production S, A, B are properly in GNF form

Q. $S \rightarrow SS|0S1|01$

→

$S \rightarrow SS|0SC_1|0C_1$

$C_1 \rightarrow 1$

A ∝ β

Now, by using Lemma Rule,

$A \rightarrow A\alpha|\beta$

then

$A \rightarrow \beta|\beta B$

$B \rightarrow \alpha|\alpha B$

$S \to OSC_1 \mid OC_1 \mid OSC_1B \mid OC_1B$

$B \to S \mid SB$

$C_1 \to 1$

Now, substituting S in B

$S \to OSC_1 \mid OC_1 \mid OSC_1B \mid OC_1B$

$B \to OSC_1 \mid OC_1 \mid OSC_1B \mid OC_1B \mid OSC_1B \mid OC_1B \mid OSC_1BB \mid$ ~~OSC~~ OC_1BB.

$C_1 \to 1$

Convert to GNF

$G = (\{A_1, A_2, A_3\}, \{a, b\}, P, A_1\}$ where,

P consist of

$A_1 \to A_2A_3$

$A_2 \to A_3A_1 \mid b$

$A_3 \to A_1A_2 \mid a$

Substitute A_1 into A_3

$A_1 \to A_2A_3$

$A_2 \to A_3A_1 \mid b$

$A_3 \to A_2A_3A_2 \mid a$

Now, Substitute A_2 into A_3

$A_1 \to A_2A_3$

$A_2 \to A_3A_1 \mid b$

$A_3 \to A_3A_1A_3A_2 \mid a \mid bA_3A_2$

A_3 : A → A α | β

By lemma Rule,

$$A \to A\alpha \mid \beta$$

then

$$A \to \beta \mid \beta B$$
$$B \to \alpha \mid \alpha B$$

$\therefore$ $A_1 \to A_2A_3$
$A_2 \to A_3A_1 \mid b$
$A_3 \to bA_3A_2 \mid a \mid bA_3A_2B \mid aB$
$B \to A_1A_3A_2 \mid A_1A_3A_2B$.

Now, A_3 is in GNF form, so substituting in A_2

$A_1 \to A_2A_3$
$A_2 \to bA_3A_2A_1 \mid aA_1 \mid bA_3A_2BA_1 \mid aBA_1 \mid b$
$A_3 \to bA_3A_2 \mid a \mid bA_3A_2B \mid aB$
$B \to A_1A_3A_2 \mid A_1A_3A_2B$

Now, A_2 is in GNF form
$\therefore$ Substituting A_2 & ~~A_3~~ in A_1

$A_1 \to bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3A_2BA_1A_3 \mid aBA_1A_3 \mid bA$
$A_2 \to bA_3A_2A_1 \mid aA_1 \mid bA_3A_2BA_1 \mid aBA_1 \mid b$
$A_3 \to bA_3A_2 \mid a \mid bA_3A_2B \mid aB$
$B \to A_1A_3A_2 \mid A_1A_3A_2B$

Now, A_1, A_2 & A_3 are in GNF form
$\therefore$ Substituting A in B

$A_1 \rightarrow bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3A_2BA_1A_3 \mid aBA_1A_3 \mid bA_3$

$A_2 \rightarrow bA_3A_2A_1 \mid aA_1 \mid bA_3A_2BA_1 \mid aBA_1 \mid b$

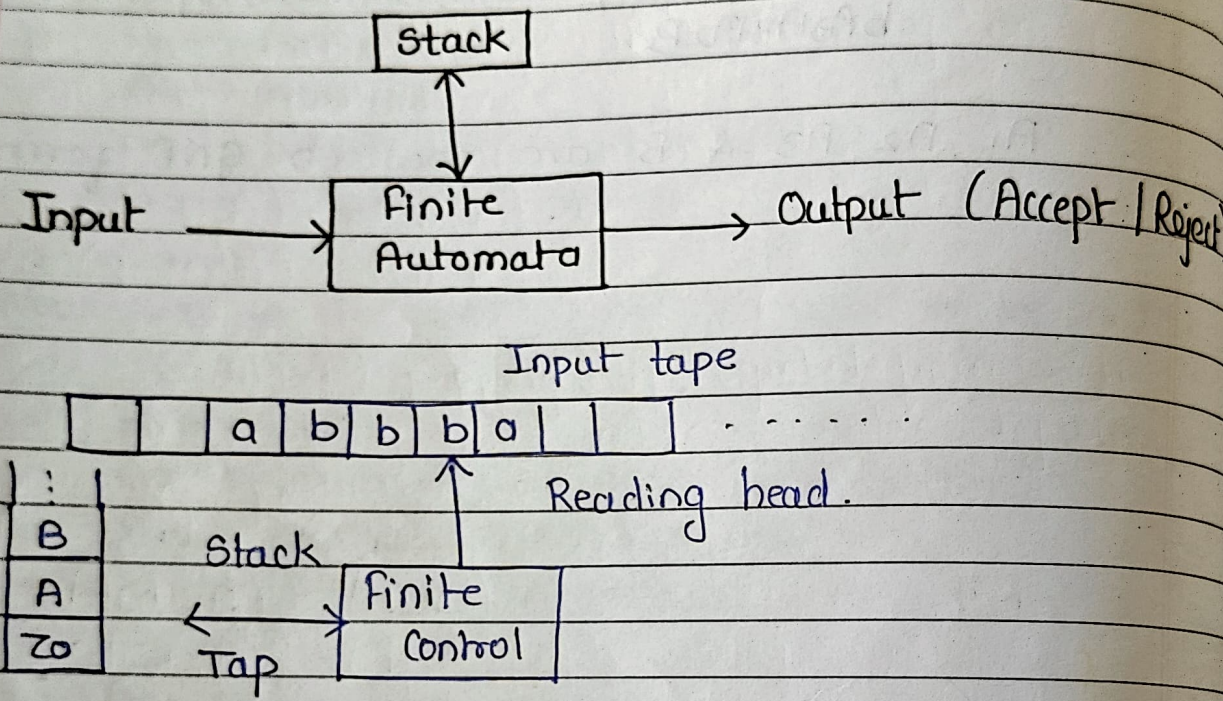
$A_3 \rightarrow bA_3A_2 \mid a \mid bA_3A_2B \mid aB$

$B \rightarrow bA_3A_2A_1A_3A_3A_2 \mid aA_1A_3A_3A_2 \mid bA_3A_2BA_1A_3A_3A_2$
$aBA_1A_3A_3A_2 \mid bA_3A_3A_2 \mid bA_3A_2A_1A_3A_3A_2B \mid$
$aA_1A_3A_3A_2B \mid bA_3A_2BA_1A_3A_2B \mid aBA_1A_3A_3A_2B \mid$
$bA_3A_3A_2B$

$\therefore$ A_1, A_2, A_3 & B are now in GNF form.

* Push - Down Automata

Push down automata is nothing but finite state machine which represent CFG.
The block diag. of PDA is as follows :-



Mathematically PDA is defined as seven tuples as -

$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, z_0, F \rangle$$

where,

- Q - finite set of states
- Σ - finite set of input symbols.
- Γ - finite set of stack symbols.
- δ - mapping function. $Q \times (\Sigma \cup \{\epsilon\}) * \Gamma \rightarrow Q$.
- q_0 - initial state of PDA
- z_0 - start symbol of stack
- F - finite set of final state.

- Z_0 represents the bottom symbol of the stack or it is the start position of PDA Z_0 will be the stack top symbol.

- Each move of the control unit reads a symbol from the input tape, while at the same time changing the contents of the stack through the usual stack operations.

- Each move of the control unit is determined by the current input symbol as well as by the symbol currently on top of the stack.
The result of the move is a new state of the control unit & the chance in the top of the stack.

$011 \subseteq 110$ or $110 \subseteq 011$

Q. Construct a PDA for $L = \{\omega c \omega^R : \omega \in (0+1)\}$

→ After getting first symbol ('0' or '1') we will push 'A' and 'B' respectively onto the stack using following production. In this time stack top will contain i.e Z_0.

$\delta(q_0, 0, Z_0) = (q_0, AZ_0)$
$\delta(q_0, 1, Z_0) = (q_0, BZ_0)$

Next symbol will have four possibilities as follows:-

$0 \begin{cases} 0 & = 00 \\ 1 & = 01 \end{cases}$

$1 \begin{cases} 0 & = 10 \\ 1 & = 11 \end{cases}$

	in stack
$\delta(q_0, 0, A) = (q_0, AA)$	00
$\delta(q_0, 0, B) = (q_0, AB)$	01
$\delta(q_0, 1, A) = (q_0, BA)$	10
$\delta(q_0, 1, B) = (q_0, BB)$	11

Then we will get seperator 'c', after getting this we will change the state by using following production.

$$\delta(q_0, c, A) = (q_1, A)$$
$$\delta(q_0, c, B) = (q_1, B)$$

Now, we start popping off symbol from S. the stack we find string in reverse order by using following production.

$$\delta(q_1, 0, A) = (q_1, \epsilon)$$
$$\delta(q_1, 1, B) = (q_1, \epsilon)$$

At last stack will be empty if string will be in exact reverse order so stack top will be Z_0, and string will be exhausted. we will make stack empty using following production.

$$\delta(q_1, \epsilon, Z_0) = (q_1, \epsilon)$$

Because w may be empty string so tape will contain only 'c' so we need another production to make stack empty.

$$\delta(q_0, c, Z_0) = (q_1, \epsilon)$$

Q. Construct PDA for $L = \{ WW^R : W \in \{a+b\}^+ \}$ (1001 or 0110)

→ The string formed by a's and b's named as W that will be formed by reverse of 'W'.

$M = (\{q_0, q_1\}, \{a, b\}, \{A, B, Z_0\}, \delta, q_0, Z_0, \phi)$

After getting first symbol ('a' or 'b') we push 'A' & 'B' respectively onto the stack using following productions.
In this, the stack top will contain top symbol i.e Z_0

$\delta(q_0, a, Z_0) = (q_0, AZ_0)$
$\delta(q_0, b, Z_0) = (q_0, BZ_0)$

In the given language, there is no seperator bet^n 'W' & 'W^R'.
So whenever symbol 'a' will appear on the tape & stack top will contain 'A'.
We have two possibilities,
- One according to 'W' (push) and
- One according to 'W^R' (pop)

and similarly for B...

So next symbol will have 4 possibilities.

$\delta(q_0, a, A) = \{(q_0, AA), (q_1, \epsilon)\}$
$\delta(q_0, b, A) = (q_0, BA)$
$\delta(q_0, a, B) = (q_0, AB)$
$\delta(q_0, b, B) = \{(q_0, BB), (q_1, \epsilon)\}$

Now, we start popping off symbols from the stack if we find string in reverse order, by using following production.

$$\delta(q_1, a, A) = \{(q_1, \epsilon)\}$$
$$\delta(q_1, b, B) = \{(q_1, \epsilon)\}$$

At last stack will be empty if string will be in exact reverse order.
So top of stack will be z_0, and string will exhausted.

$$\delta(q_1, \epsilon, z_0) = (q_1, \epsilon)$$

Q. $L = \{ww^R : w \text{ is } (0+1)^*\}$

→ Same as above example. only ϵ is allowed in this language.
Then we have to add one extra production.

$$\delta(q_0, \epsilon, z_0) = (q_1, \epsilon)$$

Q. Construct a PDA for $L = \{0^n 1^n \mid n \geq 0\}$ 0011

→ $L^* = \{\epsilon, 01, 0011, 000111, \ldots\}$

For first 0
$\delta(q_0, 0, Z_0) = (q_0, AZ_0)$

For Remaining 0
$\delta(q_0, 0, A) = (q_0, AA)$

For first 1
$\delta(q_0, 1, A) = (q_1, \epsilon)$

For Remaining 1
$\delta(q_1, 1, A) = (q_1, \epsilon)$

For stack empty (for Z_0)
$\delta(q_1, \epsilon, Z_0) = (q_2, \epsilon)$

For ϵ from L^*
from Initial q_0 to final q_2
$\delta(q_0, \epsilon, Z_0) = \{q_2, \epsilon\}$

Q. Construct a PDA for $L = \{a^n b^m c^{n+m} \mid m, n \geq 1\}$

→ $L^* = \{abcc, aabbcccc, \ldots\ldots\}$

$\delta(q_0, a, Z_0) = (q_0, AZ_0)$ For first A

$\delta(q_0, a, A) = (q_0, AA)$ for remaining A

$\delta(q_0, b, A) = (q_1, AA)$ For first B

$\delta(q_1, b, A) = (q_1, AA)$ For remaining B

$\delta(q_1, c, A) = (q_2, \epsilon)$ For first C

$\delta(q_2, c, A) = (q_2, \epsilon)$ For remaining C

$\delta(q_2, \epsilon, Z_0) = (q_3, \epsilon)$ Stack empty.

Q. Construct a PDA for $L = \{a^n b^n \mid n \geq 1\}$

→ $L^* = \{ab, aabb, aaabbb, \ldots\}$