

(1)

Module - 3

Context-Free Grammars (CFG)

A context-free grammar (or CFG) to be a grammar in which each rule must:

- * have a left-hand side that is a single nonterminal &
- * have a right-hand side that is ϵ or terminal; nonterminal one or more.

A context grammar G is 4 tuple (or) Quad tuple

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

where

V is set of variables/non terminals

T is set of terminals

P is set of production

Each production is of the form $\alpha \rightarrow \beta$

where α is a string from variable & α cannot be ϵ , but β is string from $(VUT)^*$. Hence it can include ϵ also.

(Eg)

Obtain grammar to generate string consisting of any no of a's



$$\delta(s, a) = s$$

$$\downarrow \quad \downarrow$$

$$s \rightarrow a s$$

$$S \rightarrow a s / \epsilon$$

$$S \rightarrow \epsilon$$

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

$$V = \{s\}$$

$$T = \{a\}$$

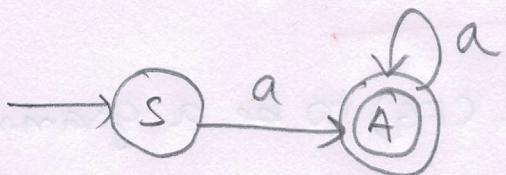
$$P = \{s \rightarrow a s / \epsilon\}$$

s is start symbol

Eg

Obtain a grammar to generate string consisting of at least 1a.

Sol?



$$\delta(S, a) = A$$

$$S \rightarrow RA$$

$$\delta(A, a) = A$$

$$A \rightarrow aA$$

$$A \rightarrow \epsilon$$

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

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

$$T = \{a\}$$

$$P = \{S \rightarrow aA\}$$

$$A \rightarrow aA / \epsilon$$

S is Start symbol.

Eg

Obtain a grammar to generate string consisting of even no of a's

Sol?



$$\delta(S, a) = A$$

$$S \rightarrow aA$$

$$S \rightarrow \epsilon$$

$$\delta(A, a) = S$$

$$A \rightarrow aS$$

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

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

$$T = \{a\}$$

$$P = \{S \rightarrow aA / \epsilon\}$$

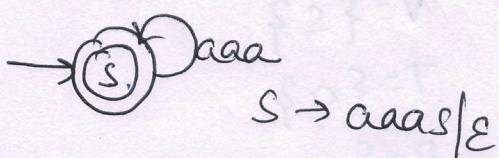
$$A \rightarrow aS$$

S is Start symbol.

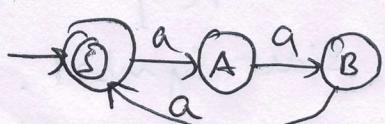
Eg

Obtain a grammar to generate string consisting of multiples of 3 a's.

Sol?



or



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

$$V = \{S\}$$

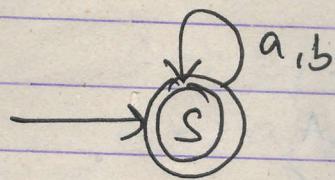
$$T = \{a\}$$

$$P = \{S \rightarrow aaaas / \epsilon\}$$

S is Start symbol.

③ obtain grammar to generate string consisting of any number of a's & b's

Soln



$$S \rightarrow E$$

$$S \rightarrow aS$$

$$S \rightarrow bS$$

$$\text{so } S \rightarrow \epsilon/aS/bS$$

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

$$V = \{ S \}$$

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

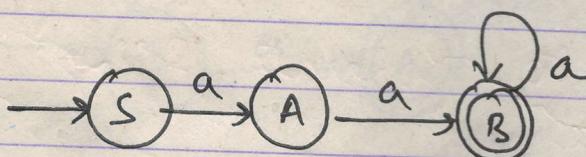
$$P = \{$$

$$S \rightarrow \epsilon/aS/bS \}$$

S is start symbol

④ obtain grammar to generate string consisting of at least two a's

Soln



$$S \rightarrow aA$$

$$A \rightarrow aB$$

$$B \rightarrow \epsilon$$

$$B \rightarrow aB$$

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

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

$$T = \{ a \}$$

$$P = \{$$

$$S \rightarrow aA$$

$$A \rightarrow aB$$

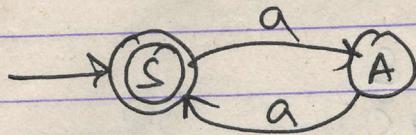
$$B \rightarrow \epsilon/aB \}$$

~~B $\rightarrow aB$~~

8. is the start symbol

(eg) obtain grammar to generate string consisting of even number of a's.

Sol'



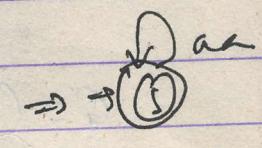
$$S \rightarrow \epsilon$$

$$S \rightarrow aA$$

$$A \rightarrow aS$$

or

$$S \rightarrow \epsilon | aas$$



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

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

$$T = \{ a \}$$

$$P = \{ S \rightarrow \epsilon | aA \}$$

$$A \rightarrow aS \}$$

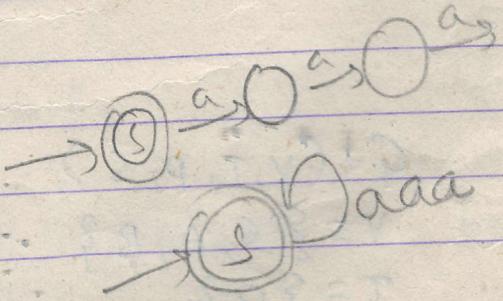
where S is start symbol.

(eg)

obtain grammar to generate string consisting of multiples of three a's.

Sol'

$$S \rightarrow \epsilon | aaas$$



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

$$V = \{ S \}$$

$$T = \{ a \}$$

$$P = \{ S \rightarrow \epsilon | aaas \}$$

where S is start symbol.

(eg)

Obtain grammar to generate strings of a's & b's such that string length is multiple of 3

Soln

$$S \rightarrow \epsilon / A A A S$$

$$A \rightarrow a / b$$

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

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

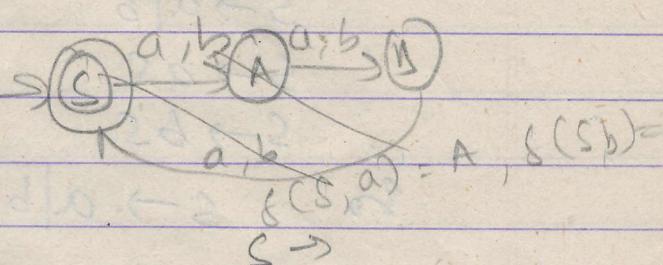
$$T = \{a, b\}$$

$$P = \{$$

$$S \rightarrow \epsilon / A A A S$$

$$A \rightarrow a / b\}$$

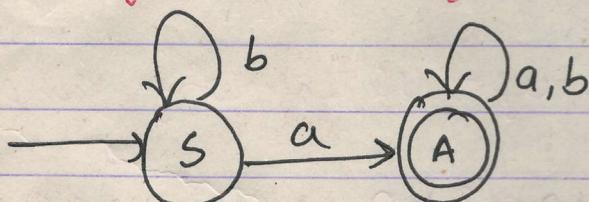
S is the start symbol.



(eg)

Obtain grammar to generate string consisting of any number of a's & b's with at least one a.

Soln



$$S \rightarrow bS$$

$$S \rightarrow aA$$

$$A \rightarrow \epsilon / aA / bA$$

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

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

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

$$P = \{ S \rightarrow aA / bS$$

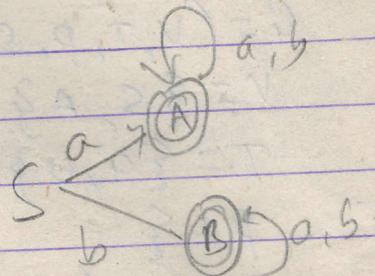
$$A \rightarrow aA / bA / \epsilon \}$$

S is the start symbol.

(eg) obtain grammar to generate string consisting of any no of a's & b's with at least one a or at least one b.

Sol:

$$\begin{aligned} S &\rightarrow a \mid b \\ S &\rightarrow aS \\ S &\rightarrow bS \\ \text{so } S &\rightarrow a \mid b \mid aS \mid bS \end{aligned}$$



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

$$V = \{ S \}$$

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

$$P = \{ S \rightarrow a \mid b \mid aS \mid bS \}.$$

S is the start symbol.

$$S \rightarrow aA \mid bB$$

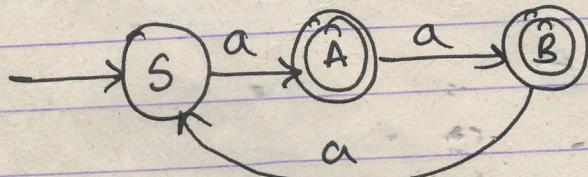
$$\begin{aligned} S &\rightarrow aA \\ A &\rightarrow aa \mid bA \mid \epsilon \end{aligned}$$

$$\begin{aligned} S &\rightarrow bB \\ B &\rightarrow ab \mid bB \mid \epsilon \end{aligned}$$

(eg) obtain grammar to accept the following language

$$L = \{ w : |w| \bmod 3 > 0 \text{ where } w \in \{a\}^* \}$$

Sol:



$$S \rightarrow a \mid aa \mid aaaS$$

$$S \rightarrow aA$$

$$A \rightarrow ab \mid \epsilon$$

$$B \rightarrow aS \mid \epsilon$$

$$1 \bmod 3 = 1$$

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

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

$$T = \{ a \}$$

$$P = \{ S \rightarrow aA$$

$$A \rightarrow aB | \epsilon$$

$$B \rightarrow aS | \epsilon \}$$

S is the start symbol.

Grammars from Regular Expressions

(e.g.)

obtain grammar to generate strings of a's & b's having a substring ab.

Soln: The RE representing strings of a's & b's having a substring is given by

$$(a+b)^* ab (a+b)^*$$

↓ ↓ ↓
 A ab A

$$\begin{aligned} S &\rightarrow AabA \\ A &\rightarrow \epsilon | aA | bA \end{aligned}$$

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

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

$$T = \{ab\}$$

$$P = \{$$

$$S \rightarrow AabA$$

$$A \rightarrow \epsilon | aA | bA\}$$

S is start symbol.

(e.g.)

obtain grammar to generate string of a's ending with string ab.

Soln

$$(a+b)^* ab$$

↓ ↓,
 A ab

$$\begin{aligned} S &\rightarrow Aab \\ A &\rightarrow \epsilon | aA | bA \end{aligned}$$

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

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

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

$$P = \{$$

$$S \Rightarrow Aab$$

$$A \rightarrow aA | bA | \epsilon$$

• S is the start symbol.

(e) obtain grammar to generate strings of a's & b's starting with ab.

Sol

$$ab(a+b)^*$$

$$\Downarrow \quad \Downarrow$$

$$ab \quad A$$

$$S \rightarrow abA$$

$$A \rightarrow \epsilon | aA | bA$$

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

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

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

$$P = \{ \quad S \rightarrow abA$$

$$A \rightarrow \epsilon | aA | bA \}$$

S is the start symbol.

(e) obtain the grammar to generate the following language

$$L = \{w : n_a(w) \bmod 2 = 0 \text{ where } w \in \{a, b\}^*\}$$

Sol¹

$$\begin{array}{c}
 b^* a b^* a b^* \\
 \downarrow \downarrow \downarrow \downarrow \downarrow \\
 s a s a s \\
 S \rightarrow s a s a s \\
 S \rightarrow \epsilon | b s \\
 S \rightarrow \epsilon | b s | s a s a s
 \end{array}$$

$$\begin{array}{c}
 b^* a b^* a b^* \\
 S \Rightarrow A a A a A | b A | c \\
 A \Rightarrow \epsilon | b A \\
 \cancel{s} \cancel{s}
 \end{array}$$

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

$$V = \{S\}$$

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

$$P = \{ S \rightarrow \epsilon | b s | s a s a s \}$$

s is start symbol.

Grammars for other languages

(e.g.)

Obtain a grammar to generate the following language

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

$$S \Rightarrow \epsilon$$

$$(e.g.) \quad S \Rightarrow a s b$$

$$\Rightarrow a b$$

Sol²

$$S \rightarrow \epsilon | a s b$$

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

$$V = \{S\}$$

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

$$P = \{ S \rightarrow \epsilon | a s b \}$$

s is the start symbol

$$S \Rightarrow a s b$$

$$\Rightarrow a a s b b$$

$$\Rightarrow \underline{a a a b b b}$$

(Q) obtain a grammar to generate the following language

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

Sol:

$$L = \{ab, aabb, aaabbb, \dots\}$$

$$S \rightarrow ab | asb$$

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

$$V = \{S\}$$

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

$$P = \{S \rightarrow ab | asb\}$$

S is the start symbol.

(Q) obtain a grammar to generate the following lang

$$L = \{a^{n+1} b^n : n \geq 0\}$$

Sol:

$$L = \{a^1 ab, a^2 abb, a^3 aaaabb, \dots\}$$

$$S \rightarrow a | asb$$

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

$$V = \{S\}$$

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

$$P = \{S \rightarrow a | asb\}$$

S is start symbol.

(Ex) obtain a grammar to generate the following lang

$$L = \{a^n b^{n+1} : n \geq 0\}$$

Sol:

$$S \rightarrow b \mid aSb$$

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

$$V = \{S\}$$

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

$$P = \{S \rightarrow b \mid aSb\}$$

S is start symbol.

(Ex)

obtain a grammar to generate the following lang

$$L = \{a^n b^{n+2} : n \geq 0\}$$

Sol:
Two extra b's should be generated. So, the final grammar to generate given lang is

$$S \rightarrow bb \mid aSb$$

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

$$V = \{S\}$$

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

$$P = \{S \rightarrow bb \mid aSb\}$$

S is start symbol.

(Eg) Obtain a grammar to generate the following language

$$L = \{a^n b^{2n} : n \geq 0\}$$

Sol:

$$S \rightarrow \epsilon | aSbb$$

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

$$V = \{S\}$$

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

$$P = \{S \rightarrow \epsilon | aSbb\}$$

S is start symbol.

(Eg) Let $\Sigma = \{a, b\}$ obtain a grammar G generating set of all palindromes over Σ .

Sol:

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

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

$$V = \{S\}$$

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

$$P = \{S \rightarrow \epsilon$$

$$S \rightarrow a | b$$

$$S \rightarrow aSa | bSb\}$$

S is the start symbol.

Madam

s → msms | aSa

(Eg)

Obtain a grammar to generate a lang consisting of all non-palindromes over $\{a, b\}$

Sol:

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

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

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

$$\begin{aligned} P = \{ & \\ & S \rightarrow aSa \mid bSb \\ & S \rightarrow A \\ & A \rightarrow aBb \mid bBa \\ & B \rightarrow aB \mid bB \mid \epsilon \} \\ S \text{ is start symbol.} & \end{aligned}$$

(Eg)

Obtain a grammar to generate the following language

$$L = \{WW^R \text{ where } W \in \{a, b\}^*\}$$

$$L = \{aa, bb, abba, baab, \dots\}$$

Sol:

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

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

$$V = \{S\}$$

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

$$P = \{S \rightarrow \epsilon \mid aSa \mid bSb\}$$

S is start symbol.

(eg) Obtain the grammar to generate the language

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

Solⁿ

$$L = \{ \underbrace{0^m 1^m}_S 2^n \mid m \geq 1 \text{ & } n \geq 0 \}$$

$$S \rightarrow A B$$

The variable A should produce m no of 0's & followed by n no of 1's.

$$A \rightarrow 01 \mid 0A1$$

B should produce any n of 2's

$$B \rightarrow \epsilon \mid 2B$$

$$S \rightarrow A B$$

$$A \rightarrow 01 \mid 0A1$$

$$B \rightarrow \epsilon \mid 2B$$

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

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

$$T = \{ 0, 1, 2 \}$$

$$P = \{ S \rightarrow A B, \\ A \rightarrow 01 \mid 0A1, \\ B \rightarrow \epsilon \mid 2B \}$$

S is start symbol.

(eg) Obtain the grammar to generate the lang

$$L = \{ w \mid n_a(w) = n_b(w) \}$$

sol:

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

String starting & ending with same letter

Eg: abba, babb

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

$$V = \{ S \}$$

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

$$P = \{$$

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

}

S is start symbol.

(eg) Obtain a RG grammar to generate a string balanced parentheses.

sol:

$$S \rightarrow (S) \mid \epsilon \mid SS$$

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

$$V = \{ S \}$$

$$T = \{ (,) \}$$

$$P = \{ S \rightarrow (S) \mid SS \mid \epsilon \}$$

S is start symbol.

(eg)

obtain a grammar to generate the lang:

$$L = \{a^i b^j \mid i \neq j, i \geq 0 \text{ & } j \geq 0\}$$

Soln

$$L = \{a^i b^j \mid i \neq j, i \geq 0, j \geq 0\}$$

$$S \rightarrow aSb \mid A \mid B$$

$$A \rightarrow a \mid aa$$

$$B \rightarrow b \mid bb$$

$$L = \{0^i 1^j \mid i \neq j, i \geq 0, j \geq 0\}$$

$$S \rightarrow 0S1 \mid A \mid B$$

$$A \rightarrow 0 \mid 0A$$

$$B \rightarrow 1 \mid 1B$$

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

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

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

$$P = \{S \rightarrow aSb \mid A \mid B, A \rightarrow a \mid aa, B \rightarrow b \mid bb\}$$

S is start symbol.

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

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

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

$$P = \{S \rightarrow 0S1 \mid A \mid B$$

$$A \rightarrow 0 \mid 0A$$

$$B \rightarrow 1 \mid 1B$$

(eg)

obtain a grammar to generate the language

$$L = \{a^{n+2} b^m \mid n \geq 0 \text{ & } m > n\}$$

Soln

$$S \rightarrow aAB$$

$$A \rightarrow aAb \mid ab$$

$$B \rightarrow bB \mid \epsilon$$

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

$$V = \{S, A, B\}, T = \{a, b\}, P = \{S \rightarrow aAB, A \rightarrow aAb \mid ab, B \rightarrow bB \mid \epsilon\}$$

S is start symbol

(Q)

Obtain a grammar to generate the language

$$L = \{a^n b^m \mid n \geq 0, m > n\}$$

Sol?

$$n=0 \quad n=1 \quad n=2$$

$$m \geq 1 \quad m \geq 2 \quad m \geq 3$$

$$L = \{\boxed{\epsilon} b b^*, \boxed{a} b b b^*, \boxed{aa} b b b b^*, \dots\}$$

~~SO SO SO ABB~~

$S \rightarrow aSb/B$ [Generates $a^n b^n \mid n \geq 0$ followed by at least one b]

$B \rightarrow bB/b$ [Generates one or more b 's]

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

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

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

$$P = \{S \rightarrow aSb/B, B \rightarrow bB/b\}$$

S is start symbol.

(Q)

Obtain a grammar to generate the language

$$L = \{a^n b^{n-3} \mid n \geq 3\}$$

Sol?

$$n=3 \quad n=4 \quad n=5$$

$$L = \{aaa, aaaab, aaaaaab, \dots\}$$

$$P = \{S \rightarrow aaat \\ \rightarrow aAb \mid \epsilon\}$$

Derivation.

Defⁿ: Let $A \rightarrow \alpha B \gamma$ & $B \rightarrow \beta$ are production in grammar G, where α, β and γ are strings of terminal and/or non-terminals. A and B are non-terminals. The non-terminal A produces the string $\alpha \beta \gamma$ by replacing the non-terminal B in $\alpha B \gamma$ by the string β by applying the production $B \rightarrow \beta$. It can be written as

$$A \Rightarrow \alpha \beta \gamma$$

This process of obtaining string of terminals &/or non-terminals from the start symbol by applying some or all productions is called derivation.

If a string is obtained by applying only one prodⁿ, then it is called one-step derivation & is denoted by the symbol ' \Rightarrow '. If one or more productions are applied to get the string $\alpha \beta \gamma$ from A then we write

$$A \Rightarrow^+ \alpha \beta \gamma$$

If two or more prodⁿ are applied to get the string $\alpha \beta \gamma$ from A, then we write

$$A \xrightarrow{*} \alpha \beta \gamma$$

(Eg)

Consider the grammar shown below from which any arithmetic expⁿ can be obtained.

$$E \rightarrow E + E$$

$$E \rightarrow E - E$$

$$E \rightarrow E * E$$

$$E \rightarrow E / E$$

$$E \rightarrow id.$$

E is considered to be the start symbol. Obtain the string id+id*id & show the derivation for it same

sol?

$$E \Rightarrow E + E$$

$$\Rightarrow id + E$$

$$\Rightarrow id + E * E$$

$$\Rightarrow id + id * E$$

$$\underline{E \Rightarrow id + id * id}$$

Sentence or Sentential form

Defⁿ: Let $G = (V, T, P, S)$ be a grammar. The string w obtained from the grammar G such that $S \Rightarrow^* w$ is called Sentence of grammar G . Here w is string of terminals.

In the above (Eg) id+id*id is the sentence of the grammar.

If there is a derivation $S \xrightarrow{*} \alpha$, where α contains

Strings of terminals and/or non-terminals, then α is called ~~sentient~~^{sent} of sentential form of G .
In the derivation shown in above eg.

$E+E$, $id+E$, $id+E*E$, $id+id+E$, $id+id*id$
are all sentential forms of the grammar.

Language

Defn: Let $G = (V, T, P, S)$ be a grammar. The language $L(G)$ generated by the grammar G is

$$L(G) = \{ w \mid S \xrightarrow{*} w \text{ & } w \in T^* \}$$

i.e., w is a string of terminals (may be ϵ) obtained from the start symbol S by applying an arbitrary number of productions.

Leftmost Derivation

Defⁿ: In the derivation process if a left most variable is replaced at every step, then the derivation is said to be leftmost.

(Eg) obtain the leftmost derivation for the string $id + id * id$ using the following grammar

$$E \rightarrow E + E$$

$$E \rightarrow E - E$$

$$E \rightarrow E * E$$

$$E \rightarrow E | E$$

$$E \rightarrow E^n E$$

$$E \rightarrow id$$

\wedge : denotes exponent of

Solⁿ

$$E \xrightarrow[lm]{+} E + E$$

$$\Rightarrow id + E$$

$$\Rightarrow id + E * E$$

$$\Rightarrow id + id * E$$

$$\Rightarrow id + id * id$$

The string $id + id * id$ is obtained from the start symbol E by applying leftmost derivation & it can be written as

$$E \xrightarrow[lm]{+} id + id * id$$

Eg) Obtain the leftmost derivation for the string aaabbabbba using the following grammar

$$S \rightarrow aB | bA$$

$$A \rightarrow aS | bAA | a$$

$$B \rightarrow bS | aBB | b$$

$$\begin{matrix} S \\ \xrightarrow{\text{in}} \\ aB \end{matrix}$$

(Applying $S \rightarrow aB$)

$$\Rightarrow aAB\bar{B} \quad (B \rightarrow aBB)$$

$$\Rightarrow aaAB\bar{B}\bar{B} \quad (B \rightarrow aBB)$$

$$\Rightarrow aaab\bar{B}\bar{B} \quad (B \rightarrow b)$$

$$\Rightarrow aaabb\bar{B} \quad (B \rightarrow b)$$

$$\Rightarrow aaabb\bar{a}\bar{B} \quad (B \rightarrow aBB)$$

$$\Rightarrow aaabbab\bar{B} \quad (B \rightarrow b)$$

$$\Rightarrow aaabbabb\bar{s} \quad (B \rightarrow bS)$$

$$\Rightarrow aaabbabb\bar{A} \quad (S \rightarrow bA)$$

$$\Rightarrow aaabbabbba \quad (A \rightarrow \omega)$$

=====.

Rightmost Derivation

Defⁿ: In the derivation process if a right most variable is replaced at every step, then the derivation is said to be rightmost.

(Eg)

obtain the rightmost derivation for the string
 $id + id * id$ using following grammar.

$$E \rightarrow E + E$$

$$E \rightarrow E * E$$

$$E \rightarrow E - E$$

$$E \rightarrow E / E$$

$$E \rightarrow id$$

sol:

$$E \xrightarrow{rm} E + E$$

$$\Rightarrow E + E * E$$

$$\Rightarrow E + E * id$$

$$\Rightarrow E + id * id$$

$$\Rightarrow id + id * id$$

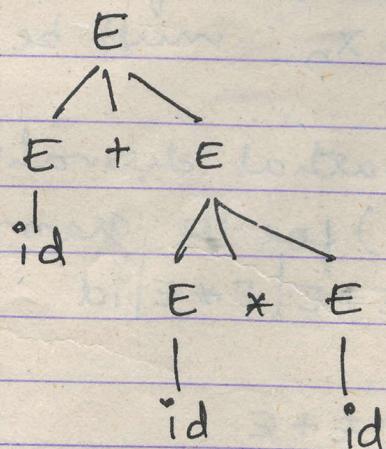
Derivation Tree (parse tree)

The derivation can be shown in the form of a tree. Such trees are called derivation or parse trees.

Defⁿ (Parse tree or derivation tree) : Let $G = (V, T, P, S)$ be a CFG. The tree is derivation tree (parse tree) with the following properties.

1. The root has the label S
2. Every vertex has a label which is in $(V \cup T \cup \epsilon)$
3. Every leaf node has label from T & an interior vertex has a label from V.
4. If a vertex is labeled A & if $x_1, x_2, x_3, \dots, x_n$ are all children of A from left, then $A \rightarrow x_1 x_2 x_3 \dots x_n$ must be a production in P.

Parse tree for above eg.



The string $\text{id} + \text{id} * \text{id}$ is called yield of this tree.

Yield of the tree

The yield of a tree is the string of symbols obtained by only reading the leaves of the tree from left to right with considering the E-symbols. The yield of a tree is always a terminal string.

Defⁿ (partial parse tree or partial derivation tree): Let $G = (V, T, P, S)$ be a CFG. The tree is partial derivation tree (parse tree) with the following properties:

- ① The ^{root}_x has the label S
- ② Every vertex has a label which is in (VUTVE)
- ③ Every leaf node has a label from (VUTVE).
- ④ If a vertex is labeled A & if $x_1, x_2, x_3, \dots, x_n$ are all children of A from left. Then $A \rightarrow x_1, x_2, x_3, \dots, x_n$ must be a prodⁿ in P.

eg consider the partial derivation (by applying right most derⁿ) for the grammar.

$$E \rightarrow E + E \mid E * E \mid id$$

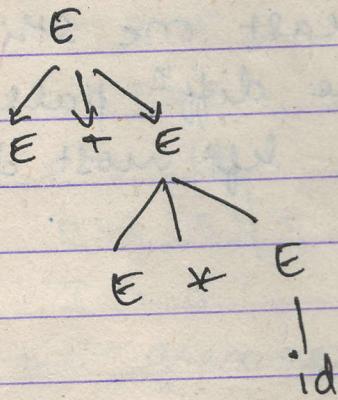
is shown below

$$E \rightarrow E + E$$

$$\Rightarrow E + E * E$$

$$\Rightarrow E + E * id$$

for this partial right most derivation, the partial derivation tree is shown below



It is clear from the parse tree & partial parse tree that all the leaves in parse tree are the symbols from (TUE) whereas in partial parse tree the leaves will be from (VUTUE).

Ambiguous Grammar

Defⁿ: Let $G = (V, T, P, S)$ be a context free grammar. A grammar G is ambiguous if & if there exists at least one string $w \in T^*$ for which two or more diffⁿ parse trees exists by applying either the left most derivation or right most derivation.

Note :

- 1) obtain the leftmost derivation & get a string w . obtain the rightmost derivation & get a string w . for both the derivations construct the parse tree. If there are two diffⁿ parse trees, then the grammar is ambiguous.
- 2) obtain the string w by applying leftmost derivation twice & construct the parse tree. If the two parse trees are different, the grammar is ambiguous.
- 3) obtain the string w by applying rightmost derivation twice & construct the parse tree. If the two parse trees are diffⁿ, the grammar is ambiguous.

(eg) Consider the grammar shown below

$$E \rightarrow E+E$$

$$E \rightarrow E-E$$

$$E \rightarrow E * E$$

$$E \rightarrow E/E$$

$$E \rightarrow (E) | I$$

$$I \rightarrow id$$

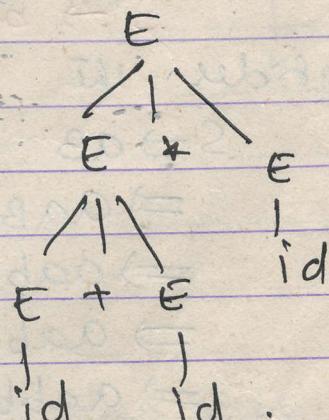
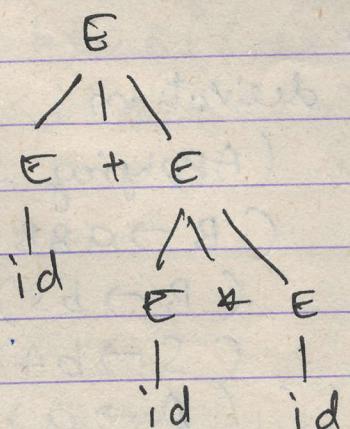
Show that the grammar is ambiguous.

The sentence $id + id * id$ can be obtained from leftmost derivation in two ways as shown below

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

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

The corresponding derivation trees for the two leftmost ~~derivation~~ derivation are shown below



Since the two parse trees are diffⁿ for the same sentence $id + id * id$ by applying leftmost derivation, the grammar is ambiguous.

(eg)

Is the following grammar ambiguous?

$$S \rightarrow aS \mid x$$

$$X \rightarrow aX \mid a$$

Sol:

Consider the two leftmost derivation for the string aaaa.

$$\begin{aligned} S &\Rightarrow aS \\ &\Rightarrow aaaS \\ &\Rightarrow aaas \\ &\Rightarrow aaax \\ &\Rightarrow aaaa \end{aligned}$$

$$\begin{aligned} S &\Rightarrow X \\ &\Rightarrow aX \\ &\Rightarrow aax \\ &\Rightarrow aaax \\ &\Rightarrow aaaa \end{aligned}$$

Since there are two leftmost derivations for the same sentence aaaa, the given grammar is ambiguous.

Show that the grammar is ambiguous for the string

(eg)

Is the following grammar ambiguous? aabbab

$$S \rightarrow aB \mid bA$$

$$A \rightarrow aS \mid bAA \mid a$$

$$B \rightarrow bS \mid aBB \mid b$$

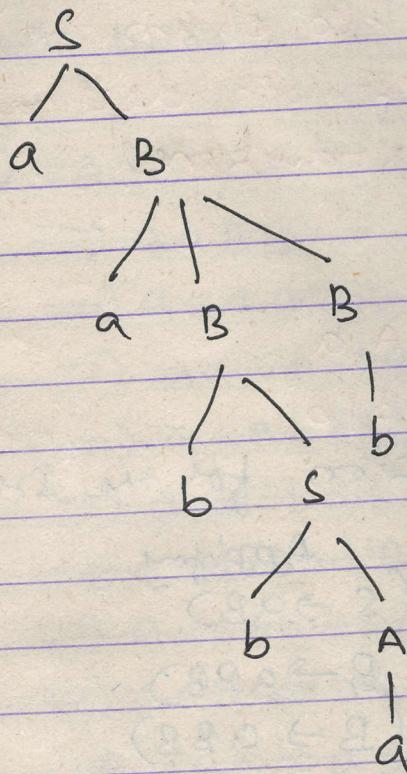
$$\begin{aligned} S &\Rightarrow aB \\ &\Rightarrow abS \\ &\Rightarrow abab \\ &\Rightarrow abab \end{aligned}$$

Sol?

Consider the leftmost derivation

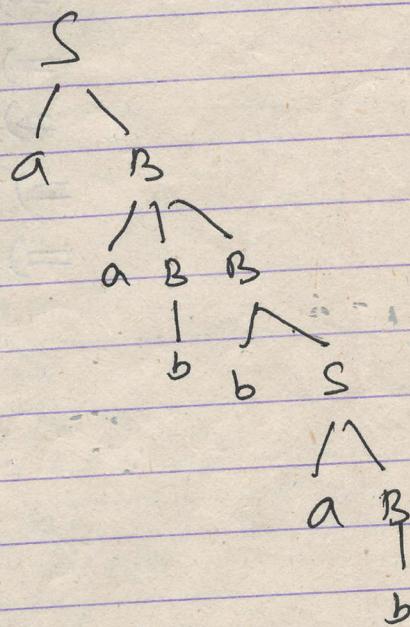
$$\begin{aligned} S &\Rightarrow aB && (\text{Applying } S \rightarrow aB) \\ &\Rightarrow aabbB && (B \rightarrow aBB) \\ &\Rightarrow aabsB && (B \rightarrow bS) \\ &\Rightarrow aabbAB && (S \rightarrow bA) \\ &\Rightarrow aabbAB && (A \rightarrow a) \\ &\Rightarrow aabbab && (B \rightarrow b) \end{aligned}$$

Parse tree



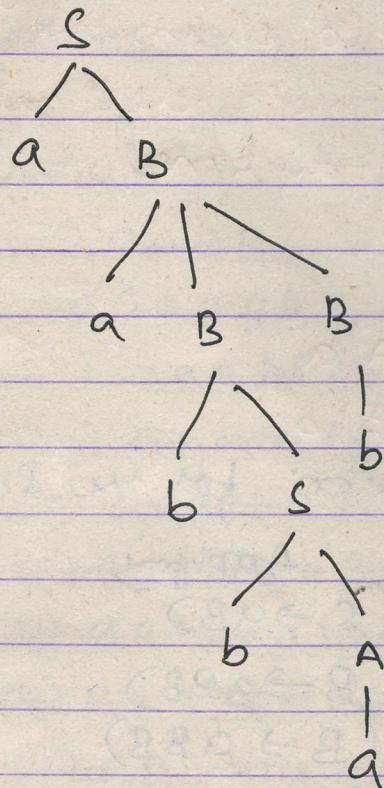
The same string aabbab can be obtained again by applying leftmost derivation as shown below.

$$\begin{aligned}
 & S \xrightarrow{\text{apply } S \rightarrow aB} aB \\
 & \Rightarrow aAB \xrightarrow{B \rightarrow aBB} aABB \\
 & \Rightarrow aabB \xrightarrow{B \rightarrow b} aabb \\
 & \Rightarrow aabbs \xrightarrow{B \rightarrow bS} aabbaB \\
 & \Rightarrow aabbab \xrightarrow{S \rightarrow aB} aabbab \\
 & \Rightarrow aabbab \xrightarrow{B \rightarrow b} aabbab
 \end{aligned}$$



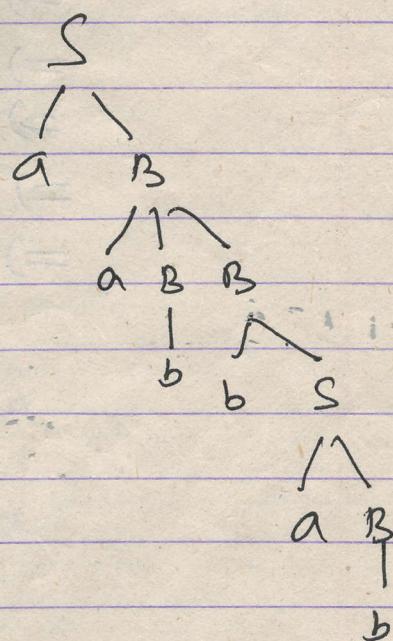
There are two parse trees for the string aabbab by applying leftmost derivation so the given grammar is ambiguous.

Parse tree



The same string aabbab can be obtained again by applying leftmost derivation as shown below.

$\text{Applying } S \Rightarrow aB$
 $\Rightarrow aAB$ ($S \Rightarrow aB$)
 $\Rightarrow aABBB$ ($B \Rightarrow aBB$)
 $\Rightarrow aabbB$ ($B \Rightarrow b$)
 $\Rightarrow aabbs$ ($B \Rightarrow bS$)
 $\Rightarrow aabbabS$ ($S \Rightarrow aB$)
 $\Rightarrow aabbab$ ($B \Rightarrow b$)



There are two parse trees for the string aabbab by applying leftmost derivation & so the given grammar is ambiguous.

Eg)

Obtain the string aaabbabba by applying leftmost derivation & the parse tree for the grammar shown below. Is it possible to obtain the same string again by applying leftmost derivation but by selecting different productions?

$$S \rightarrow aB | bA$$

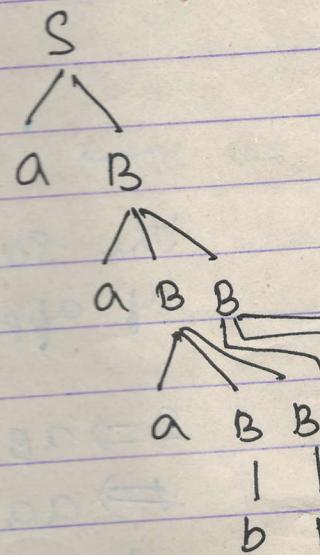
$$A \rightarrow aS | bAA | a$$

$$B \rightarrow bS | aBB | b$$

Sol:

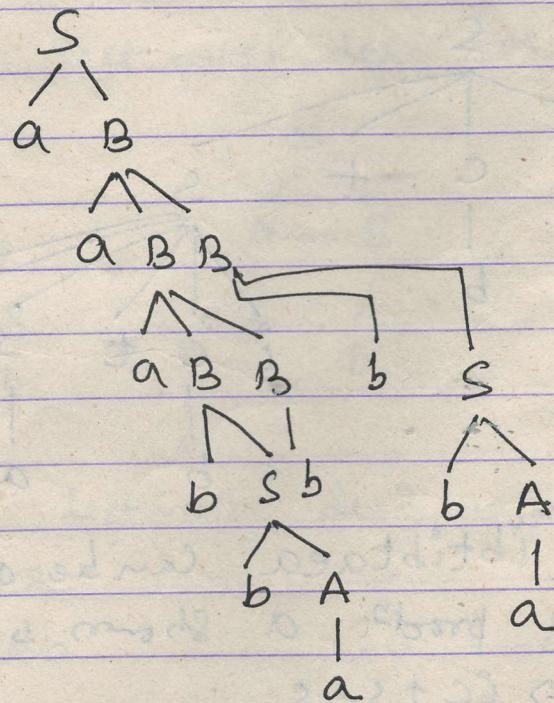
The leftmost derivation for the string aaabbabba

Applying
 $S \Rightarrow aB \quad (S \rightarrow aB)$
 $\Rightarrow aAB \quad (B \rightarrow aBB)$
 $\Rightarrow aaABB \quad (B \rightarrow aBB)$
 $\Rightarrow aaabBB \quad (B \rightarrow b)$
 $\Rightarrow aaabbB \quad (B \rightarrow b)$
 $\Rightarrow aaabbAB \quad (B \rightarrow aBB)$
 $\Rightarrow aaabbabB \quad (B \rightarrow b)$
 $\Rightarrow aaabbabbs \quad (B \rightarrow bS)$
 $\Rightarrow aaabbabbbA \quad (S \rightarrow bA)$
 $\Rightarrow aaabbabbbba \quad (A \rightarrow a)$



The leftmost der^u for the same string
 aaabbabbba but by applying diff^u set of prod^u
 is shown below:

- $$\begin{aligned}
 S &\Rightarrow aB & (S \rightarrow aB) \\
 &\Rightarrow aAB B & (B \rightarrow aBB) \\
 &\Rightarrow aaABBB & (B \rightarrow aBB) \\
 &\Rightarrow aaabsBB & (B \rightarrow bs) \\
 &\Rightarrow aaabbABB & (S \rightarrow bA) \\
 &\Rightarrow aaabbabBB & (A \rightarrow a) \\
 &\Rightarrow aaabbabB & (B \rightarrow b) \\
 &\Rightarrow aaabbabbS & (B \rightarrow bS) \\
 &\Rightarrow aaaabbabbba & (S \rightarrow bA) \\
 &\Rightarrow aaabbabbba & (A \rightarrow a)
 \end{aligned}$$



(eg)

IS THE following grammar ambiguous?

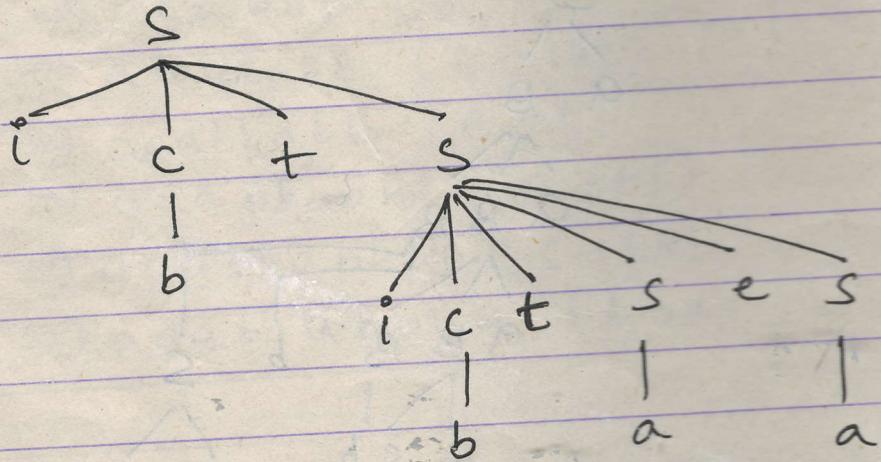
$$S \rightarrow icts \mid ictses \mid a$$

$$C \rightarrow b$$

Soln: The string ibtibtaea can be obtained by applying the leftmost deriv as shown below

$$\begin{aligned} S &\Rightarrow icts \\ &\Rightarrow ibtS \\ &\Rightarrow ibticts \\ &\Rightarrow ibtibtSes \\ &\Rightarrow ibtibtaes \\ &\Rightarrow ibtibtaea \end{aligned}$$

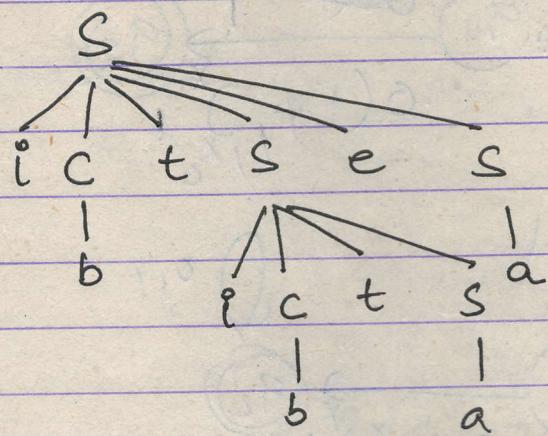
pulse tree



The string ibtibtaea can be obtained by using diff set of prod as shown below -

$$\begin{aligned} S &\Rightarrow ictses \\ &\Rightarrow \cancel{i}btses \\ &\Rightarrow ibticts \\ &\Rightarrow ibtibtSes \\ &\Rightarrow ibtibtaes \\ &\Rightarrow ibtibtaea \end{aligned}$$

The parse tree for this is shown below



Is the grammar ambiguous?

$$S \rightarrow AB \mid aab$$

$$A \rightarrow a \mid Aa$$

$$B \rightarrow b$$

$$S \rightarrow AB \quad S \rightarrow aab$$

$$\Rightarrow Aab$$

$$\Rightarrow aab$$

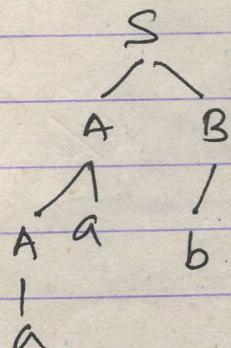
consider the left most deriv for the string aab

$$S \Rightarrow AB$$

$$\Rightarrow AaB$$

$$\Rightarrow aaB$$

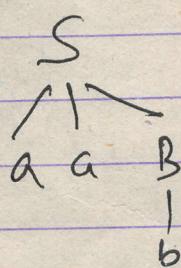
$$\Rightarrow aab$$



consider the left most deriv again for the string aab

$$S \Rightarrow AAB$$

$$\Rightarrow aab$$



Since there are two parse trees for the string aab, the given grammar is ambiguous.

Properties of context-free languages

The goal of this section is to show that every CFL (without ϵ) is generated by a CFG in which all prod \cong s are of the form $A \rightarrow BC$ or $A \rightarrow a$, where $A, B \in V$ & C are variables & a is a terminal. This form is called Chomsky Normal form (CNF).

To get there, we need to make a number of preliminary simplifications, which are themselves useful in various ways:

- 1) We must eliminate useless symbols, those variables or terminals that do not appear in any derivation of a terminal string from the start symbol.
- 2) We must eliminate ϵ -prod \cong , those of the form $A \rightarrow \epsilon$ for some variable A .
- 3) We must eliminate unit prod \cong , those of the form $A \rightarrow B$ for variable $A \neq B$.

Eliminating useless symbols:

We say a symbol x is useful for a grammar $G = (V, T, P, S)$ if there is a some derivation of the form $S \xrightarrow{*} \alpha x \beta \xrightarrow{*} w$, where w is in T^* . Note that x may be in either V or T , & the sentential form $\alpha x \beta$ might be the first or last in the derivation. If x is not useful, we say it is useless.

Evidently, omitting useless symbols from a grammar will not change the language generated. So we may as well detect & eliminate all useless symbols.

our approach to eliminating useless symbols begins by identifying the two things a symbol has to be able to do to be useful:

- 1) we say x is generating if $x^* \Rightarrow w$ for some terminal string w . Note that every terminal is generating, since w can be that terminal itself, which is derived by zero steps.
- 2) we say x is reachable if there is a derivation $S^* \Rightarrow x \in B$ for some $x \in B$.

surely a symbol that is useful will be both generating & reachable. If we eliminate the symbols that are not generating first & then eliminate from the remaining grammar those symbols that are not reachable we have only the useful symbols left.

Ex: Eliminate useless symbols from the following grammar

$$S \rightarrow AB | a$$

$$A \rightarrow a.$$

HY: Step: find the set of vae & the prod^{nt} from which we get only string of terminals.

OV	nr	prod ^{nt}
\emptyset	S, A	$S \rightarrow a$ $A \rightarrow a$
S, A	S, A	$S \rightarrow a$ $A \rightarrow a$

The resulting grammar, from which we get only string of terminals is given by.

$$G_1 = (V_1, T_1, P_1, S)$$

Where

$$V_1 = \{ S, A \}$$

$$T_1 = \{ a \}$$

$$P_1 = \{ S \rightarrow a, A \rightarrow a \}$$

Step: Eliminate the symbol & production which are not reachable from the start symbol.

P'	T'	V'
-	-	S
$S \rightarrow a$	a	S

So the final grammar obtained after eliminating useless symbols & productions is given by

$$G' = (V', T', P', S)$$

Where

$$V' = \{ S \}$$

$$T' = \{ a \}$$

$$P' = \{ S \rightarrow a \}$$

S is the start symbol.

Eliminate the useless symbols in the grammar.

$$S \rightarrow aA | bB$$

$$A \rightarrow aA | a$$

$$B \rightarrow bB$$

$$D \rightarrow ab | Ea$$

$$E \rightarrow ac | d$$

OV	nv	Prod \Rightarrow
\emptyset	A, E, D	$A \rightarrow a$ $E \rightarrow d$ $D \Rightarrow ab$
A, E, D	A, E, D, S	$S \rightarrow aA$ $A \rightarrow aA a$ $D \rightarrow ab Ea$
A, D, E, S	A, D, E, S	-

The resulting grammar $G_1 = (V_1, T_1, P_1, S)$ where

$$V_1 = \{A, D, E, S\}$$

$$T_1 = \{a, b, d\}$$

$$P_1 = \{ S \rightarrow aA, A \rightarrow aA | a, D \rightarrow ab | Ea, E \rightarrow d \}$$

S is the start symbol.

P'	T'	V'
$S \rightarrow aA$ $A \rightarrow aA/a$	a a	s s, A s, A

The resulting grammar $G' = (V', T', P', S)$
where

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

$$T' = \{a\}$$

$$P' = \{S \rightarrow aA, A \rightarrow aA/a\}$$

S is the start symbol.

(Q) Eliminate useless symbols from the following grammar

$$S \rightarrow aAB$$

$$A \rightarrow AB/B$$

$$B \rightarrow aB/b/bC$$

$$D \rightarrow EA$$

$$E \rightarrow a/aE/bC$$

Sol)

OV	NV	Productions
\emptyset	B, E	$B \rightarrow b$ $E \rightarrow a$
B, E	A, B, D, E	$A \rightarrow aB/B$ $B \rightarrow aB$ $D \rightarrow EA$ $E \rightarrow aE$

OV	nv	prod^n.
A, B, D, E	S, A, B, D, E	$S \rightarrow aAB$
S, A, B, D, E	-	aaa -

$$G_1 = (V_1, T_1, P_1, S)$$

$$V_1 = \{ S, A, B, D, E \}$$

$$T_1 = \{ a, b \}$$

$$P_1 = \{ S \rightarrow aAB, A \rightarrow aB/B, B \rightarrow aB/b, D \rightarrow Ea, E \rightarrow aE/a \}$$

prod^n(p')	T'	V'
-	-	S
S \rightarrow aAB	a	S, A, B
A \rightarrow aB/B	a	S, A, B
B \rightarrow aB/b	a, b	S, A, B

$$G' = (V', T', P', S)$$

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

$$T' = \{ a, b \}$$

$$P' = \{ S \rightarrow aAB, A \rightarrow aB/B, B \rightarrow aB/b \}$$

S is the start symbol.

Eg) Eliminate useless symbols from the following grammar

$$S \rightarrow aA|a|Bb|cc$$

$$A \rightarrow AB$$

$$B \rightarrow a|Aa$$

$$C \rightarrow cCD$$

$$D \rightarrow ddd$$

Sol): Step 1:

OV	nv	prod?
\emptyset	S, B, D	$S \rightarrow a$ $B \rightarrow a$ $D \rightarrow ddd$
S, B, D	S, A, B, D	$S \rightarrow Bb$ $A \rightarrow AB$
S, A, B, D	S, A, B, D	$S \rightarrow aA$ $A \rightarrow AB$ $B \rightarrow Aa$

$$G_1 = (V_1, T_1, P_1, S)$$

$$V_1 = \{ S, A, B, D \}$$

$$T_1 = \{ a, b, d \}$$

$$P_1 = \{ S \rightarrow a | Bb | aA, A \rightarrow AB, B \rightarrow a | Aa, D \rightarrow ddd \}$$

Production (P')

	T'	V'
$S \rightarrow a Bb aA$	-	S
$A \rightarrow aB$	a, b	S, A, B
$B \rightarrow a Aa$	a, b	S, A, B
	a, b	S, A, B

The resulting grammar is

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

$$T' = \{a, b\}$$

$$P' = \{S \rightarrow a | Bb | aA, A \rightarrow aB, B \rightarrow a | Aa\}$$

S is the start symbol.

Eliminating ϵ -productions.

A production of the form $A \rightarrow \epsilon$ is undesirable in a CFG, unless an empty string is derived from the start symbol. Suppose, the language generated from a grammar G does not derive any empty string & the grammar consist of ϵ -productions. Such ϵ -prodⁿs can be removed.

Defn: Let $G = (V, T, P, S)$ be a CFG. A production in P of the form

$$A \rightarrow \epsilon$$

is called an ϵ -prodⁿ or null prodⁿ. After applying the production the variable A is erased. For each A in V , if there is a derivation of the form

$$A \xrightarrow{*} \epsilon$$

then A is a nullable variable.

(Eg) In grammar if there is any prod of the form

$$B \rightarrow \epsilon$$

$$C \rightarrow \epsilon$$

are ϵ -prodⁿ and the variable B, C are nullable variables. If there is a production

$$A \rightarrow BC$$

as both B, C are nullable variables, then A is also a nullable variable.

Eliminate all ϵ -productions from the grammar

$$S \rightarrow ABCa/bD$$

$$A \rightarrow BC/b$$

$$B \rightarrow b/\epsilon$$

$$C \rightarrow C/\epsilon$$

$$D \rightarrow d$$

$$B \rightarrow \epsilon$$

$$C \rightarrow \epsilon$$

$$A \rightarrow BC$$

$V = \{B, C, A\}$ are all nullable variables.

productions	Resulting prod ⁿ
$S \rightarrow ABCa$	$S \rightarrow ABCa/BCa/ACa/ABA/ca/aa/Ba/a$
$S \rightarrow bD$	$S \rightarrow bD$
$A \rightarrow BC/b$	$A \rightarrow BC/B/C/b$
$B \rightarrow b/\epsilon$	$B \rightarrow b$
$C \rightarrow C/\epsilon$	$C \rightarrow C$
$D \rightarrow d$	$D \rightarrow d$

The grammar $G^1 = (V^1, T^1, P^1, S)$ where

$$V^1 = \{S, A, B, C, D\}$$

$$T^1 = \{a, b, c, d\}$$

$$P^1 = \{ S \rightarrow ABCa/BCa/ACa/ABA/ca/aa/Ba/a/bD \}$$
$$A \rightarrow BC/B/C/b$$

$$B \rightarrow b$$

$$C \rightarrow C$$

$$D \rightarrow d$$

S is the start symbol.

(Eq)

Eliminate all ϵ -productions from the grammar

$$S \rightarrow BAAB$$

$$A \rightarrow 0A2 | 2AO | \epsilon$$

$$B \rightarrow AB | 1B | \epsilon$$

Solⁿ

$$A \rightarrow \epsilon$$

$$B \rightarrow \epsilon$$

$$S \rightarrow BAAB$$

so S, A, B are all nullable variables.

Productions

$$S \rightarrow BAAB$$

$$A \rightarrow 0A2$$

$$A \rightarrow 2AO$$

$$B \rightarrow AB$$

$$B \rightarrow 1B$$

Resulting prod $\sim(p)$

$$S \rightarrow BAAB | AAB | BAB | BAA | AB | BB | BA | AA | A | B$$

$$BB | BA | AA | A | B$$

$$A \rightarrow 0A2 | 02$$

$$A \rightarrow 2AO | 2O$$

$$B \rightarrow AB | B | A$$

$$B \rightarrow 1B | 1$$

The grammar $G = (V', T', P', S)$ where

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

$$T' = \{0, 1, 2\}$$

$$P' = \{S \rightarrow BAAB | AAB | BAB | BAA | AB | BB | BA | AA | A | B\}$$

$$A \rightarrow 0A2 | 02 | 2AO | 2O$$

$$B \rightarrow AB | B | A | 1B | 1\}$$

S is the start symbol.

Eliminating unit productions

A unit production is a prodⁿ of the form

$A \rightarrow B$, where both A & B are variables.

Eliminate all unit prodⁿs from the grammar

$$S \rightarrow AB$$

$$A \rightarrow a$$

$$B \rightarrow C \mid b$$

$$C \rightarrow D$$

$$D \rightarrow E \mid bc$$

$$E \rightarrow d \mid Ab$$

Non unit prodⁿ

unit prodⁿ

$$S \rightarrow AB$$

$$B \rightarrow C$$

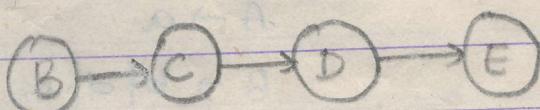
$$A \rightarrow a$$

$$C \rightarrow D$$

$$B \rightarrow b$$

$$D \rightarrow E$$

$$D \rightarrow bc$$



$$E \rightarrow d \mid Ab$$

$$E \rightarrow d \mid Ab$$

Since $D \xrightarrow{*} E$

$$D \rightarrow d \mid Ab$$

The resulting D prodⁿ are

$$D \rightarrow d \mid Ab \mid bc$$

Since $C \xrightarrow{*} D \xrightarrow{*} E$

$$C \rightarrow d | Ab | bC$$

Since $B \xrightarrow{*} C, C \xrightarrow{*} D, D \xrightarrow{*} E$

$$B \rightarrow b | d | Ab | bC$$

The final grammar obtained after eliminating unit productions is shown below:

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

$$T = \{ a, b, d \}$$

$$P = \{ S \rightarrow AB$$

$$A \rightarrow a$$

$$B \rightarrow b | d | Ab | bC$$

$$C \rightarrow bC | d | Ab$$

$$D \rightarrow bC | d | Ab$$

$$E \rightarrow d | Ab$$

}

S is the start symbol.

(Eq)

Eliminate unit productions from the grammar

$$S \rightarrow A0 | B$$

$$B \rightarrow A | 11$$

$$A \rightarrow 0 | 12 | B$$

Non-unit prod^u

$S \rightarrow A0$

$B \rightarrow 11$

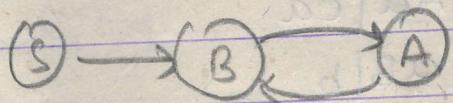
$A \rightarrow 0|12$

Unit prod^u

$S \rightarrow B$

$B \rightarrow A$

$A \rightarrow B$



It is clear from the dependence graph that
 $S \xrightarrow{*} B$, $S \xrightarrow{*} A$, $B \xrightarrow{*} A$ & $A \xrightarrow{*} B$ so the new
 prod^u for S, A & B are

$S \rightarrow A0|11|0|12$

$B \rightarrow 0|12|11$

$A \rightarrow 0|12|11$

The resulting grammar without unit prod^u

$G' = (V', T', P', S)$

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

$T' = \{0, 1, 2\}$

$P' = \{S \rightarrow A0|11|0|12,$

$B \rightarrow 0|12|11,$

$A \rightarrow 0|12|11\}$

S is the start symbol.

(Eq)

eliminate unit prodⁿs from the grammar

$$S \rightarrow Aa | B | ca$$

$$B \rightarrow aB | b$$

$$C \rightarrow Db | D$$

$$D \rightarrow E | d$$

$$E \rightarrow ab$$

Sol?

Nm-unit prodⁿ

$$S \rightarrow Aa | ca$$

$$B \rightarrow aB | b$$

$$C \rightarrow Db$$

$$D \rightarrow d$$

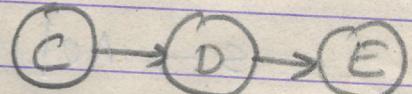
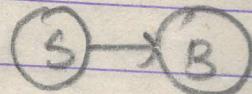
$$E \rightarrow ab$$

Unit prodⁿ

$$S \rightarrow B$$

$$C \rightarrow D$$

$$D \rightarrow E$$



$$S \xrightarrow{*} B$$

$$S \rightarrow Aa | aB | b | ca$$

$$\cancel{C \rightarrow D} \quad D \xrightarrow{*} E$$

$$D \rightarrow ab | d$$

$C \xrightarrow{*} D \quad D \xrightarrow{*} E$

$C \xrightarrow{*} Db \mid ab \mid d$

The resulting grammar

$$G' = (V', T', P', S)$$

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

$$T' = \{ a, b, d \}$$

$$P' = \{ S \rightarrow Aa \mid aB \mid b \mid ca \}$$

$$B \rightarrow aB \mid b$$

$$C \rightarrow Db \mid ab \mid d$$

$$D \rightarrow d \mid ab$$

$$E \rightarrow ab \}$$

S is the start symbol.

Chomsky Normal form.

Every nonempty CFL with ϵ has a grammar G in which all productions are in one of two forms, either

1) $A \rightarrow BC$, where A, B & C are variable

2) $A \rightarrow a$, where A is a variable & a is a terminal

Further, G has no useless symbols. Such a grammar is said to be in chomsky normal form, or CNF.

(Eg) Convert the following grammar to CNF

- ① remove ϵ -prod
- ② remove unit-prod
- ③ remove useless

$$S \rightarrow OA | IB$$

$$A \rightarrow OAA | IS | I$$

$$B \rightarrow IBB | OS | O$$

Soln

Std form

$$A \rightarrow I$$

$$B \rightarrow O$$

$$S \rightarrow OA | IB$$

$$S \rightarrow B_0 A | B_1 B , \quad A \rightarrow B_0 AA | B_1 S , \quad B \rightarrow BBB | B_0 S$$

$$B_0 \rightarrow O$$

$$B_0 \rightarrow O$$

$$B_0 \rightarrow O$$

$$B_1 \rightarrow I$$

$$B_1 \rightarrow I$$

$$B_1 \rightarrow I$$

Grammar obtained

$$V_1 = \{ S, A, B, B_0, B_1 \}$$

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

$$P_1 = \{ S \rightarrow B_0 A | B_1 B , \quad A \rightarrow B_0 AA | B_1 S | I , \quad B \rightarrow B_1 BB | B_0 S \}$$

$$B_0 \rightarrow O$$

$$B_1 \rightarrow I \}$$

S is the start symbol.

$$\begin{aligned}
 S &\rightarrow B_0 A \mid B_1 B \\
 A &\rightarrow B_0 AA \mid B_1 S \mid 1 \\
 B &\rightarrow B_1 BB \mid B_0 S \mid 0 \\
 B_0 &\rightarrow 0 \\
 B_1 &\rightarrow 1
 \end{aligned}$$

CNF prod'y are

$$\begin{aligned}
 S &\rightarrow B_0 A \mid B_1 S \\
 A &\rightarrow B_1 S \mid 1 \\
 B &\rightarrow B_0 S \mid 0 \\
 B_0 &\rightarrow 0 \\
 B_1 &\rightarrow 1
 \end{aligned}$$

Productions which are not in CNF form.

$$\begin{aligned}
 A &\rightarrow B_0 AA \\
 B &\rightarrow B_1 BB \\
 A &\rightarrow B_0 AA, \quad A \rightarrow B_0 D_1, \\
 D_1 &= AA \\
 \cancel{B \rightarrow B_1 BB} & \quad B \rightarrow B_1 BB \\
 B &\rightarrow B_1 D_2 \\
 D_2 &= BB
 \end{aligned}$$

$$\begin{aligned}
 G_1 &= \{ S, A, B, B_0, B_1, D_1, D_2 \} \\
 T_1 &= \{ 0, 1 \}
 \end{aligned}$$

$$\begin{aligned}
 P_1 &= \{ S \rightarrow B_0 A \mid B_1 S, \quad A \rightarrow B_0 D_1 \mid B_1 S \mid 1, \quad B \rightarrow B_1 D_2 \mid B_0 S \mid 0 \\
 B_0 &\rightarrow 0, \quad D_1 \rightarrow AA \\
 B_1 &\rightarrow 0, \quad D_2 \rightarrow BB \} \\
 S &\text{ is the start symbol}
 \end{aligned}$$

(Eq)

Convert the following grammar to CNF

$$I \rightarrow c | d | I_a | I_b | I_0 | I_i$$

$$F \rightarrow I | (E)$$

$$T \rightarrow F | T * F$$

$$E \rightarrow T | E + T$$

(Eq)

Find a grammar equivalent to

$$S \rightarrow AB | CA$$

$$A \rightarrow a$$

$$B \rightarrow BC | AB$$

$$C \rightarrow aB | b$$

with no useless symbols

(Eq)

Begin with the grammar

$$S \rightarrow ASB | \epsilon$$

$$A \rightarrow aAS | a$$

$$B \rightarrow SbS | A | bb$$

(a) Are there any useless symbols? Eliminate them if so.

(b) Eliminate ϵ -productions

(c) Eliminate unit-productions

(d) put the grammar into CNF.

Greibach Normal Form (GNF).

In GNF there is no restriction on the number of symbols on the right hand side, but there is restriction on the terminals & variables appear on the right hand side of the prodⁿ.

Defⁿ: Let $G = (V, T, P, S)$ be a CFG. The CFG G is said to be in GNF if all the productions are of the form

$$\boxed{A \rightarrow \alpha\beta}$$

where $\alpha \in T$ and $\beta \in V^*$, i.e. the first symbol on the right hand side of the prodⁿ must be a terminal & it can be followed by zero or more variables.

(Q)

Convert the following grammar to GNF.

$$S \rightarrow AB \mid 0$$

$$A \rightarrow 00A \mid 1A1$$

$$B \rightarrow 1A1$$

$$S \rightarrow ABA_1 \mid 0$$

$$A \rightarrow A_0 A_0 A \mid A_1 A A_1$$

$$B \rightarrow A_1 A A_1$$

$$A_1 \rightarrow 1$$

$$A_0 \rightarrow 0$$

Now restrict the no of variables on the right hand side of the prodⁿ to two & the resulting grammar in CNF notation is

$$S \rightarrow AD_1 | 0$$

$$A \rightarrow A_0 D_2 | A_1 D_3$$

$$B \rightarrow A_1 D_3$$

$$A_1 \rightarrow 1$$

$$A_0 \rightarrow 0$$

$$D_1 \rightarrow BA_1$$

$$D_2 \rightarrow A_0 A$$

$$D_3 \rightarrow AA_1$$

Now let us rename all the variables as shown below:

$$\text{Let } S = A_1, A = A_2, B = A_3, A_0 = A_4, A_1 = A_5,$$

$$D_1 = A_6, D_2 = A_7, D_3 = A_8$$

Now the grammar can be rewritten as:

$$A_1 \rightarrow A_2 A_6 | 0$$

$$A_2 \rightarrow A_4 A_7 | A_5 A_8$$

$$A_3 \rightarrow A_5 A_8$$

$$A_5 \rightarrow 1$$

$$A_4 \rightarrow 0$$

$$A_6 \rightarrow A_3 A_5$$

$$A_7 \rightarrow A_4 A_2$$

$$A_8 \rightarrow A_2 A_5$$

In the above prodⁿ, note that A_4 & A_5 -prodⁿ are in GNF

consider A_3 prodⁿ: Substituting for A_5 in A_3 -prodⁿ we get

$$A_3 \rightarrow A_5 A_8 = 1 A_8 \rightarrow \text{GNF}.$$

consider A_2 prodⁿ

$$A_2 \rightarrow A_4 A_7 \mid A_5 A_8 = 0 A_7 \mid 1 A_8 \rightarrow \text{GNF}.$$

Consider A_1 prodⁿ.

$$A_1 \rightarrow A_2 A_6 \mid 0 = (0 A_7 \mid 1 A_8) A_6 \mid 0 = 0 A_7 A_6 \mid 1 A_8 A_6 \mid 0 \xrightarrow{\text{GNF}}$$

Consider A_6 prodⁿ.

$$A_6 \rightarrow A_3 A_5 \Rightarrow \cancel{(A_5 A_8)} A_5$$

$$A_6 \rightarrow A_5 A_8 A_5 = 1 A_8 A_5 \rightarrow \text{GNF}$$

Consider A_7 prodⁿ

$$A_7 \rightarrow A_4 A_2 = 0 A_2 \rightarrow \text{GNF}.$$

Consider A_8 prodⁿ

$$\begin{aligned} A_8 \rightarrow A_2 A_5 &= (0 A_7 \mid 1 A_8) A_5 \\ &= 0 A_7 A_5 \mid 1 A_8 A_5 \rightarrow \text{GNF} \end{aligned}$$

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

$$V = \{ A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8 \}$$

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

$$P = \{$$

$$A_1 \rightarrow 0 A_7 A_6 \mid 1 A_8 A_6 \mid 0$$

$$A_2 \rightarrow 0 A_7 \mid 1 A_8$$

$$A_3 \rightarrow 1 A_8$$

$$A_4 \rightarrow 0$$

$$A_5 \rightarrow 1$$

$$A_6 \rightarrow 1 A_8 A_5$$

$$A_7 \rightarrow 0 A_2$$

$$A_8 \rightarrow 0 A_7 A_5 \mid 1 A_7 A_5$$

S is start symbol.

(Eg)

Convert the following grammar

$$A \rightarrow BC$$

$$B \rightarrow CA \mid b$$

$$C \rightarrow AB \mid a$$

into GNF.