★ <u>Context Free Grammer</u> (CFG) : G

A CFG is a 4 tuple

$G = (V, \Sigma, S, P)$

productions / set of Productions

Variables / Non terminal symbol $\{A, B, C\}$

Terminal symbols $\{0, 1\}$ $\{a, b\}$

starting symbol $S \in V$

① <u>Variables</u> : V

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

Non terminal symbols

② <u>Terminal</u> : Σ

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

Leaf Nodes in tree.

③ <u>S</u> — starting symbol

$S \rightarrow A$

$S \rightarrow Aa$

④ <u>Productions</u> →

$$\left.\begin{array}{l} A \rightarrow ab \\ B \rightarrow Ca \\ C \rightarrow \Lambda \end{array}\right\} \text{set of Productions}$$

Draw the regular Language of the following $L = \{0,1\}^* \{10\}$

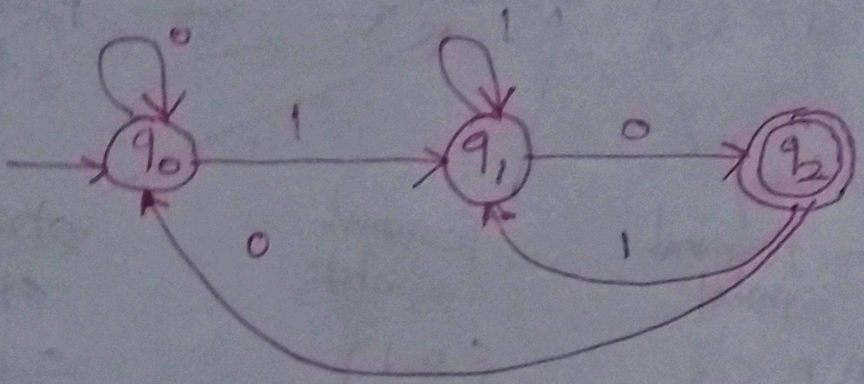


Fig:- DFA for $\{0,1\}^* \{10\}$

Consider a Language which is accepted by given DFA ie

$w = 1100010$

Initial strings	State
Λ	q_0
1	q_1
11	q_1
110	q_2
1100	q_0
11000	q_0
110001	q_1
1100010	q_2

Now Convert given table into CFG form

$q_0 \to 1 q_1$ / $q_0 \xrightarrow{1} q_1$

$q_1 \to 1 q_1$ / $q_1 \xrightarrow{1} q_1$

$q_1 \to 0 q_2$ / $\boxed{q_1 \xrightarrow{0} q_2}$

So it having shorthand notation

$q_0 \to a q_1$

where $\boxed{a \in \Sigma}$ &

q_0 & q_1 are non terminal symbol

$\boxed{q_0, q_1 \in V}$

when last production is their then

we can take $q_0 \to b$

$\forall$ $\boxed{q_0 \in V}$ $\boxed{b \in \Sigma}$

$\forall$ V=N. Terminal symbol

Σ = Terminal symbol.

* Union, Concatenation & Kleene on CFG

① **Union** :- If L_1 & L_2 are two context freee Language which having

$G_1 = (V_1, \Sigma, S_1, P_1)$

$G_2 = (V_2, \Sigma, S_2, P_2)$

Union of G_1 & G_2 will be noted as

$\boxed{G_u = G_1 \cup G_2}$

$\boxed{L(G_u) = L_1 \cup L_2}$

$G_u = (V_u, \Sigma, S_u, P_u)$

$\forall$ $\boxed{V_U = V_1 \cup V_2 \cup \{S_U\}}$

$S_U = S_U$ starting symbol in new/union CFL

$\boxed{P_U = P_1 \cup P_2 \cup \{S_U \rightarrow S_1 | S_2\}}$

Concatenation

We have G_1 & G_2 as two CFG

$G_c = (V_c, \Sigma, S_c, P_c)$

$\boxed{V_c = V_1 \cup V_2 \cup \{S_c\}}$

N.T. / variables

S_c = start symbol in G_c

$\boxed{P_c = P_1 \cup P_2 \cup \{S_c \to S_1 S_2\}}$

Kleene*

$G_1 = (V_1, \Sigma, S_1, P_1)$

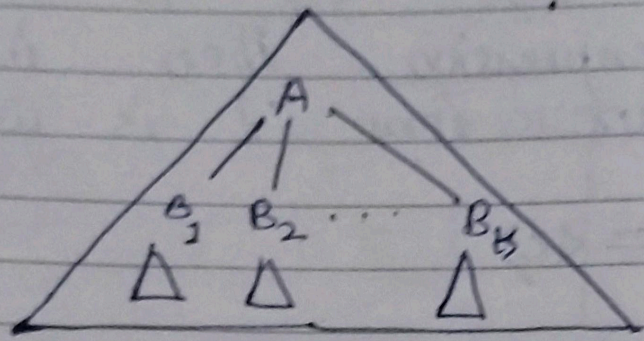
then $G_K = (V_K, \Sigma, S_K, P_K)$

$\boxed{V_K = V_1 \cup \{S_K\}}$

S_K = New starting symbol

$\boxed{P_K = P_1 \cup \{S_K \to S_1 S_K \mid \Lambda\}}$

By using Chomsky Normal form



☆ * Union Concatenation & *'s of CFL's

G = V, Σ, S, P

V, Σ/T, S, P

V = set of variables

T = Terminal Symbol

S = starting symbol

P = Production

G = { {A,S,B}, {a,b,c,d}

{ S → A

A → B

B → ab

S → c }

}

If L_1 & L_2 are CFL then $L_1 \cup L_2$, $L_1 \cdot L_2$, L_1^* also CFL.

Proof : By Constructive method.

Consider $G_1 = \{V_1, \Sigma, S_1, P_1\}$ &

$G_2 = \{V_2, \Sigma, S_2, P_2\}$ generating L_1 & L_2 respectively

① **Union** = $G_u = \{V_u, \Sigma, S_u, P_u\}$

generating $L_1 \cup L_2$.

If necessary we rename the elements of V_2 then

$V_1 \cap V_2 = \phi$

$V_u = V_1 \cup V_2 \cup \{S_u\}$

$\forall$ S_u = New start symbol not in V_1 & V_2

$P_u = P_1 \cup P_2 \cup \{S_u \rightarrow S_1 | S_2\}$

if x is in L_1 or L_2 then

$S_u \overset{*}{\Rightarrow} x$ in grammar G_u

because we start the derivation in

$S_u \to S_1$ or $S_u \to S_2$ & continue the derivation of x in G_1 or G_2

$\therefore L_1 \cup L_2 \subseteq L(G_u)$

If x is derivable from S_u in G_u the first step in any derivation must be

$S_u \Rightarrow S_1$ or $S_u \Rightarrow S_2$

In the first case all subsequent productions are must be production in G_1 because no variables in V_2 are involved. & thus $x \in L_1$ & second case

$x \in L_2$

$$\boxed{L(G_u) \subseteq L_1 \cup L_2}$$

ii) <u>Concatenation</u>

A grammer $G_c = (V_c, \Sigma, S_c, P_c)$ generating $L_1 \cdot L_2$

$\forall \quad V_1 \cap V_2 = \phi$

$V_c = V_1 \cup V_2 \cup \{S_c\}$

$P_c = P_1 \cup P_2 \cup \{S_c \to S_1 \cdot S_2\}$

if $x \in L_1 L_2$ then $x = x_1 x_2 \ \forall \ x_i \in L_i$ for i we may then derive x in G_c as follows

$$S_c \Rightarrow S_1 S_2 \Rightarrow^* x_1 S_2 \Rightarrow^* x_1 x_2 = x$$

x_1 in G_1 &

x_2 in G_2

if x is derived from S_c then $S_c \Rightarrow S_1 \cdot S_2$

x must be derived from $x = x_1 x_2$

$\forall$ for each i, x_i can be derived from S_i in G_c

$\because V_1 \cap L_2 = \phi$, being derivable from S_i in G_c means being derivable from S_i in G_i & so

$$\boxed{x \in L_1 L_2}$$

iii) **Kleene** :-

generating L_1^* A grammer $G^* = \{V, \Sigma, S, P\}$

$V = \{V_1 \cup \{S_*\}\}$

$\forall \; S_1 \notin V_1$

L_1^* contains string of the form

$x = x_1 x_2 \dots x_K$

$\forall \; x_i \in L_1$

Each x_i is desired from S_1, then to desire x from S_* it is enough to be able to desire a string of K S_1's

$S_* \rightarrow S_1 S \mid \Lambda$ in P

$\therefore \; P = P_1 \cup \{S_* \rightarrow S_1 S \mid \Lambda\}$

If $x \in L(G^*)$ then on other hand $x = \Lambda$ or x can be desired from some string of the form

S_1^K in G^*, 2nd case only productions in G^* beginning with S_1 are those G_1 we conclude

$$\boxed{x \in L(G_1)^K \subseteq L(G_1)^*}$$

Chapter 3
Grammers & Languages

Chapter - III

☆ * Grammer & Context free Grammer

Grammer - - set of formal rules
- checks correctness of sentence or syntax
- FA & Grammer are classified by chomsky

* Context free grammer (CFG)

- A Context free grammer G is 4 tuple

$G = (V, \Sigma, S, P)$

where

$V \Rightarrow$ variables, Non Terminal symbol
$\Sigma \Rightarrow$ Terminal symbol $\{a, b\}$ $\{0, 1\}$ etc
$S \Rightarrow$ start symbol
$P \Rightarrow$ productions in grammer.
$S \to A$, $S \to a$, $A \to b$, $A \to \epsilon$

- If L_1 & L_2 are context free language then it will give
1) Union
2) Concatnation
3) Kleene

$G_1 = (V_1, \Sigma, S_1, P_1)$ & $G_2 = (V_2, \Sigma, S_2, P_2)$

then

$G_U = (V_U, \Sigma, S_U, P_U)$
$G_C = (V_C, \Sigma, S_C, P_C)$
$G^* = (V, \Sigma, S, P)$

Regular Grammers

A Grammer $G=(V, \Sigma, S, P)$ is regular if every production take one of two forms

$B \to aC$

$B \to a$

$\forall$ B, C $\Rightarrow$ are variables / Non Terminal Symbols

a $\Rightarrow$ Terminal Symbol.

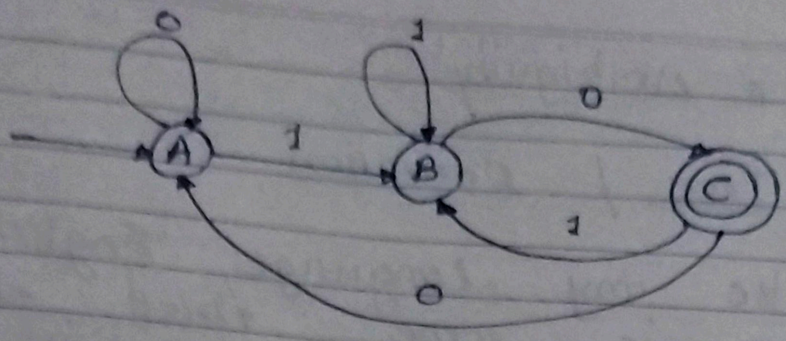
Eg:- Draw the regular language of following $L = \{0,1\}^*\{10\}$

ie set of all string over $\{0,1\}$ & end at 10

Let take $x = \underline{110010}$ ie string end with 10.

Substring Proceed so far	State.
Λ	A
1	B
11	B
110	C
1100	A
11000	A
110001	B
1100010	C.

state or string end with 10 Contain Accepting or C state.



If we observe the lines of table then

$A \Rightarrow 1B \Rightarrow 11B \Rightarrow 110C \Rightarrow 1100A \Rightarrow 11000A$
$\Rightarrow 110001B \Rightarrow 1100010C.$

If we convert in Grammer then it will

$A \rightarrow 1B$
$B \rightarrow 1B$
$B \rightarrow 0C$
$C \rightarrow 0A$
$A \rightarrow 0A$
$A \rightarrow 1B$
$B \rightarrow 0C$

This of the form

$P \rightarrow aQ \quad \forall \quad a \in \{0, 1\}$

$P \rightarrow aQ \Rightarrow \boxed{P \xrightarrow{a} Q} \rightarrow$ single production

$\underline{\underline{110C \Rightarrow 1100}} \Rightarrow$ More than 1 Production

★ 3.1★ Derivation & Ambiguity

Derivation Tree / Derivation

- like any language English or other there is some fixed structure like that in CFG it contain also fixed pattern called Derivation tree or parse tree.

Rules

1) Root node in parse / derivation tree is always labeled with start symbol.

2) All the terminal symbols are labeled with some terminal symbols of the grammer so nodes are called terminal or leaf node

3) If string contain n symbols then it arranged from left to right.

4) Non terminal symbols labbled with nonterminal string.

5) If the string contain $\{\wedge\}, \epsilon, \lambda$ empty symbol then it will be ignored.

Eg:- 1) If the Grammer $G = \{V, \Sigma, S, P\}$ in such a way that.

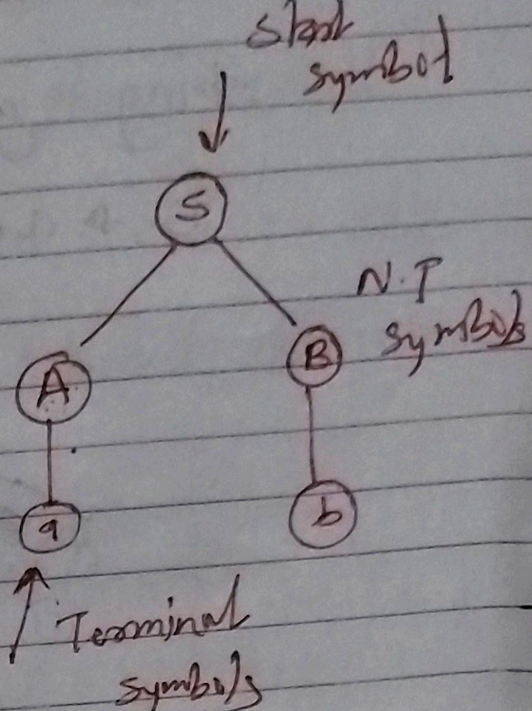
① $V = \{S, A, B\}$ → variables

② $T/\Sigma = \{a, b\}$ → terminal symbol

③ $P \Rightarrow \{ S \rightarrow AB$ ✓ productions
$A \rightarrow a$ ✓
$B \rightarrow b$ ✓
$\}$

④ $S \Rightarrow$ starting symbol

Derivation Tree



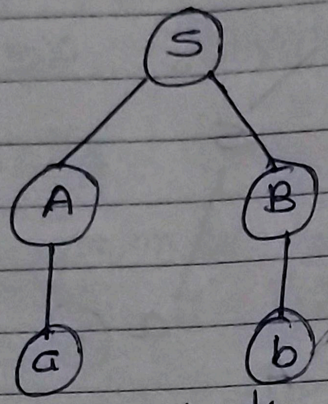


Fig:- Derivation tree for string ab

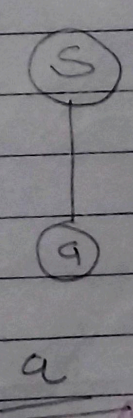
Eg 2) Draw the derivation tree for string aabbaa using following grammar

① $G = (V_n, \Sigma, P, S)$

② $V_n = \{S, A\}, \Sigma = \{a, b\}$

③ $P = \{$
$S \rightarrow aAS$ ✓
$A \rightarrow SbA$ ✓
$S \rightarrow a$ ✓
$A \rightarrow ba$ ✓
$\}$

④ S — starting symbol



Now s is start symbol so any string generated will stored at s

$S \rightarrow aAS$

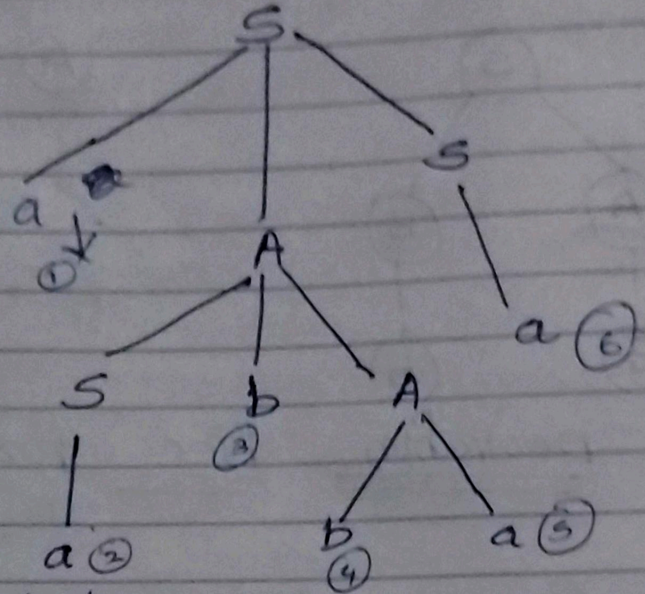
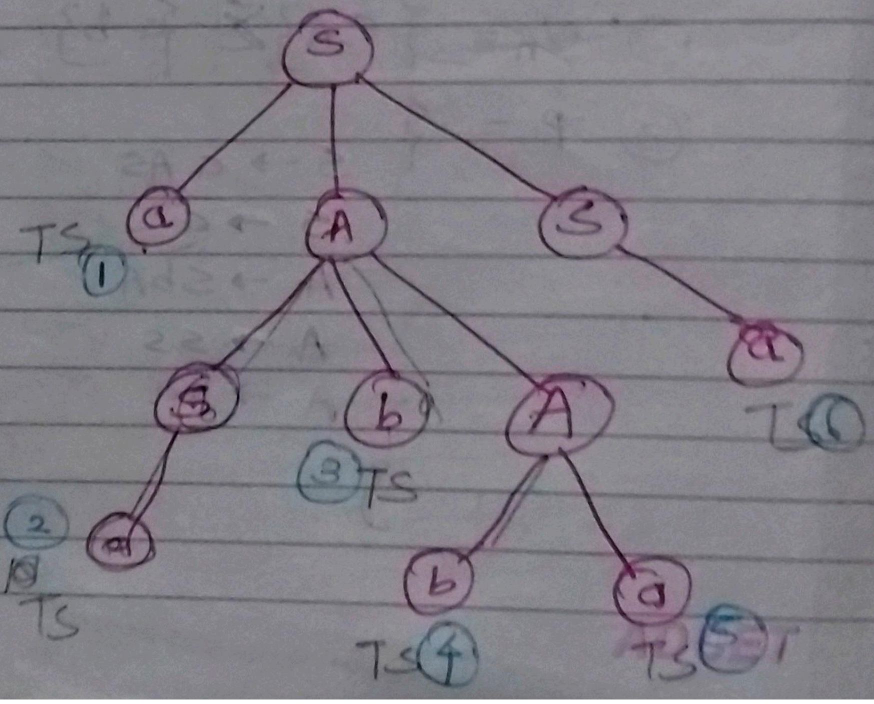


Fig:- Derivation Tree

Reading from left the string is $\underline{\underline{aabbaa}}$ hence Ans.

① $S \rightarrow aAS$



$\boxed{aabbaa}$

5

★ **Ambiguity / Ambiguous Grammar**

In case of CFG they called as Ambiguous if,

i) It contain terminal string with more than one parse tree

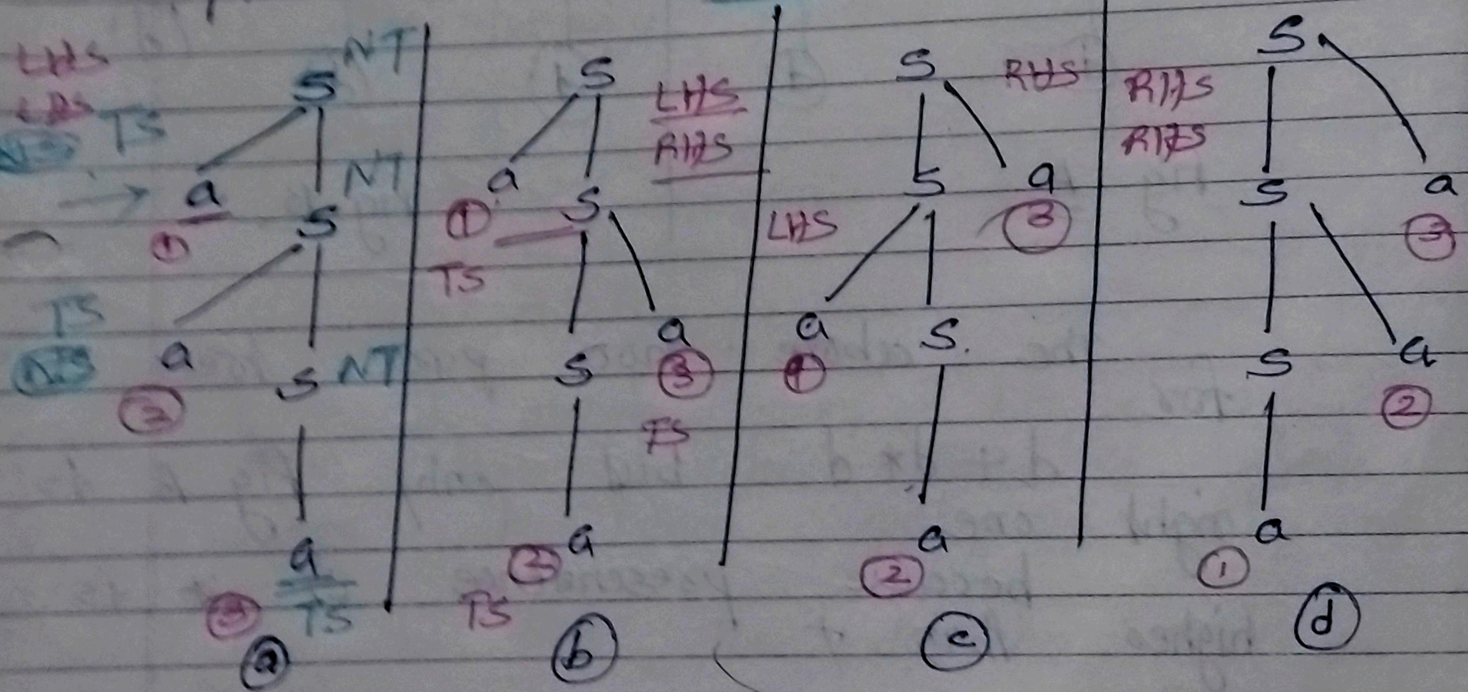
ii) more than one leftmost derivation or leftmost derivation

eg. i) If $G = (V, \Sigma, P, S)$ be a grammar with

$a = (\{S\}, \{a\}, P, S\}$

| $V = \{S\}$
| $\Sigma / T = \{a\}$

$\boxed{P = \{S \to aS \mid Sa \mid a\}}$ find out parse tree for aaa.



by Above 4 way we can represent the aaa so it is ambiguous grammer

ADCET

Eg 2) If $G = (V, \Sigma, S, P)$ be grammer with

$G = (\{S\} \{+, *, d\}, P, S\}$

$P = \{S \to S + S * d\}$

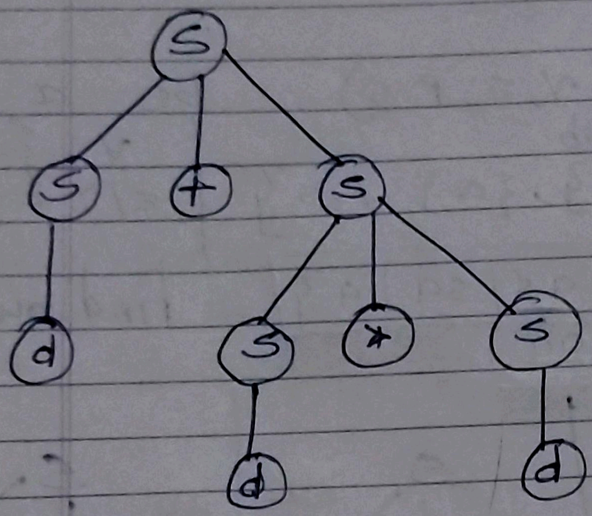


Fig A

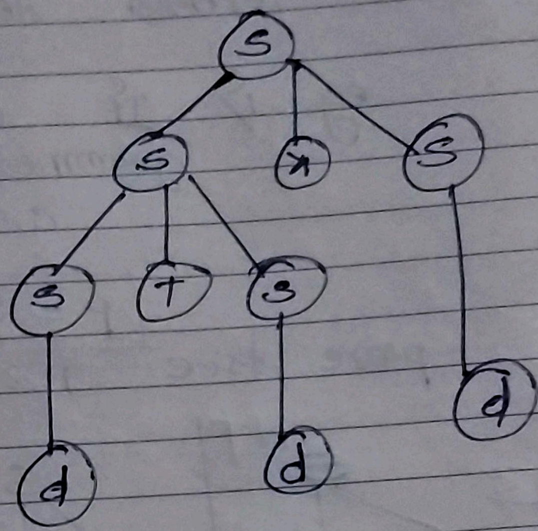


Fig B

The above two parse tree for $d + d * d$ but only Fig A is right one because presendence of $*$ is higher than $+$

V = {E}

Ambiguous Grammer

If given grammer having more than one derivation tree then it is called as ambiguous grammer.

G (Given Grammer)

DT_1 DT_2

Derivation Tree 1

Derivation Tree 2

eg. The following grammer.

$S \rightarrow A$

$S \rightarrow bSa$

$A \rightarrow \epsilon$

$A \rightarrow AcA$

It accepts string bbcccaa draw the derivation tree & check weather it having ambiguous or not.

Now to find the derivation tree we having

- left derivation
- Right derivation

Now we can draw following trees.

String ⇒ bbcccaa

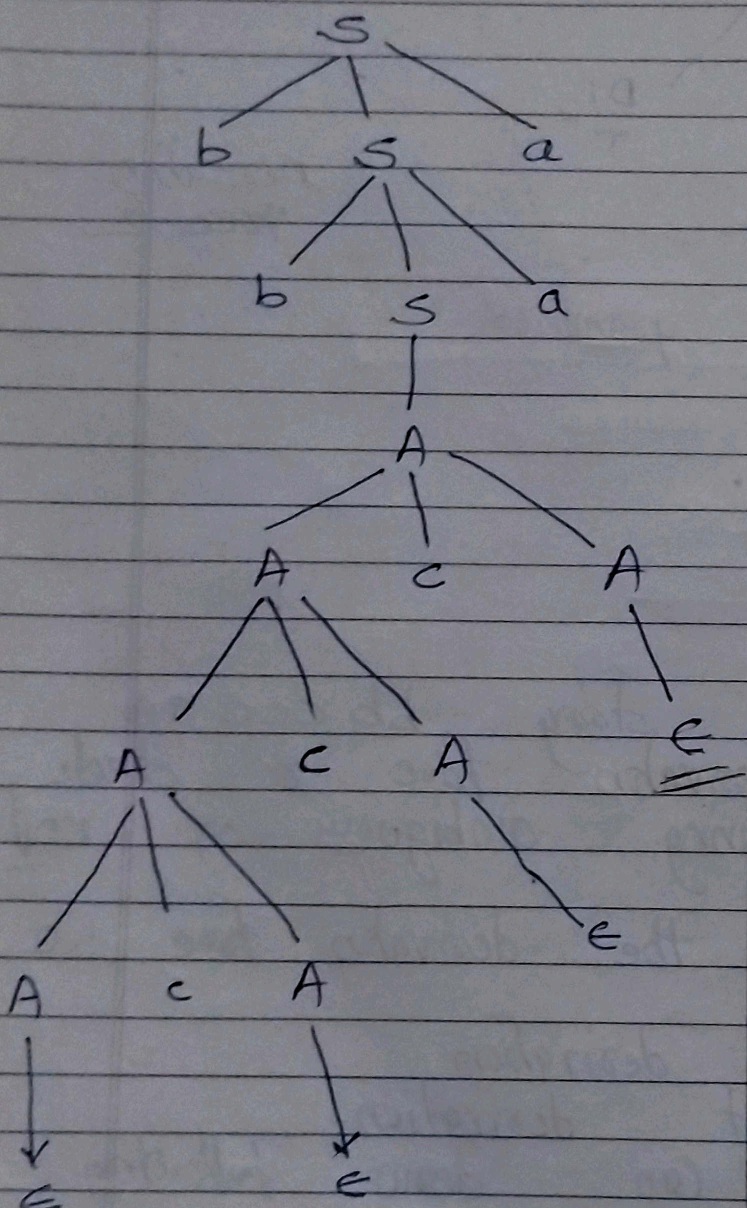
$S \to A$

$S \to bSa$

$A \to \epsilon$

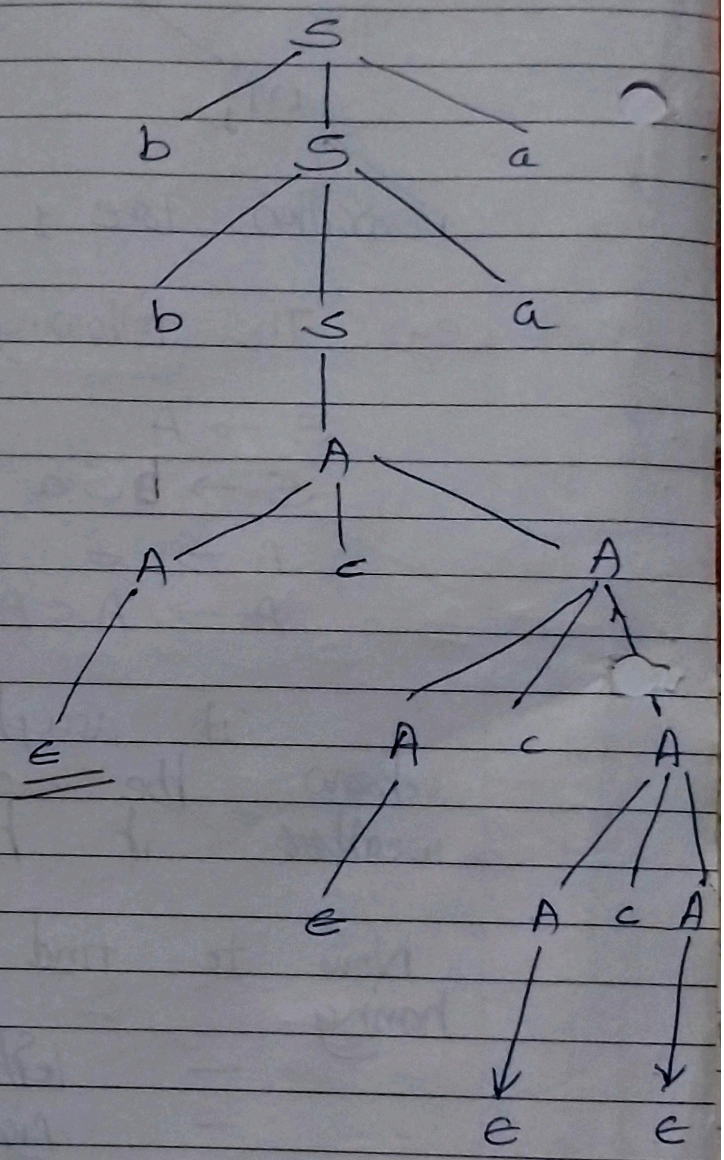
$A \to AcA$

Tree No : 01



bbcccaa Accepting

Tree No : 02



bbcccaa Accepting

* $V = \{E\}$

$\Sigma = \{id\}$

$S = E$

$P \Rightarrow$ $E \rightarrow E+E$

$E \rightarrow E * E$

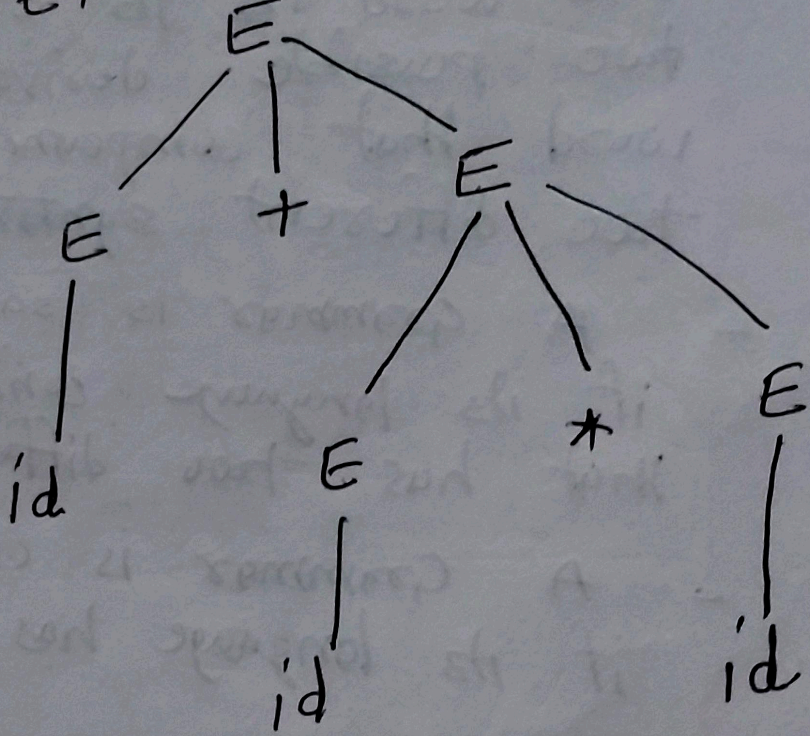
$E \rightarrow id$

By Using above 3 productions we can construct

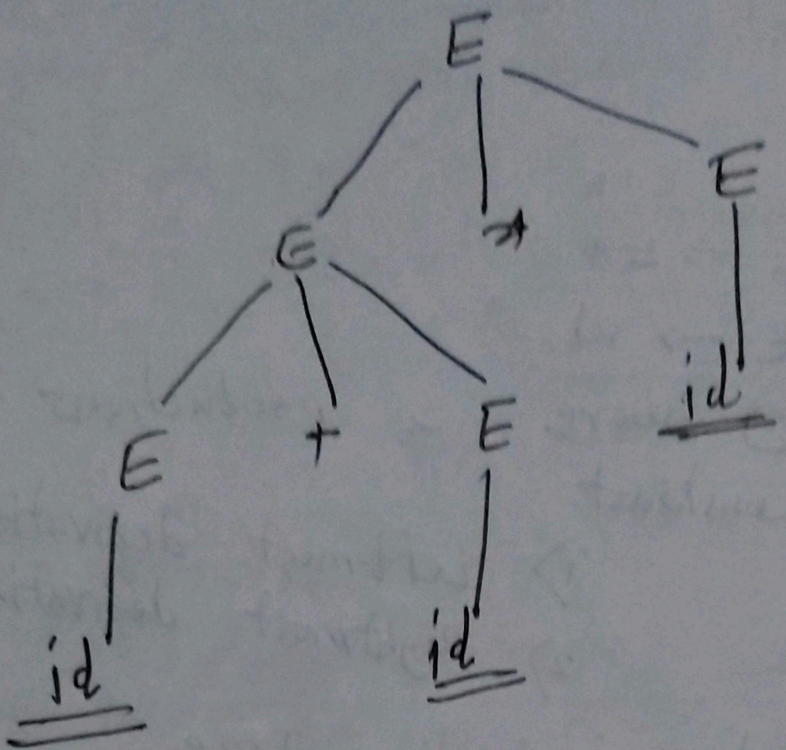
1) Leftmost derivation tree
2) Rightmost derivation tree

① Leftmost Derivation Tree

$E \rightarrow E+E$



Right Most derivation



- A CFG is ambiguous if for atleast one word in its CFL there are two possible derivations of the word that compoun correspond to two different syntax or parse tree
- A grammar is said to be ambiguous if its language contains some string that has two different parse trees
- A Grammar is called unambiguous if its language has exactly one parse tree

* **Simplification of CFG**
  1) Eliminating Useless symbol
  2) Eliminating unit productions.
  3) Eliminate $\wedge$ productions / ϵ

① **Useful & Useless Symbols**

Useful	Useless
ⓐ Derives a string of terminals	ⓐ Does not derive a string of terminals
ⓑ It is used in derivation of atleast one w in L(CFG)	ⓑ It does not occour in derivation sequence of any w in L(CFG)

Eg:- Find useful & Useless symbol in $G = (V, \Sigma, S, P)$

$V = \{S, A, B\}$ → NT symbols

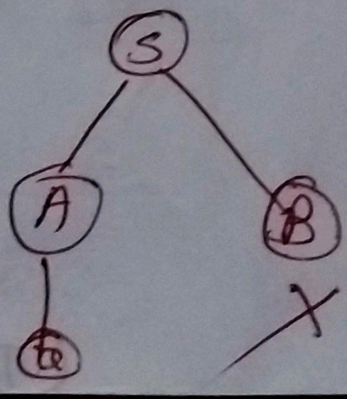
$T/\Sigma = \{a\}$ → T Symbols

$P \Rightarrow \{ S \to a / AB$ & $A \to a \}$

→ The starting symbol S can be terminate at 1) $S \to a$

2) If we take 2) $S \to AB$

Hence A & B are not useful are useless symbols



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

$V = \{S, A, B, C\}$; S = starting state

$T/\Sigma = \{a, b\}$

$P: \Rightarrow \quad S \rightarrow aS \mid A \mid C$

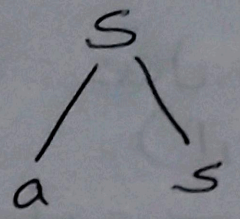
$A \rightarrow a$

$B \rightarrow aa$

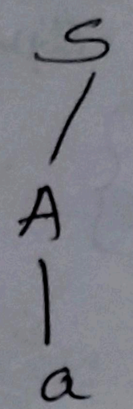
$C \rightarrow aCb$

→ As S is starting symbol. The starting symbol is always useful.

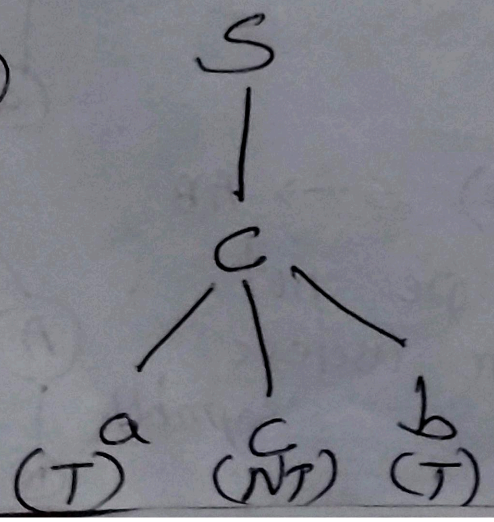
Now check reachability from start state

① 

$S \rightarrow aS$ is useful

② 

$A \rightarrow a$ is useful

③ 

Not useful

$S \rightarrow C$

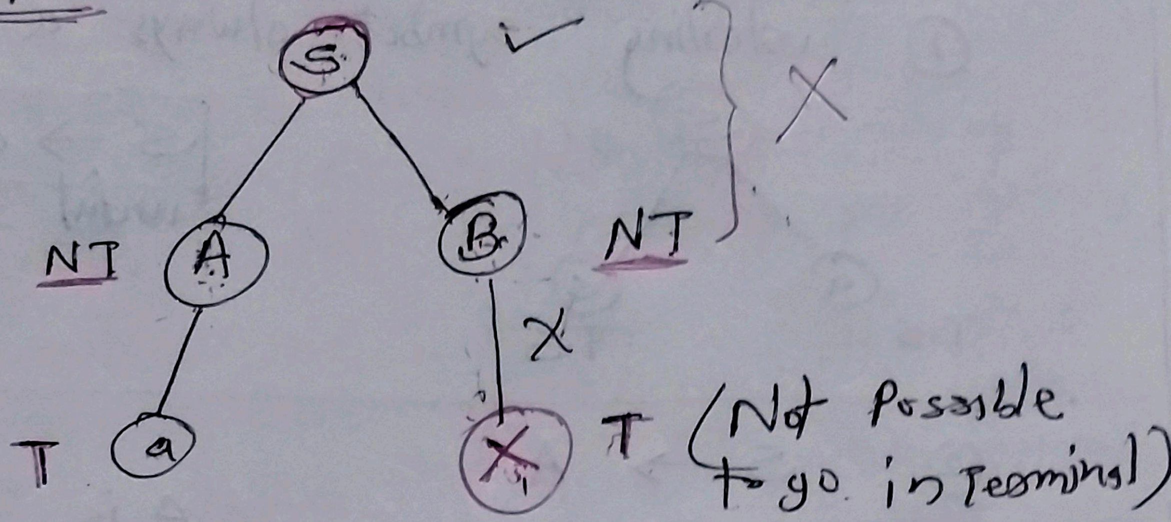
$C \rightarrow aCb$ &

$B \rightarrow aa$

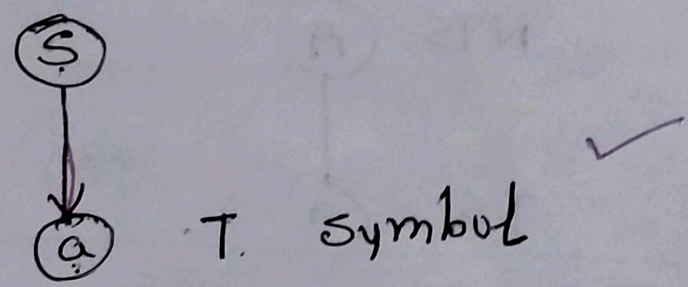
$S \rightarrow a \mid AB$
$A \rightarrow a$

$$\boxed{\begin{array}{l} S \rightarrow a \\ S \rightarrow AB \\ A \rightarrow a \end{array}}$$

The starting symbol is always useful

① 

② $S \rightarrow a$



from Above trees
① Useful symbol → S
② Useless symbol → A & B

Final Production will

$$\boxed{S \rightarrow a}$$

$P \Rightarrow \{ S \to aS$ ✓
$S \to A$ ✓
$S \to C$ ✗
$A \to a$ ✓
$B \to aa$ ✗
$C \to aCb \}$ ✗

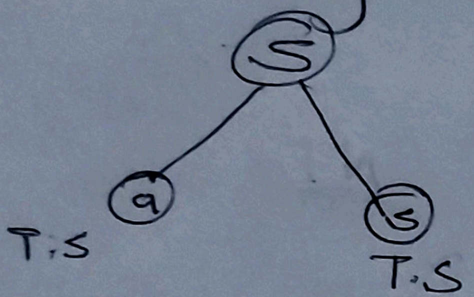
$S \Rightarrow$ starting symbol

$T / \Sigma = \{a, b\}$

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

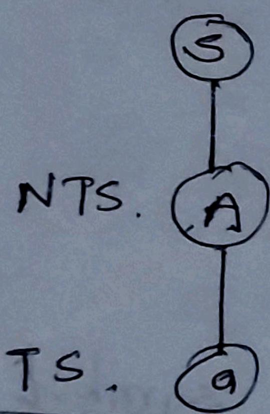
$\Rightarrow$ **Useful & Useless Symbol**

① starting symbol always a useful.



$\boxed{S \to aS}$ is useful production

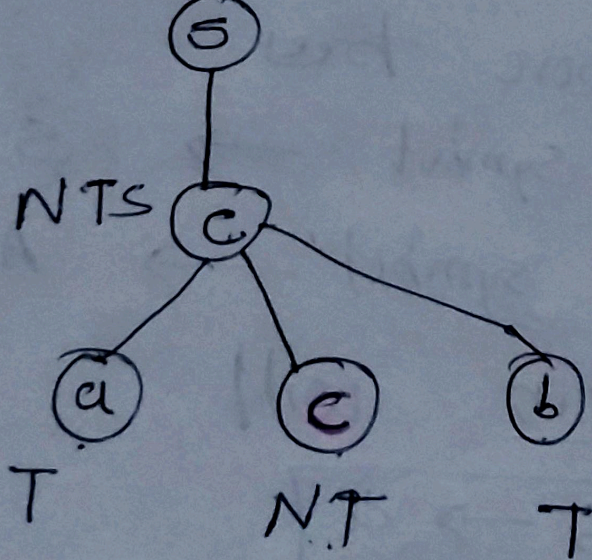
② $S \to A$



A is useful &

$\boxed{\begin{array}{l} S \to A \\ A \to a \end{array}}$ are useful

③ $S \to C$



$\boxed{\begin{array}{l} S \to C \\ C \to aCb \end{array}}$ are useless

$B \to aa$ is useless production

* $G = (V, \Sigma, S, P)$ $V = \{S, A, B, C\}$

$T = \{a, b\}$ & $P \Rightarrow$

$S \rightarrow AB \mid CA$

$B \rightarrow BC \mid AB$

$A \rightarrow a$

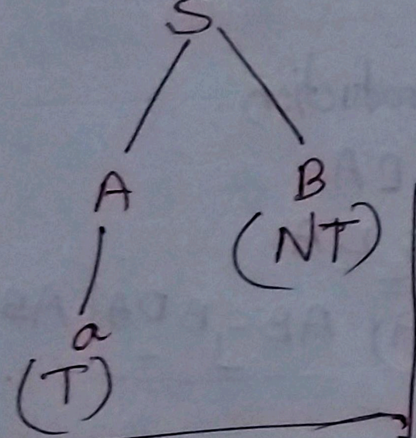
$C \rightarrow aB \mid b$

Ans

S - starting state

→ As S is starting state it must be consider as useful variable

① Now Take 1st Production out of 7 productions / Also 5th production we got

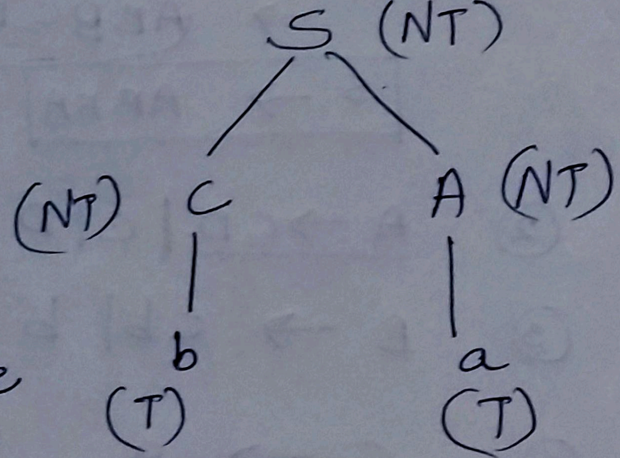


Hence

$S \rightarrow AB$ (1st)

$B \rightarrow AB$ (4th) are useless

2) Use 2nd, 5th & 7th Production we get



from this tree we can say that.

2nd, 5th & 7th productions are useful

3) We can say that the productions 3rd & 6th are useless hence we can conclude that.

$\Rightarrow$ $S - CA$, $A \rightarrow a$ & $C - b$ are useful

* Elimination of Λ Productions

The Production of the form $A \to \Lambda$ is called null production. It may also represent like

$$A \to \Lambda$$
$$A \to \epsilon$$

$S \to ABCBCDA$ ✓

$A \to CD$

$B \to Cb$

$C \to a \mid \Lambda$ $\Rightarrow$ $C \to \Lambda$

$D \to bD \mid \Lambda$ $\quad$ $D \to \Lambda$

Step 1 :- Take first Production

$S \to ABCBCDA$

$S \to ABC_1BC_2DA$

$\to ABBC_2DA \mid ABC_1BDA \mid ABC_1BC_2A$

$\boxed{S \to ABBA}$ ✓

(2) $A \to CD \mid C \mid D$

(3) $B \to Cb \mid b$

(4) $C \to a$

(5) $D \to bD \mid b$

$S \to ABCBCDA$
$A \to CD$
$B \to Cb$
$C \to a \mid \Lambda$
$D \to bD \mid \Lambda$

$\Rightarrow$ $C \to \Lambda$ & $D \to \Lambda$ are null productions

① $S \to ABCBCDA$
$\to AB\Lambda B\Lambda\Lambda A$
$\boxed{S \to ABBA}$

② $A \to CD$
$A \to \Lambda D \mid C\Lambda$
$\boxed{A \to CD \mid D \mid C}$

③ $B \to Cb$
$B \to \Lambda b \to b$
$\boxed{B \to Cb \mid b}$

④ $C \to a \mid \Lambda$
$\boxed{C \to a}$

⑤ $D \to bD \mid \Lambda$
$D \to b\Lambda \longrightarrow b$
$\boxed{D \to bD \mid b}$

Finding Nullable Variables

$S \to ABCBCDA$

$A \to CD$

$B \to Cb$

$C \to a \mid \Lambda$

$D \to bD \mid \Lambda$

① $G = (V, \Sigma, S, P)$

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

$\Sigma = \{a, b, \Lambda\}$

S = starting symbol

② $C \to \Lambda$ & $D \to \Lambda$

So C & D are nullable variables

③ $S \to ABCBCDA$

$\to AB\Lambda B\Lambda\Lambda A$

$\to ABBA$

$\to CD\,Cb\,Cb\,CD$

$\to \Lambda\Lambda\Lambda b\Lambda b\Lambda\Lambda$

$\boxed{S \to bb}$ S is not nullable

④ $A \to CD$

$A \to \Lambda\Lambda$

$A \to \Lambda$ A is nullable

⑤ $B \rightarrow Cb$

$B \rightarrow \Lambda b$

$\boxed{B \rightarrow b}$

B is not nullable

Finally

Nullable Variables A, C, D

Not Nullable — B, S

Example 2

$S \rightarrow ACB \mid cbB \mid Ba$

$A \rightarrow da \mid BC$

$B \rightarrow gC \mid \Lambda$

$C \rightarrow ha \mid \epsilon$

① Remove Null productions

② Find Nullable Variables.

$\Rightarrow$ Null Productions are $B \rightarrow \Lambda$

$C \rightarrow \Lambda$

$S \rightarrow ACB \mid cbB \mid Ba$

$$\boxed{\begin{array}{l} S \rightarrow ACB \mid AC \mid AB \mid A \\ S \rightarrow cbB \mid cb \\ S \rightarrow Ba \mid a \end{array}}$$

(2) $A \to da \mid BC$

$$\boxed{\begin{aligned} &A \to da \\ &A \to BC \mid B \mid C \end{aligned}}$$

(3) $B \to gC \mid \Lambda$

$$\boxed{B \to gC \mid g}$$

(4) $\boxed{C \to ha}$

Nullable Variables

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

$B \to \Lambda$ & $C \to \Lambda$

B & C are nullable

$S \to ACB \mid cbB \mid Ba$

$S \to A\Lambda\Lambda \mid cb\Lambda \mid \Lambda a$

$\boxed{S \to A \mid cb \mid a}$ S is not nullable

$\boxed{C \to ha}$ $\in$

C is ~~not~~ nullable.

$A \rightarrow da \mid BC$

$A \rightarrow da \mid \wedge\wedge$

$\boxed{A \rightarrow da}$ A is not nullable

Nullable are B & C

Not Nullable are A & S

re

Eg: Remove Null / Epsilon Product

$$S \rightarrow AB$$
$$A \rightarrow aA \mid a$$
$$B \rightarrow bB \mid \epsilon.$$

$\Rightarrow$

$$\boxed{\begin{array}{l} S \rightarrow AB \mid A \\ A \rightarrow aA \mid a \\ B \rightarrow bB \mid b \end{array}}$$

step I :- Now we have epsilon symbol.

$$B \rightarrow \epsilon$$

Initially

$$S \rightarrow AB$$
$$A \rightarrow aA \mid a$$
$$B \rightarrow bB$$

step II :- Now replace $B \rightarrow \epsilon$ whereever applicable at right side

$$S \rightarrow A\epsilon$$
$$A \rightarrow aA \mid a$$
$$B \rightarrow b\epsilon$$

ie

$$S \rightarrow A$$
$$A \rightarrow aA \mid a$$
$$B \rightarrow b$$

Hence final result will be

$$S \rightarrow AB \mid A$$
$$A \rightarrow aA \mid a$$
$$B \rightarrow bB \mid b$$

* Removing Unit Productions

If $G = (V, \Sigma, S, P)$ is CFG & we have a productions such a way that:

$$\boxed{V_1 \rightarrow V_2}$$

$\forall$ V_1 & V_2 are NT symbols then $V_1 \rightarrow V_2$ called unit production

$$\boxed{V_1, V_2 \in V}$$

eg:-

a) $S \rightarrow A$ — Unit P.

b) $S \rightarrow a$ — Not Unit P.

c) $S \rightarrow \wedge / \epsilon$ — Null Production

d) $S \rightarrow Aa$ — Not UNIT P

(1)

$$\left.\begin{aligned} S &\rightarrow AB \\ A &\rightarrow a \\ B &\rightarrow C \mid b \\ C &\rightarrow D \\ D &\rightarrow E \\ E &\rightarrow a \end{aligned}\right\} \text{set of productions}$$

i) Identify Unit & non unit productions.

Unit Production

1 NT symbol $\rightarrow$ 1 N.T Symbol

Unit production:-

$B \rightarrow C$

$C \rightarrow D$

$D \rightarrow E$

Non Unit production

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b$

$E \rightarrow a$

Eg:-

$S \rightarrow ABC \mid \Lambda$

$B \rightarrow a \mid b$

$C \rightarrow D$

$E \rightarrow aB$

$F \rightarrow AB$

Unit Production

$C \rightarrow D$

Null production

$S \rightarrow \Lambda$

Non Unit Production

$S \rightarrow ABC$

$B \rightarrow a \mid b$

$E \rightarrow aB$

$F \rightarrow AB$

★☆ Removing Unit Productions

Production of the form

$A \rightarrow B$, $B \rightarrow S$.

are called unit production

eg.

$S \rightarrow A | bb$ — (I)

$A \rightarrow B | b$ — (II)

$B \rightarrow S | a$ — (III)

from (I) & (III)

$B \rightarrow bb | a$ — (IV)

from (II) & (IV)

$A \rightarrow bb | a | b$ — (V)

from (I) & (II) & (V)

$S \rightarrow b | bb$

The final Result will be

$$\boxed{\begin{array}{l} S \rightarrow bb \\ A \rightarrow b \\ B \rightarrow a \end{array}}$$

Eliminating useless variables from a Context free Grammars

A Context free Grammar is $\{V, \Sigma, S, P\}$ contains some

$V =$ Nonterminal

$\Sigma = T =$ Terminal symbol.

If the given Nonterminal symbol is not in use called as useless symbol or variable.

Now consider following grammer

$G = \{V, \Sigma, S, P\}$

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

$\Sigma = \{a, b\}$

$S \rightarrow AA \mid 1A0 \mid B \mid 0A1 \mid 01$ —— (I)

A & B are Nonterminals & 0 & 1 are terminals

$S \rightarrow AA \mid 1A0 \mid 0A1 \mid 01$ —— (II)

B is useless symbol

both (I) & (II) are identical

Remove those strings whose string contain at least one non used string

Eg:-

$G = \{V, T, S, P\}$

$V = \{S, C, D\}$

$T = \{d\}$

$P \Rightarrow S \rightarrow CD \mid d$

$D \rightarrow d$

from above production we can say C is useless so we can remove C

$S \rightarrow d$

$D - d.$

$G = \{\{S\}, \{d\}, \{S\}\{P\}\}.$

Now we observe $S \rightarrow d$ doesn't contain any Nonterminal symbol D in it so it become

$\boxed{S \rightarrow d}$ only by removing $D \rightarrow d$

Example

$$\boxed{S \to a \mid aA \mid B \mid C}$$

$A \to aB \mid \epsilon$

$B \to Aa$

$C \to cCD$

$D \to fff$

1) Unit Production
2) Useful Variables
3) Useless Variable
4) $\wedge$ Production
5) Nullable Variables

① Unit Production

$G = (V, \Sigma, S, P)$ &

we have production such a way that

$$\boxed{1\ NT \to 1\ NT}$$

$$\boxed{NT \in V}$$

eg :- $A \to B$ } Unit

$S \to D$ etc } Production

$A \to a$ } Not Unit

$B \to b$ } Production

Unit Production :- $S \to B \mid C$

Non Unit Production :- $S \to a \mid aA$

$A \to aB$

$C \to cCD$

$B \to Aa$

$D \to fff$

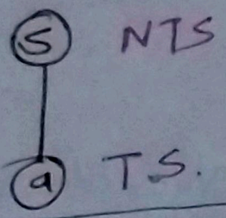
Useless & Usefull Variable

$S \rightarrow a \mid aA \mid B \mid C$ ✓
$A \rightarrow aB \mid \epsilon$ ✓
$B \rightarrow Aa$ ✓
$C \rightarrow cCD$
$D \rightarrow fff$

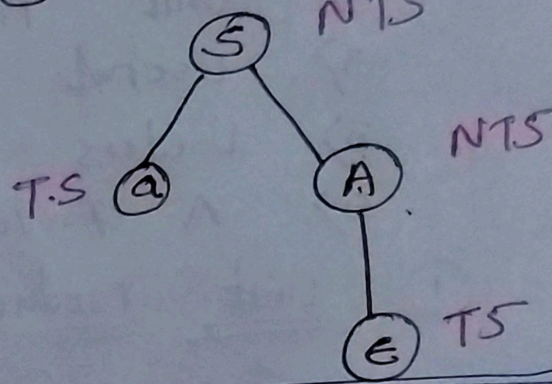
$\boxed{\text{Useful :- S, A, B}}$

$\boxed{\text{Useless :- C, D}}$

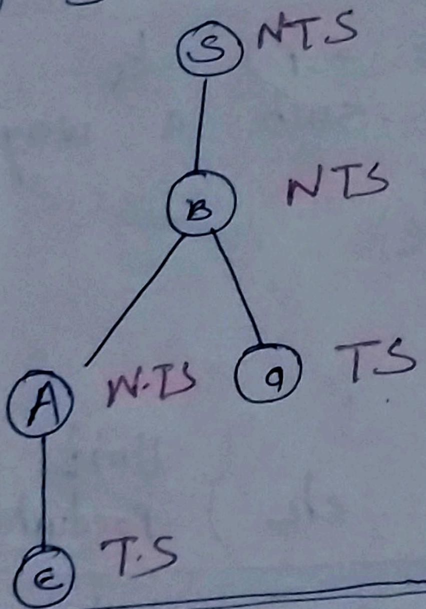
① $S \rightarrow a$ ✓



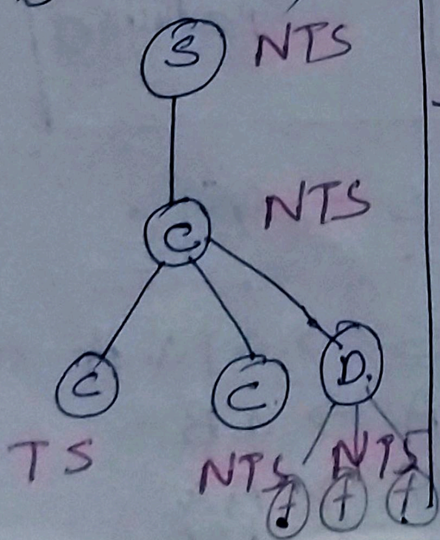
② $S \rightarrow aA$ ✓



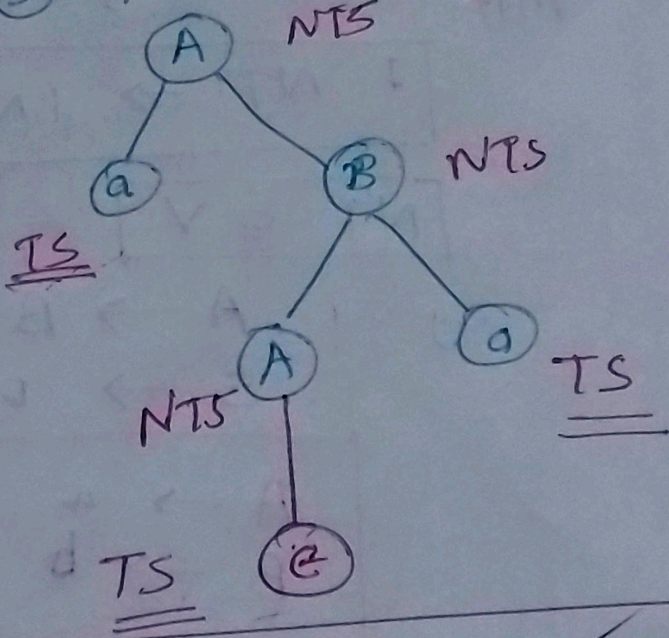
③ $S \rightarrow B$ ✓



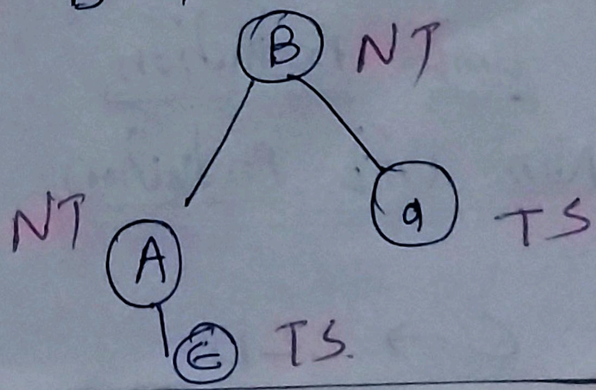
④ $S \rightarrow C$ ✗



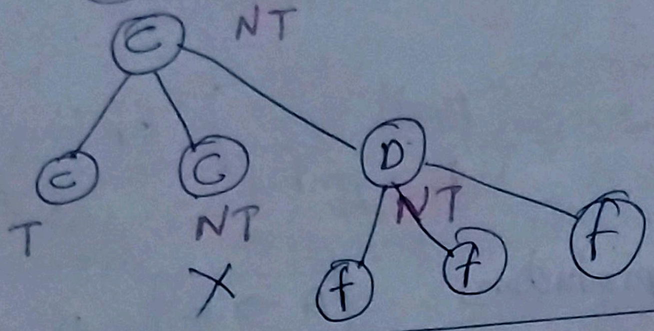
⑤ $A \rightarrow aB$ ✓



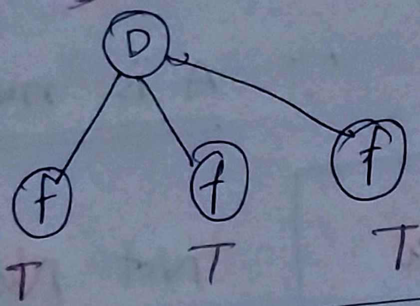
⑥ $B \rightarrow Aa$ ✓



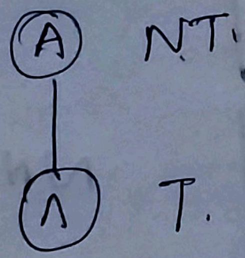
7 $C \rightarrow cCD$ ✗



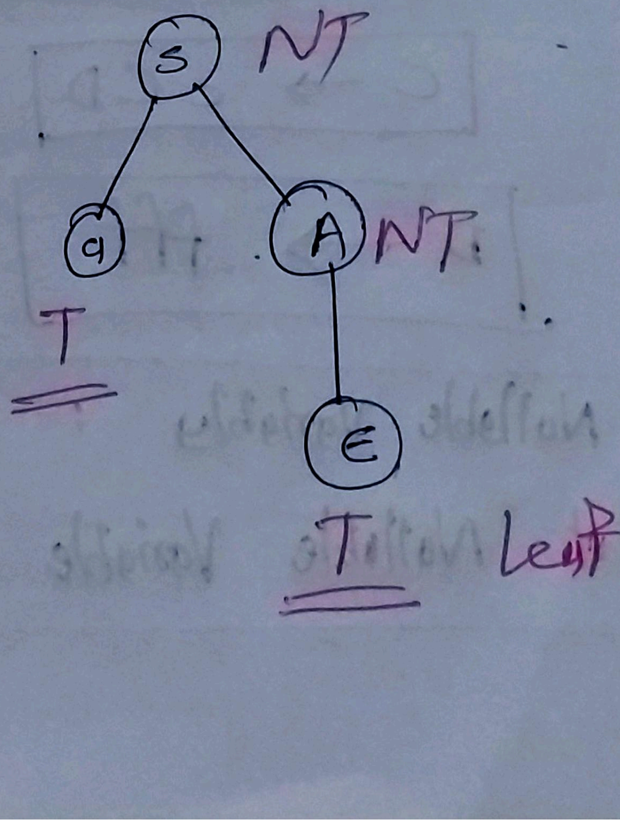
8 $D \rightarrow fff$ ✗



9 $A \rightarrow \epsilon / \Lambda$ ✓ Epsilon $S \rightarrow a A$



finally



☆ Null Production

$A \rightarrow \epsilon$

N.T. symbol

Null symbol / Epsilon symbol

☆ In given productions $\boxed{A \rightarrow \epsilon}$ is null production

☆ Nullable Variables

$A \rightarrow \epsilon$ ∴ A is nullable

$$\boxed{\begin{array}{l} S \rightarrow a \\ S \rightarrow aA \mid aA \mid a \\ S \rightarrow B \mid Aa \mid Aa \mid a \\ S \rightarrow c \end{array}}$$

Not Nullable productions

$\boxed{B \rightarrow Aa \mid \epsilon a \mid a}$ NNP

$\boxed{C \rightarrow cCD}$ NNP

$\boxed{D \rightarrow fff}$ NNP

Nullable Variables :- A

Not Nullable Variable :- S, B, C, D

$Y_1 \rightarrow c_2$

3.2 BNF & CNF Notations
[Boyce Code Normal form
Chomskey Normal form]

1) <u>Chomskey Normal form</u>

In a CFG the production is of the form $A \to BC$
$A \to a$

& $A \to 2NT$
$S \to IS$

ie
Nonterminal/variables $\to$ string of two non terminals
Nonterminal $\to$ Terminal symbol.
then they called. Chomskey Normal form

Eg:-
If two productions $A \to aAb$ &
$B \to ab$ &

replace
$X_a \to a$, $X_b \to b$

$A \to X_a A X_b$
$B \to X_a X_b$

Also
$A \to BCDBCE$
would be replace by

$A \to BY_1$
$Y_1 \to CY_2$

2) Chomsky Normal form (CNF)

$Y_2 \rightarrow DY_3$
$Y_3 \rightarrow BY_4$
$Y_4 \rightarrow CE$

Hence New variables Y_1, Y_2, Y_3, Y_4 are productions so it is CNF.

- If CFG $G = (V, \Sigma, S, P)$ be in CNF & $G' = (V', \Sigma, S, P')$ then

$$\boxed{L(G') = L(G) - \{\Lambda\}}$$

2) Chomsky Normal form (CNF)

If The given CFG only of the form $G = (V, \Sigma, S, P)$

Nonterminal → string of exactly two Nonterminals

1 NT → 2 NT

Nonterminal → one terminal called as chomsky Normal form

1 NT → 1 TS

eg

$A \to ab$ ✗

$B \to Ab$ ✗

$A \to A$ ✗

If they CFL contain ϵ symbol then convert into CFG & then to CNF with removal of ϵ in CNF

Eg :-

$G = \{V, T, S, P\}$

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

$T = \{a, b\}$

$P \Rightarrow \quad S \to bA / aB$

$A \to bAA / aS / a$

$B \to aBB / bS / a$

Now take $b = C_b$ ∀ C_i = Non terminal Symbol

$a = C_a$

$P \Rightarrow \quad S \to C_b / C_a B$

$A \to C_b AA / C_a S / a$

$B \rightarrow C_a BB / C_b S / a$

Now. first rule / stril is in CNF

Now

$A \rightarrow C_b AA / C_a S / a$

Now $AA \rightarrow D$
$BB \rightarrow E$

$A \rightarrow C_b D / C_a S / a$

$B \rightarrow C_a E / C_b S / a$

Hence required CNF

Eg:-

1) $S \rightarrow aSa \mid bSb \mid a \mid b \mid aa \mid bb$.

2) $S \rightarrow 1A / 0B$

$A \rightarrow 1AA / 0S / 0$

$B \rightarrow 0BB / 1$.

Chomsky Normal form (CNF)

$$\boxed{\begin{array}{l} 1\ NT \to 2\ NT \\ 1\ NT \to 1\ TS \end{array}}$$

$S \to bA \mid aB$

$A \to bAA \mid aS \mid a$

$B \to aBB \mid bS \mid a$

① $S \to bA$

Take $\boxed{b = S_1}$ in eqn ①

∀ b = Terminal symbol

S_1 = Non Terminal Symbol.

$\boxed{S \to S_1 A.}$ CNF

1 NT

2 NT

② $S \to aB$

put $a = S_2$

$\boxed{S \to S_2 B}$ CNF

③ $A \to bAA$

$A \to S_1 AA$

FACULTY ... Tal - Hatkanangale ... (11)

$A \rightarrow S_1 A A$

put $S_3 \rightarrow S_1 A$ in (17)

$\boxed{A \rightarrow S_3 A}$ CNF

(4) $A \rightarrow aS$

$\boxed{A \rightarrow S_2 S}$ CNF

(5) $A \rightarrow a$ Already CNF

(6) $B \rightarrow aBB$

$B \rightarrow S_2 BB$

$\boxed{B \rightarrow SB}$ CNF

(7) $B \rightarrow bS$

$\boxed{B \rightarrow S_1 S}$ CNF

(8) $\boxed{B \rightarrow a}$ Already CNF.

Required CNF

$V = \{S, A, B, S_1, S_2, S_3\}$

$T/\Sigma = \{a, b\}$

$$P = \left\{ \begin{array}{l} S \rightarrow S_1 A \mid S_2 B \\ A \rightarrow S_3 A \mid S_2 S \mid a \\ B \rightarrow SB \mid S_1 S \mid a \end{array} \right\}$$

S = starting Symbol

D. Y. PATIL TECHNICAL CAMPUS
FACULTY OF ENGINEERING & MANAGEMENT
Talsande, Tal - Hatkanangale, Dist - Kolhapur.

201 - 201					Semester - I / II					
...gramme					Paper No. :					
... No. :					Date :					
...ject :					Sign of Supervisor :					
...stion No	1	2	3	4	5	6	7	Total	Sign. of Examiner	
...ks Obtained										
...imum Marks										

...of Supplements : 1 =

BNF & CNF Notations :-

Convert given CFG into BNF & CNF

1) **BNF :-** Bacos Nayr Form (BNF)
CFG Notation with minor changes in original form.

Bacous Normal form (BNF)

→ It is same as CFG.
→ It can add some convienent symbol else
1) Kleene *
2) Union |
3) ~~Metalanguage~~ / Metalevel paranthesis
( )

It is metalanguage for CFL
Essential in Compiler Construction

eg digit = 0|1|2|3|4|5|6|7|8|9

Backus Normal Form

The compact Notation, used to represent production rule called BNF

eg. ① $\langle rest \rangle \rightarrow \langle letter \rangle \langle rest \rangle$

$\langle rest \rangle \rightarrow \langle digits \rangle \langle rest \rangle$

$\langle rest \rangle \rightarrow \epsilon$

by using BNF it will be

$\langle rest \rangle \rightarrow \langle letter \rangle \langle rest \rangle \mid \langle digit \rangle \langle rest \rangle \mid \epsilon$

(ii) $S \rightarrow SS$

$S \rightarrow a$

$S \rightarrow \epsilon$

The BNF will be

$S \rightarrow SS \mid a \mid \epsilon$

☆ Converting CFG to CNF

Let ⇒ G be the grammer of the productions

$S \rightarrow AACD$
$A \rightarrow aAb \mid \Lambda$
$C \rightarrow aC \mid a$
$D \rightarrow aDa \mid bDb \mid \Lambda$

Convert into CNF.

1) **Eliminating Λ Productions** in above example the nullable variables are A & D.

$S \rightarrow AACD \mid ACD \mid AAC \mid CD \mid AC \mid C$

$A \rightarrow aAb \mid ab$

$C \rightarrow aC \mid a$

$D \rightarrow aDa \mid bDb \mid aa \mid bb$

2) **Eliminating Unit Productions**

Now from Production i) & iii) it is

$S \rightarrow aC \mid a$

∀ we delete $S \rightarrow C$

3) Restricting the right side of production to single terminals or strings of two or more variables

$S \to AACD \mid ACD \mid AAC \mid CD \mid AC \mid X_aC \mid a$

$A \to X_aAX_b \mid X_aX_b$

$C \to X_aC \mid a$

$D \to X_aDX_a \mid X_bDX_b \mid X_aX_a \mid X_bX_b$

$X_a \to a$

$X_b \to b$

4) Final step to CNF

$S \to AT_1 \qquad T_1 \to AT_2 \qquad T_2 \to CD$

$S \to AU_1 \qquad U_1 \to CD$

$S \to AV_1 \qquad V_1 \to AC$

$S \to CD \mid AC \mid X_aC \mid a$

$A \to X_aW_1 \qquad W_1 \to AX_b$

$A \to X_aX_b$

$C \to X_aC \mid a$

$D \to X_aY_a \qquad Y_1 \to DX_a$

$D \to X_bZ_1 \qquad Z_1 \to DX_b$

$D \to X_aX_b \mid X_bX_b$

$X_a \to a$

$X_b \to b$

* Convert CFG into CNF

1) $S \rightarrow ASA$

$S \rightarrow BSB$

$S \rightarrow AA$

$S \rightarrow BB$

$S \rightarrow a$

$S \rightarrow b$

$A \rightarrow a$

$B \rightarrow b$

Soln - Expect 1st & 2nd Production all productions are in CNF

i) $S \rightarrow AX_1$

$X_1 \rightarrow SA$

ii) $S \rightarrow BX_2$

$X_2 \rightarrow SB$

Hence required CNF is

$S \rightarrow AX_1$

$S \rightarrow BX_2$

$S \rightarrow AA$

$S \rightarrow BB$

$S \rightarrow a$

$S \rightarrow b$

$A \rightarrow a$

$B \rightarrow b$

eg.

(a)

$$S \rightarrow ABA$$
$$A \rightarrow aA \mid \epsilon$$
$$B \rightarrow bB \mid \epsilon$$

(b)

$$S \rightarrow ABC$$
$$A \rightarrow a \mid b$$
$$B \rightarrow Bb \mid bb$$
$$C \rightarrow aC \mid cc \mid ba$$

1) Steps for Conversion of CFG to CNF

(a) Eliminate ϵ productions

(b) Eliminate unit productions

(c) Eliminate terminal on RHS.

(d) Restrict the no of variables on RHS.

Eg: $S \to ABC$
$A \to a \mid b$
$B \to Bb \mid bb$
$C \to aC \mid cc \mid ba$

$G = (V, \Sigma, S, P)$
$V = \{S, A, B, C\}$
$\Sigma = \{a, b\}$
S = starting symbol

$\Rightarrow$ CNF   1 NT $\to$ 2 NT
1 NT $\to$ 1 TS

① Take $S \to ABC$ — ①
put $S_1 \to AB$ in ①

$\boxed{S \to S_1 C}$ CNF

② $\boxed{A \to a \mid b}$ CNF ✓

③ $B \to Bb \mid bb$
take $A \to a \mid b$

$\boxed{B \to BA \mid AA}$ CNF

$C \to aC \mid CC \mid ba$ — (II)

where $A \to a|b$ in (II)

$$\boxed{C \to AC \mid CC \mid AA}$$ CNF

Non Terminal Symbols

$V = \{S, S_1, A, B, C\}$