ch-12 Context-Free Grammars.

Context free Grammars :- CFG is a collection of three things:

1. An alphabet Σ of letters called terminals from which we are going to make strings that will be the words of a language.
2. A set of symbols called non-terminals, one of which is the symbol S, standing for "start here".
3. A finite set of productions of the form

one Non-terminal → finite string of terminals and/or Nonterminals.

\# Nonterminals designated by capital letters.
terminals are designated by lower letters.

Context free Languages :- A language generated by a CFG is called a context-free language (CFL).

Examples of CFG's

1). $L = a^*$

		OR		
Prod 1	$S \rightarrow aS$		Prod 1	$S \rightarrow SS$
Prod 2	$S \rightarrow \Lambda$		Prod 2	$S \rightarrow a$
			Prod 3	$S \rightarrow \Lambda$

2) $L = (a+b)^*$ except Λ (null string)

$S \rightarrow aS$

$S \rightarrow bS$

$S \rightarrow a$

$S \rightarrow b$

Scanned by CamScanner

3) $(a+b)^*$ [To generate any word].

S → aS
S → bS
S → a
S → b
S → Λ

OR

S → aS
S → bS
S → Λ

OR

S → X
S → Y
X → Λ
Y → aY
Y → bY
Y → a
Y → b

4) lang. with 'aa' in them somewhere.

[anything aa anything] OR $(a+b)^* aa (a+b)^*$

S → XaaX
X → aX
X → bX
X → Λ

OR

S → XY
X → aX
X → bX
X → a
Y → Ya
Y → Yb
Y → a

5) strings that end with 'a'.

X → aX
X → bX
X → a

strings that start with 'a'

Y → Ya
Y → Yb
Y → a.

CFG for no three consecutive a's.

S → aX | bS | Λ
X → aY | bS | Λ
Y → bS | Λ

Scanned by CamScanner

7). For the language Even-Even.

$S \to SS$

$S \to$ Balanced S

$S \to S$ Balanced

$S \to \Lambda$

$S \to$ unBalanced S unbalanced.

Balanced $\to aa$

Balanced $\to bb$

unbalanced $\to ab$

unbalanced $\to ba$

8). for the language $a^n b^n$

$S \to aSb$

$S \to \Lambda$

). Even Palindrome.

$S \to aSa$

$S \to bSb$

$S \to \Lambda$

odd Palindrome.

$S \to aSa$

$S \to bSb$

$S \to a$

$S \to b$.

11). entire Language Palindrome.

$S \to aSa$

$S \to bSb$

$S \to a$

$S \to b$

$S \to \Lambda$

Scanned by CamScanner

for the language $a^n b a^n$

$S \to aSa$
$S \to b$.

OR $S \to aSa \mid b$

Backus - Naur form

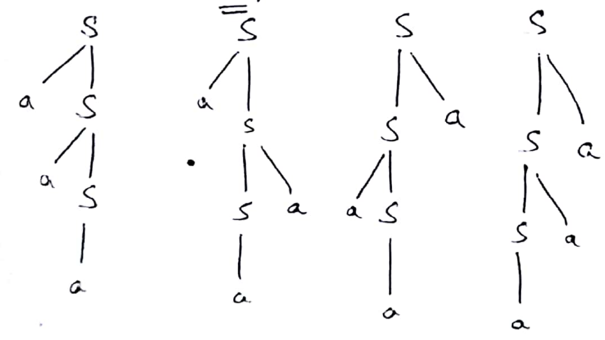
following changes are to be made in general form.

1) ::= instead of →
2) nonterminals are called variables.
3) use epsilon (ε) or lambda (λ) instead of ∧ to denote null string.
4) indicate nonterminals by writing them in angle brackets:
<S> ::= <X> | <Y>.

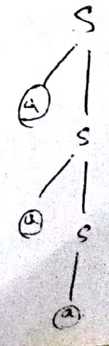
Ambiguity

A 'CFG' is called ambiguous if for atleast oneword in the language that it generates there are two possible derivations of the word that correspond to different syntax trees. If a CFG is not ambiguous, it is called unambiguous (unique p

eg. **Ambiguous** (multiple Paths)
$S \to aS \mid Sa \mid a$.
for the word a^3.



Unambiguous (unique path)
$S \to aS \mid a$ [same lang. Product
for the word a^3



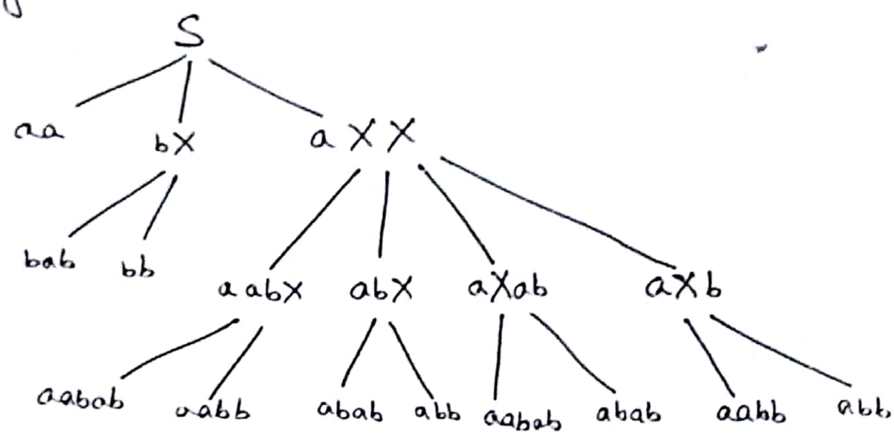
Total Language Tree

For a given CFG, we define a tree with a start symbol 'S' as its root and whose nodes are working strings of terminals and non-terminals. The descendants of each node are all the possible results of applying every applicable production to the working strings, one at a time. A string of all terminals is a terminal node in the tree.

eg.

$$S \rightarrow aa \mid bX \mid aXX$$
$$X \rightarrow ab \mid b$$

The total language tree is



exercise

i) What is the lang. CFG generates :-

$$S \rightarrow aX$$
$$X \rightarrow aX \mid bX \mid \Lambda$$

$a(a+b)^*$

2)

$$S \rightarrow XaXaX$$
$$X \rightarrow aX \mid bX \mid \Lambda$$

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

[atleast two 'a'].

Scanned by CamScanner

Semiword:- A semiword is a string of terminals (maybe none) concatenated with exactly one non-terminal (on the right). A semiword has the shape

(terminal) (terminal) .... (terminal) (Nonterminal)

(20)

Killing Λ - Productions

Language that has no Λ-productions but generates the same language.

eg 1.

$$S \to a \mid Xb \mid aYa \qquad\qquad S \to a \mid Xb \mid aYa \mid b \mid aa$$
$$X \to Y \mid \Lambda \qquad \Rightarrow \qquad X \to Y$$
$$Y \to b \mid X \qquad\qquad Y \to b \mid X$$

eg 2.

$$S \to Xa \qquad\qquad S \to Xa \mid a$$
$$X \to aX \mid bX \mid \Lambda \qquad \Rightarrow \qquad X \to aX \mid bX \mid a \mid b.$$

eg 3

$S \to XY$

$X \to Zb$

$Y \to bW$

$Z \to AB$

$W \to Z$

$A \to aA \mid bA \mid \Lambda$

$B \to Ba \mid Bb \mid \Lambda$

Nullables are:-
A
B
Z
W.

∴

$S \to XY$

$X \to Zb \mid b$

$Y \to bW \mid b$

$Z \to AB \mid A \mid B$

$W \to Z$

$A \to aA \mid bA \mid a \mid b$

$B \to Ba \mid Bb \mid a \mid b$

Scanned by CamScanner

Killing unit Productions

A production of the form

Nonterminal → one Nonterminal

is called a unit production.

eg. $S \to A \mid bb$ $\quad\Rightarrow\quad$ $S \to bb \mid b \mid a$.

$A \to B \mid b$ $\quad\quad$ $A \to b \mid a \mid bb$.

$B \to S \mid a$. $\quad\quad$ $B \to a \mid bb \mid b$.

Chomsky Normal Form (CNF)

If a CFG has only productions of the form

Nonterminal → string of exactly two Nonterminals

Nonterminal → one terminal.

eg. $S \to aSa \mid bSb \mid a \mid b \mid aa \mid bb$.

step-1. First eliminate null Productions

solⁿ. $S \to ASA \mid BSB \mid a \mid b \mid AA \mid BB$

$A \to a$

$B \to b$.

step-2. Then unit Productions.

step-3. Then convert to CNF using above Rule.

Now, introduce R's,

$S \to AR_1$

$R_1 \to SA$

$S \to BR_2$

$R_2 \to SB$

$S \to a \mid b \mid AA \mid BB$. [no changes]

which is required CNF.

eg. Convert CFG into CNF.

$S \to bA \mid aB$

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

$B \to aBB \mid bS \mid b$

Solⁿ.

$S \to YA \mid XB$

$A \to YAA \mid XS \mid a$

$B \to XBB \mid YS \mid b.$

$Y \to b$

$X \to a.$

Introduce R's.

$S \to YA \mid XB.$

$A \to YR_1 \mid XS \mid a, \quad R_1 \to AA$

$B \to XR_2 \mid YS \mid b, \quad R_2 \to BB.$

$Y \to b$

$X \to a$

eg. $S \to aaaaS \mid aaaa.$

Solⁿ

$S \to AR_1$

$R_1 \to AR_2$

$R_2 \to AR_3$

$R_3 \to AS$

$S \to AR_4$

$R_4 \to AR_5$

$R_5 \to AA$

$A \to a.$

eg.

$S \to XaX \mid bX$

$X \to XaX \mid XbX \mid c.$

$S \to XAX \mid BX$

$X \to XAX \mid XBX \mid c$

$A \to a, \ B \to b$

$S \to XR_1 \mid BX$

$X \to XR_1 \mid XR_2 \mid c$

$A \to a, \ B \to b$

$R_1 \to AX$

$R_2 \to BX$

Left most derivation.

eg If a word w is generated by a CFG by a certain derivation & at each step in the derivation, a rule of production is applied to the leftmost non-terminal in the working string, then this derivation is called a leftmost derivation.

eg. $S \rightarrow aSX \mid b$

$X \rightarrow Xb \mid a$

Soln. Leftmost derivation :-

$S \Rightarrow a\underline{S}X$

$\Rightarrow a\underline{aSX}X$

$\Rightarrow aa\underline{b}\underline{X}X$

$\Rightarrow aab\underline{Xb}X$

$\Rightarrow aab\underline{ab}X$

$\Rightarrow aababa.$

Scanned by CamScanner

Egs of CFG's.

1). even no. of a's (with atleast some a's)

A // ?? Exam IX ?? //

$S \rightarrow bS \mid aM$

$M \rightarrow bM \mid aF$

$F \rightarrow bF \mid aM \mid \Lambda$

OR. ~~$S \rightarrow XaaX$~~

~~$X \rightarrow bX \mid aaX \mid \Lambda$~~

2). $(aa+bb)^*$

$S \rightarrow aaS \mid bbS \mid \Lambda$

3). Even-Even.

$S \rightarrow aaS \mid bbS \mid abX \mid baX \mid \Lambda$

$X \rightarrow aaX \mid bbX \mid abS \mid baS.$

4). $(aa+bb)^+$

$S \rightarrow aA \mid bB$

$A \rightarrow aS \mid a$

$B \rightarrow bS \mid b$

OR. $S \rightarrow aaS \mid bbS \mid aa \mid bb.$

5). $(a+b)^* bb (a+b)^*$

$S \rightarrow XbbX$

$X \rightarrow aX \mid bX \mid \Lambda$

6). $aa^* bb^*$

$S \rightarrow XY$

$X \rightarrow XX$

$Y \rightarrow YY$

$X \rightarrow a$

OR

$S \rightarrow aXbY$

$X \rightarrow aX \mid \Lambda$

$Y \rightarrow bY \mid \Lambda$

OR

$S \rightarrow XY$

$X \rightarrow aX \mid a$

$Y \rightarrow bY \mid b$

Scanned by CamScanner

7). Palindrome (except Λ) [Non Null Palindrome].

$S \to aSa \mid bSb \mid a \mid b \mid aa \mid bb.$

8). a^{4n} for $n = 1, 2, 3, \ldots = \{a^4, a^8, a^{12} \ldots\ldots\}.$

$S \to AAAAS$

$S \to AAAA$

$A \to a.$

9). $L = \{a^n b^n a^m$ where $n, m = 1, 2, 3, \ldots$ & $n = m$ (not necessarily)

$S \to XA$

$X \to aXb \mid ab$

$A \to aA \mid a$

10). $L = \{a^n b^m a^m$ where $n, m = 1, 2, 3 \ldots$ & $n = m$ (not necessary)

$S \to AX$

$A \to aA \mid a$

$X \to bXa \mid ba$

Scanned by CamScanner

CFG's for the following languages:-

(57)

1) Lang = $a^n S$ where S starts with 'b' & length(s) = n

$S \to aSX$
$S \to ab$
$X \to a|b$

2) Lang = $a^m b^n c^p d^q$ where $m+n = p+q$

$S \to X|Y|\Lambda$
$X \to aXd|Y|\Lambda$
$X \to aAc$
$A \to aAc|Z|\Lambda$
$Y \to bYd|bBc|\Lambda$
$B \to bBc|\Lambda$
$Z \to bBc$

Q R.E. for strings that do not have 'ab' as substring.

$b^* a^*$

Q CFG for no three consecutive 'a's

$S \to aX|bS|\Lambda$
$X \to aY|bS|\Lambda$
$Y \to bS|\Lambda$

Q. CFG that do not have substring 'baa'

$S \to bX|aS|\Lambda$
$X \to aY|bS|\Lambda$
$Y \to bS|\Lambda$

$S \to aS|bX|\Lambda$
$X \to aY|bX|\Lambda$
$Y \to bX|\Lambda$

Scanned by CamScanner

Q. CFG for lang. $a^i b^j c^k$ where $i+j=k$, & $i, j, k \geq 0$

$S \rightarrow X \mid Y \mid \wedge$

$X \rightarrow aXc \mid Y \mid \wedge$

$Y \rightarrow bYc \mid \wedge$

Q. lang. that contain exactly two or three b's.

$S \rightarrow ABA$

$A \rightarrow aA \mid \wedge$

$B \rightarrow bb \mid bbb$

R.E. $a^*(bb+bbb)a^*$

Scanned by CamScanner

① CFG for $(aaa + b)^*$

$S \to X\,aaa\,X \mid Y \mid \Lambda$

$X \to aaa\,X \mid Y \mid \Lambda$

$Y \to bY \mid X \mid \Lambda$

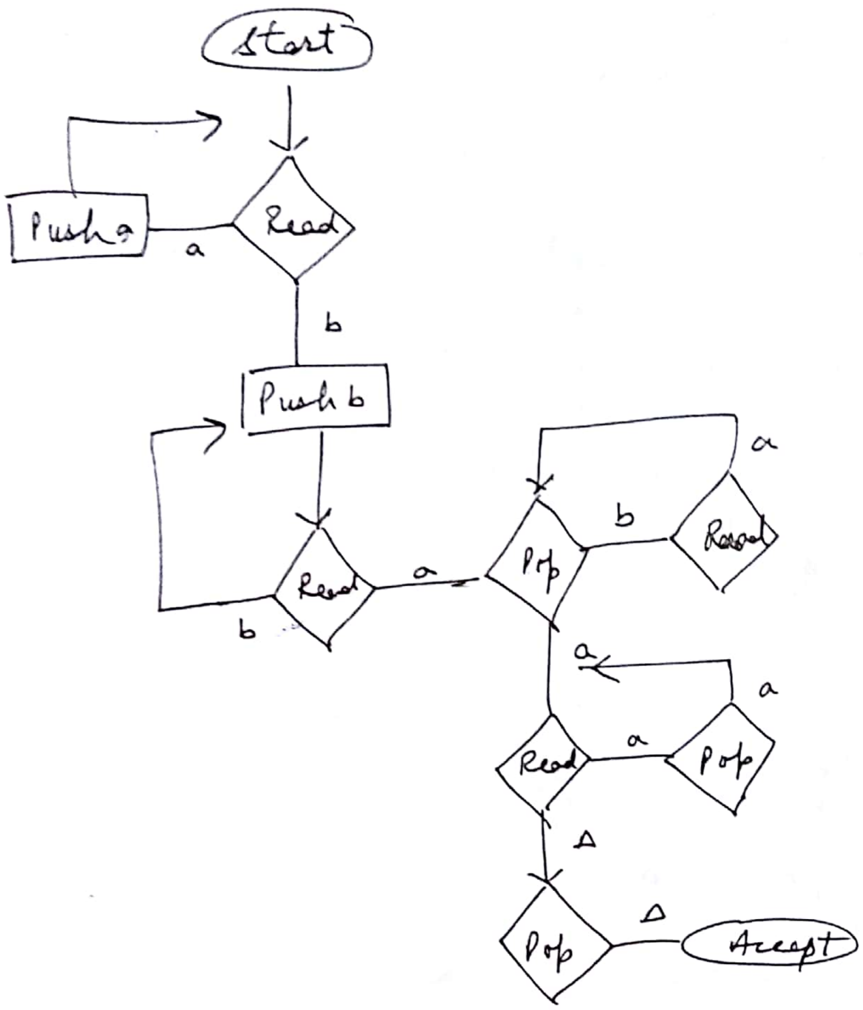
$aaaS \mid bS \mid \Lambda$

(Include Λ in all three rules).

Scanned by CamScanner

PDA for

$a^m b^n a^{m+n}$, $n \geq 1$, $m \geq 1$



$m, n \geq 1$

$a^m b^n$, $m \neq n$

$S \to aAb \mid aBb$

$A \to aAb \mid aA \mid a$

$B \to aBb \mid bB \mid b$

<u>CFG for Non-Palindrome</u>

$\cancel{S \to A \mid B \mid \Lambda}$

$S \to aSa \mid aCb \mid bSb \mid bCa$

$C \to aC \mid bC \mid \Lambda$

$\cancel{B \to bBb \mid bCa \mid ba \mid ab \mid aBa}$

$a^m b^n$ aabbb $m \neq n$, $m, n \geq 1$

$\cancel{S \to aSb \mid B \mid C}$

$\cancel{B \to bB \mid b}$

$\cancel{C \to aC \mid a}$

$m > n$

$S \to aBb$

$B \to aBb \mid aB \mid a$

$a^m b^n$, $m > n$

$\cancel{S \to aSb \mid B}$

$\cancel{B \to aB \mid a}$ $m > n$

$S \to aBb$

$B \to aBb \mid aB \mid a$

Derivation Tree (Parse Trees)

- used to represent derivations.

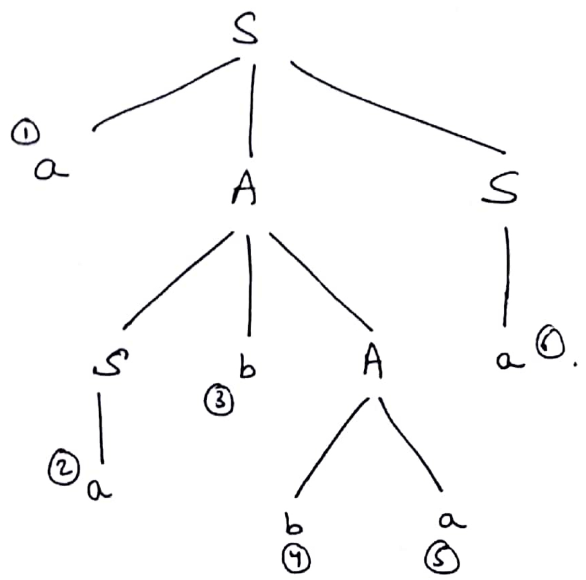
Q. Consider CFG whose Productions are

$S \rightarrow aAS \mid a$

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

Show that $S \overset{*}{\Rightarrow} aabbaa$ & construct derivation Tree whose yield is 'aabbaa'.

$\overset{1}{a}\ \overset{2}{a}\ \overset{3}{b}\ \overset{4}{b}\ \overset{5}{a}\ \overset{6}{a}.$



$S \Rightarrow a\underline{A}S$

$\Rightarrow a\underline{S}bAS$

$\Rightarrow aab\underline{A}S$

$\Rightarrow aabba\underline{S}$

$\Rightarrow aabbaa.$

Leftmost derivation.

A derivation $A \overset{*}{\Rightarrow} w$ is called a leftmost derivation if we apply a production only to the leftmost variable at every step.

Rightmost derivation

A derivation $A \overset{*}{\Rightarrow} w$ is called a rightmost derivation if we apply production to the rightmost variable at every step.

eg. Let CFG be $S \rightarrow 0B \mid 1A$

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

$B \rightarrow 1 \mid 1S \mid 0BB$.

for the string '00110101' find (a) Leftmost (b) Rightmost derivations.

(a) Leftmost derivation :-

$S \Rightarrow 0\underline{B}$

$\Rightarrow 00\underline{B}B$

$\Rightarrow 001\underline{B}$

$\Rightarrow 0011\underline{S}$

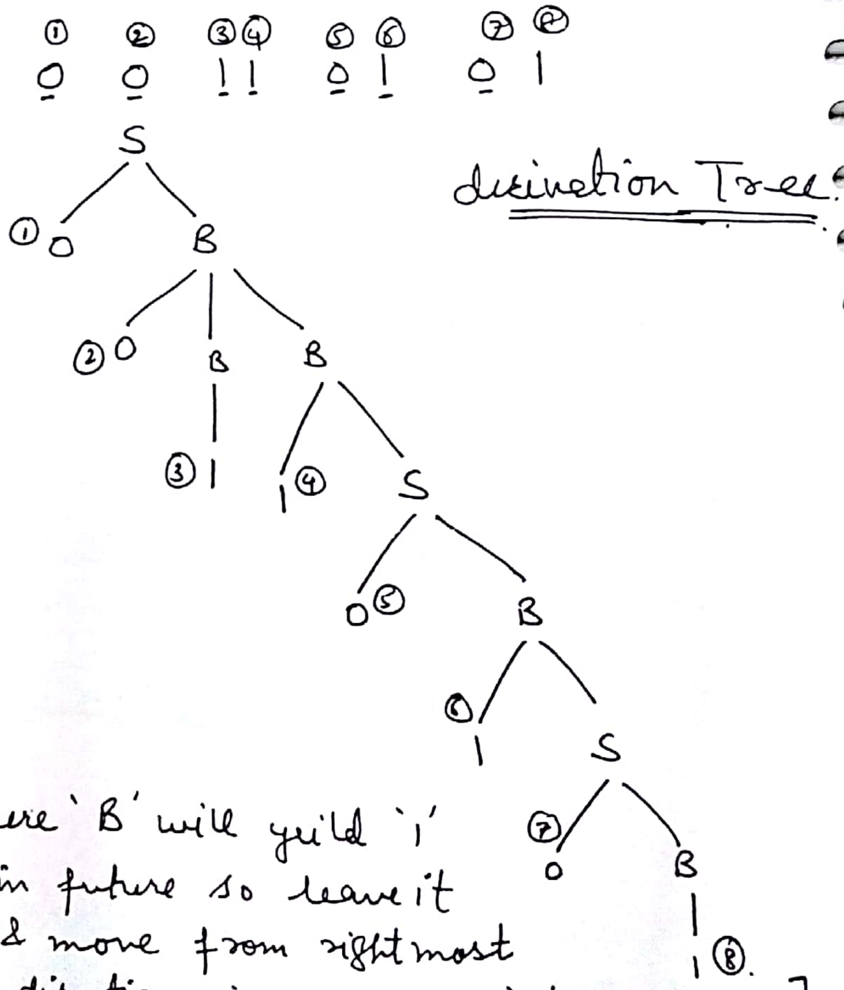
$\Rightarrow 00110\underline{B}$

$\Rightarrow 001101\underline{S}$

$\Rightarrow 0011010\underline{B}$

$\Rightarrow 00110101$

$\Rightarrow$



derivation Tree.

(b) Rightmost derivation.

$S \Rightarrow 0\underline{B}$

$S \Rightarrow 0\,0B\underline{B}$

$\Rightarrow 00B\,1\underline{S}$ ← [Imp.] [Here 'B' will yeild '1' in future so leave it & move from rightmost direction in search of '0' ie no. ⑤.]

$\Rightarrow 00B1\,0\underline{B}$

$\Rightarrow 00B10\,1\underline{S}$

$\Rightarrow 00B101\,0\underline{B}$

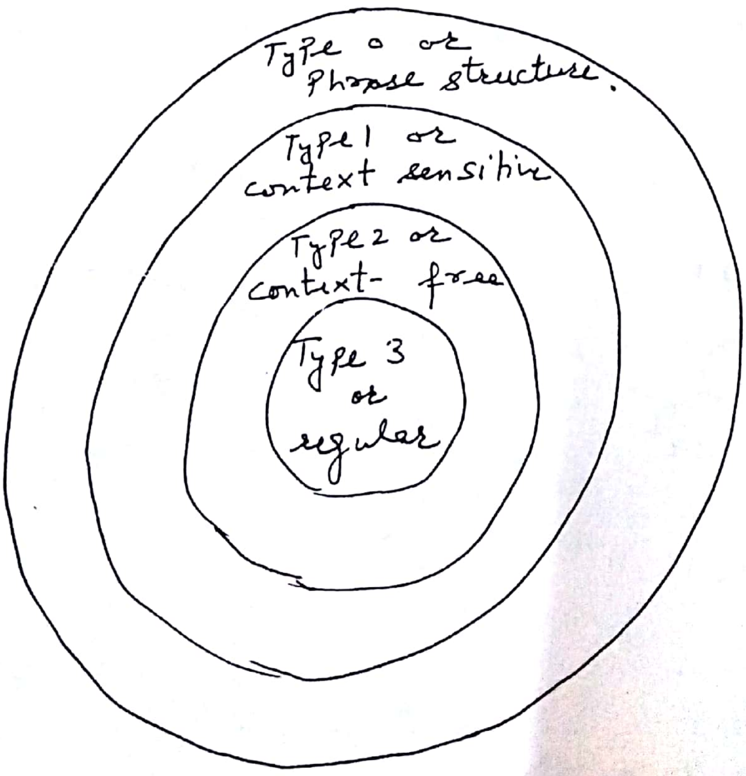
$\Rightarrow 00\,\underline{B}10101$

$\Rightarrow 00110101$

Scanned by CamScanner

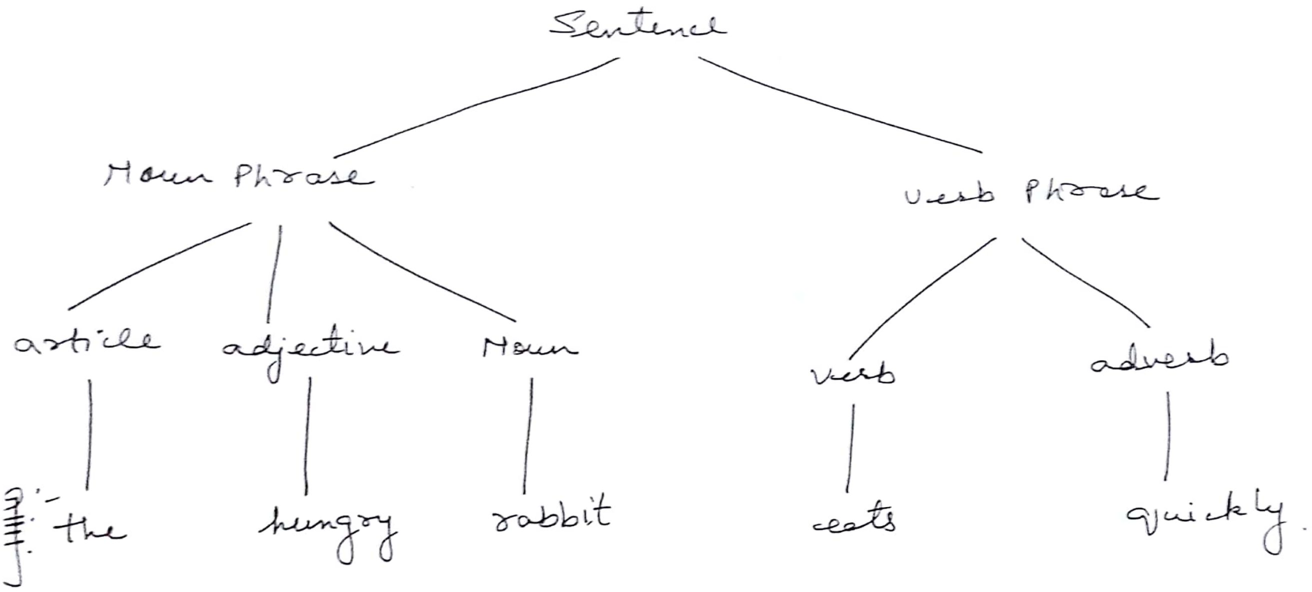
Types of grammars.

Type	grammar	Rules. $(X \rightarrow Y)$.
0	Phrase structure. [Acceptor - TM]	⇒ No restriction on Productions. X = Any string with NT and Terminals (atleast one NT) Y = any string.
1	context sensitive [TM with bounded (not infinite) Tape called LBA]	⇒ of the form $w_1 \rightarrow w_2$ where $\text{length}(w_2) \geq \text{length}(w_1)$ OR $w_1 \rightarrow \Lambda$ and w_1 is any string with NT.
2	context free [Acceptor - PDA]	⇒ of the form $w_1 \rightarrow w_2$ where w_1 is a single Nonterminal symbol. OR $w_2 = \Lambda$ and w_2 is any string.
3	Regular grammar. [Acceptor - FA]	⇒ of the form $w_1 \Rightarrow w_2$ where w_1 = Single NonTerminal, w_2 = Terminal, OR w_2 = Terminal NonTerminal OR $w_2 = \Lambda$



Scanned by CamScanner

Phrase structure.



Scanned by CamScanner