Unit-3

CFL and CFG

Page No. ______
Date

Context free Language / Context Free Grammer

In a formal language theory, context free grammer is a set of recursive rewriting rules used to generate pattern of strings.

Mathematically, Context free grammer is defined by 4 tuples.

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

V stands for set of variables or non-terminal symbols

Σ/T stands for set of terminal symbols.

S stands for start symbol &

P stands for set of production.

CFG has production rule of the form $A \rightarrow \alpha$ where α belongs to $\{V \cup \Sigma\}^*$ and $A \in V_N$.

Language generated by CFG is termed as Context free language and machine used for this purpose is PDA (Push down automata).

Q. For generating a language, equals no. of a's and b's in the form $a^n b^n$ the context grammer will be defined as $G = \{(S, A), (a, b), S, S \rightarrow aAb, A \rightarrow aAb/\epsilon\}$

Sol. $S \rightarrow aAb$

$S \rightarrow aaAbb$ $\quad\quad A \rightarrow aAb$

$S \rightarrow aaaAbbb$

$S \rightarrow aaaaAbbbb$

$S \rightarrow a^n b^n$

Terminals $\rightarrow$ Always in capital

Non-terminal $\rightarrow$ Always in small.

Page No.
Date

Derivation:

A derivation is a sequence of tokens that is used to find out whether a sequence of string is generating valid statement or not.

Getting a string from the given grammer is called derivation and tree created from that derivation is known as derivation tree.

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

$L = \{a, aa, aaa, aaaa, ----\}$

$A \rightarrow aA / a$

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

$L = \{\epsilon, a, aa, aaa, ...\}$

$A \rightarrow aA / \epsilon$

Q. $L = \{(a+b)^*\}$

$L = \{\epsilon, a, b, aa, ab, ba, bb, ---\}$

$S \rightarrow aS / bS / \epsilon$

Left most derivation:

The derivation $S \rightarrow s$ is called a left most derivation, if the production is applied only to the left most variable at every step.

Right most derivation:

A derivation $S \rightarrow s$ is called a right most derivation if production is applied only to the right most variable at every step.

Page No.
Date

Q. Set of all strings over alphabet a, b : $(a+b)(a+b)(a+b)^*$

Sol. $S \rightarrow AAB$

$A \rightarrow a/b$

$B \rightarrow aB / bB / \epsilon$

Q. Set of all strings that starts with a and end with b.

Sol. $S \rightarrow aAb$

$A \rightarrow aA / bA / \epsilon$

Q. Set of all strings that start and end with diff. symbol.

Sol. $S \rightarrow aAb / bAa$

$A \rightarrow aA / bA / \epsilon$

Q. Set of all strings that starts and ends with same symbol.

Sol. $S \rightarrow aAa / bAb$

$A \rightarrow aA / bA / \epsilon$

Q. Set of all even length string.

Sol. $S \rightarrow AAS / \epsilon$

$A \rightarrow a/b$

Q. Set of all strings of length atmost 2.

$S \rightarrow AA$

$A \rightarrow a / b / \epsilon$

Q. $L = \{a^n b^n, n \geq 1\}$

$S \rightarrow aSb / ab$ $V_N = \{S\}$ $V_t = \{a, b\}$

Q. $a^n b^n c^m, n, m \geq 1$

$S \rightarrow AB$

$A \rightarrow aAb / ab$

$B \rightarrow cB / \epsilon$

Scanned by CamScanner

Q. $a^n a^m b^n$, $n,m \geq 1$

Sol. $S \rightarrow aSb / aBb$

$B \rightarrow cB / \epsilon$

Q. Set of all palindrome

$ww^R \cup waw^R \cup wbw^R$, $w \in (a,b)^*$

Sol. $S \rightarrow aSa / bSb / a / b / \epsilon$

Q. $RE = \{(011+1)^* (01)^*\}$

Sol. $S \rightarrow AB$

$A \rightarrow 011A / 1A / \epsilon$

$B \rightarrow 01B / \epsilon$

Q. $L = \{a^n b^{2n} c^m \mid n,m >= 0\}$

Sol. $S \rightarrow AB$

$A \rightarrow aAbb / \epsilon$ $\qquad V_N = \{S, A, B\}$ $\qquad V_t = \{a, b\}$

$B \rightarrow cB / \epsilon$

Q. Following grammar generate languages of regular expression: $0^*1(0+1)^*$, $S \rightarrow A1B$, $A \rightarrow 0A / \epsilon$, $B \rightarrow 0B / 1B / \epsilon$.
Give left most and right most derivation of strings 00101.

Sol.

Left most derivation:	Right most derivation:
$S \rightarrow A1B$	$S \rightarrow A1B$
$S \rightarrow 0A1B$	$S \rightarrow A10B$
$S \rightarrow 00A1B$	$S \rightarrow A101B$
$S \rightarrow 00\epsilon 1B$	$S \rightarrow A101\epsilon$
$S \rightarrow 001B$	$S \rightarrow A101$
$S \rightarrow 0010B$	$S \rightarrow 0A101$
$S \rightarrow 00101B$	$S \rightarrow 00A101$
$S \rightarrow 00101\epsilon$	$S \rightarrow 00\epsilon 101$
$S \rightarrow 00101$	$S \rightarrow 00101$

Q. $L = \{a^n b^n c^m d^m / n,m \geq 1\}$

Sol. $Q \rightarrow SP$

$S \rightarrow aSb / ab$

$P \rightarrow cPd / cd$

Q. $L = \{a^n b^n c^n / n \geq 1\}$ (Ambiguity)

Q. $L = \{a^{m+n} b^m c^n / m,n \geq 1\}$

$L = a^n . a^m b^m c^n$

$A \rightarrow aAc / ac$ $\qquad V_N = \{A\}$ $\qquad V_t = \{a, b, c\}$

$A \rightarrow aAb / ab$

Q. $L = \{a^n b^{2n} / n \geq 1\}$

$S \rightarrow aSbb / abb$ $\qquad V_N = \{S\}$ $\qquad V_t = \{a, b\}$

Q. $a^n b^m c^{n+m} / n,m \geq 1$.

$L = a^n b^m c^m c^n$

$A \rightarrow aAc$ $\qquad V_N = \{A\}$ $\qquad V_t = \{a, b, c\}$

$A \rightarrow bAc / bc$

Sentential forms:

Consider $G = (V, T, P, S)$ be a CFG, then we can derive a string w from it. This w can obtained from $(V \cup T)^*$ where V denotes the set of non-terminal symbols and T denotes the set of terminal symbols. The derivation of w from start symbol S can be written as $S \overset{*}{\Rightarrow} w$ which is called as sentential form.

If $S \underset{lm}{\overset{*}{\Rightarrow}} w$ then w is a left-sentential form.

If $S \Rightarrow w$ then w is a right-sentential form

Page No. ____
Date

Q. Construct a CFG for the language $L = \{a^m b^n / m \neq n\}$

Sol. $S \rightarrow aSb / A / B$

$A \rightarrow aA / a$

$B \rightarrow bB / b$

Q. Build a CFG for the language $L = \{0^i 1^j 2^k \mid j > i + k\}$

Sol. $L = 0^i 1^i 1^j 1^k 2^k$

$S \rightarrow ABC$

$A \rightarrow 0A1 / \epsilon$

$B \rightarrow 1B / 1$

$C \rightarrow 1C2 / \epsilon$

Q. Find the regular grammer for the language $L = \{a^n b^m \mid n + m \text{ is even}\}$

Sol. $R.E = [(aa)^* (bb)^*] + [a(aa)^* b(bb)^*]$

$S \rightarrow AB / XY$

$A \rightarrow aaA / aa$ $\quad X \rightarrow aaX / a$

$B \rightarrow bbB / bb$ $\quad Y \rightarrow bbY / b$

Derivation Trees

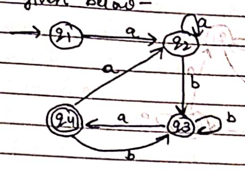
Derivation trees is a graphical representation for the derivation of the given production rules for a given CFG. It is the simple way to show how the derivation can be done to obtain some string from given set of production rules. The derivation tree is also called parse tree.

1. The root node is always a node indicating start symbol.
2. The derivation is read from left to right.
3. The leaf nodes are always terminal nodes.
4. The interior nodes are always the non terminal nodes.

Page No. ____
Date

Numerical Problem Based on Arden's Theorem:

Q. Find the regular expression using Arden's Theorem of FA given below–



Sol. $q_1 = \epsilon$ — (1)

$q_2 = q_1 a + q_2 a + q_4 a$ — (2)

$q_3 = q_2 b + q_3 b + q_4 b$ — (3)

$q_4 = q_3 a.$ — (4).

Put eqn (1) in (2), we get,

$q_2 = a + q_2 a + q_4 a$

$q_2 = (q_4 + \epsilon) a a^*$

Put eqn (4) in (3), we get,

$q_3 = q_2 b + q_3 b + q_3 ab$

$q_3 = q_2 b + q_3 (b + ab)$

$q_3 = q_2 b (b + ab)^*$

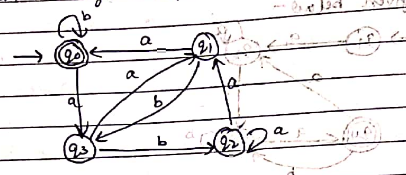
Now from eq. (4),

$q_4 = [q_2 b (b + ab)^*] a$

$q_4 = \{[(q_4 + \epsilon) a a^*][b (b + ab)^* a]\}$

Page No.
Date

Q. obtain the regular expression corresponding to the following automata:



Sol.
$q_0 = q_0 b + q_1 a + \epsilon$ —— (1)
$q_1 = q_3 a + q_2 a$ —— (2)
$q_2 = q_2 a + q_3 b$ —— (3)
$q_3 = q_1 b + q_0 a$ —— (4)

from eqn (3)
$q_2 = q_3 b a^*$
Put eqn (4) in this
$q_2 = (q_1 b + q_0 a) b a^*$ —— (5)

from eq.(2)
$q_1 = q_3 a + q_2 a$
Put q_3 from eqn (4) & q_2 from eq.(5)

So, $q_1 = (q_1 b + q_0 a)\, a + (q_1 b + q_0 a) b a^* a$

$q_1 = (q_1 b + q_0 a)(a + b a^*)$
$q_1 = q_0 a (a + ba^*) + q_1 b (a + ba^*)$
$q_1 = q_0 a (a+ba^*) [b(a+ba^*)]^*$

from eq.(1)
$q_0 = q_0 b + q_1 a + \epsilon$
$q_0 = q_0 b + q_0 a (a+ba^*) [b(a+ba^*)]^* a + \epsilon$

$q_0 = [b + a(a+ba^*)(b(a+ba^*))^* a]^*$

Page No.
Date

Q. Design CFG for $\Sigma = \{a, b\}$ that generates the set of
(i) all strings with exactly one 'a'.
(ii) all strings with atleast one 'a'
(iii) all strings with atleast 3a's.

Sol. i) P: $S \to AaA$
$A \to bA / \epsilon$
$V_N = \{S, A\}$ $V_t = \{a, b\}$

ii) P: $S \to AaA$
$A \to aA / bA / \epsilon$
$V_N = \{S, A\}$ $V_t = \{a, b\}$

iii) P: $S \to AaAaAaA$
$A \to aA / bA / \epsilon$
$V_N = \{S, A\}$ $V_t = \{a, b\}$

Q. Construct CFG for the language containing all the strings of different first and last symbol.

Sol. P: $S \to aAb / bAa$
$A \to aA / bA / \epsilon$
$V_N = \{S, A\}$ $V_t = \{a, b\}$

Q. Write CFG for the $L = \{a^n b^n c^m d^m / n \geq 1, m \geq 0\}$

Sol. P: $S \to AB$
$A \to aAb / ab$
$B \to cBd / \epsilon$
$V_N = \{S, A, B\}$ $V_t = \{a, b, c, d\}$

Q. find the CFG for the RE : $(110+11)^* (10)^*$

Sol. P: $S \to AB$
$A \to 110A / 11A / \epsilon$
$B = 10B / \epsilon$
$V_N = \{S, A, B\}$ $V_t = \{0, 1\}$

Page No. ______
Date

Q. Write CFG for any combination of a's and b's ?

Sol. P: $S \rightarrow aS / bS / \epsilon$

$V_N = \{S\}$ $V_t = \{a, b\}$

Q. Write a CFG for the language in which 2nd symbol is 0 & 4th symbol is 1 over $\Sigma = \{0, 1\}$.

Sol. P: $S \rightarrow A0A1B$

$A \rightarrow 0/1$

$B \rightarrow 0B / 1B / \epsilon$

$V_N = \{S, A, B\}$ $V_t = \{0, 1\}$

Q. Derive the string 'aabbabba' with left most derivation and right most derivation using a CFG given by

$S \rightarrow aB / bA$

$A \rightarrow a / aS / bAA$

$B \rightarrow b / bS / aBB$

Sol.

Left most Derivation	Right most Derivation
$S \rightarrow aB$	$S \rightarrow aB$
$S \rightarrow aa\underline{B}B$	$S \rightarrow aaB\underline{B}$
$S \rightarrow aab\underline{B}$	$S \rightarrow aaBbS$
$S \rightarrow aabb\underline{S}$	$S \rightarrow aaBbbA$
$S \rightarrow aabba\underline{B}$	$S \rightarrow aaBbba$
$S \rightarrow aabbab\underline{S}$	$S \rightarrow aabSbba$
$S \rightarrow aabbabb\underline{A}$	$S \rightarrow aabbAbba$
$S \rightarrow aabbabba$	$S \rightarrow aabbabba$

C

Page No. ______
Date

Q. Derive the string '1000111'

$S \rightarrow T00T$

$T \rightarrow 0T / 1T / \epsilon$ for LMD & RMD using CFG.

Sol.

Left Most Derivation	Right most Derivation
$S \rightarrow T00T$	$S \rightarrow T00T$
$S \rightarrow 1T00T$	$S \rightarrow T001T$
$S \rightarrow 10T00T$	$S \rightarrow T0011T$
$S \rightarrow 1000T$	$S \rightarrow T00111T$
$S \rightarrow 10001T$	$S \rightarrow T00111$
$S \rightarrow 100011T$	$S \rightarrow 1T00111$
$S \rightarrow 1000111T$	$S \rightarrow 10T00111$
$S \rightarrow 1000111$	$S \rightarrow 1000111$

Parse Tree or Derivation Tree

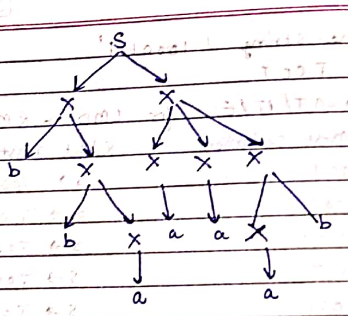
Let $G = \{V_N, V_t, P, S\}$ be a context free grammar. An ordered tree for this CFG G is a derivation tree if and only if, it has the following properties:

1. The root is labelled by the starting non-terminal of the CFG i.e. S.
2. Every leaf of the ordered tree has a label from $V_t \cup \epsilon$.
3. Every interior node of ordered tree has a label from V_N.

Q. Consider the CFG $S \rightarrow XX$

$X \rightarrow XXX / bX / Xb / a$. Find a parse tree for the string bbaaaab?

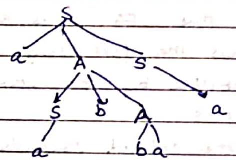
Sol



Anticlock wise . read

Q. Consider the CFG whose productions are
$S \rightarrow aAS/a$
$A \rightarrow SbA/SS/ba$
construct a derivation tree for aabbaa.

Sol

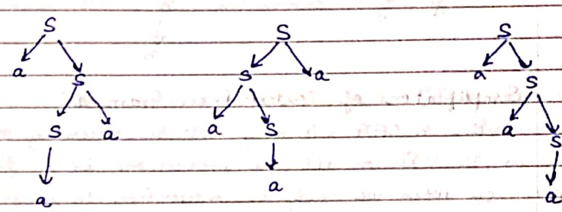


Ambiguity in Grammar & Language:

A CFG is called ambiguous if for at least one word in the language that it generates, there are two or more possible derivations of the word that corresponds to different syntax trees. If a CFG is not ambiguous, it is called unambiguous.

Q. The language of all non null strings of a's can be defined by a CFG as follows:
$S \rightarrow aS/Sa/a$
Check is it ambiguous grammar?

Sol Let $w \rightarrow aaa$

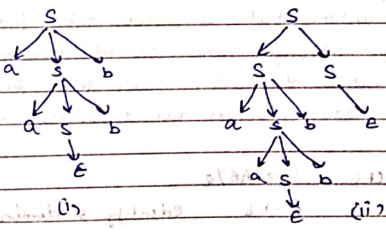


(i) (ii) (iii)

Here $w = aaa$, we can make more than ~~two~~ one parse tree hence is called ambiguous grammar.

Q. The CFG $S \rightarrow aSb/SS/\epsilon$ is ambiguous it for $w = aabb$.

Sol



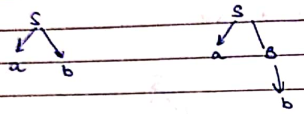
(i) (ii)

Here, we can more than one tree parse tree, hence is called ambiguous grammar.

Page No.
Date

Q. Show that the grammar $S \rightarrow aB/ab$
$A \rightarrow aAB/a$
$B \rightarrow ABb/b$ is ambiguous?

Sol. let w = ab



Simplification of Context Free Grammar:
In a CFG, it may not be necessary to use all symbols in $V_N \cup \Sigma$ or all the productions in P for deriving sentences. So when we study a context free language L(G), we try to eliminate those symbols and productions in G which are not useful for the derivation of sentences.
1) Removal of useless production
2) Removal of unit production
3) Removal of null production.

Removal of Useless Production
The production that can never take part & a derivation of any strength are called useless production. Similarly, a variable that can never take part & derivation of any strength is called useless variable.

Q. Consider a CFG $S \rightarrow AB/a$
$A \rightarrow b$. Identify & eliminate useless symbol.

Sol. $S \rightarrow AB$
In this production, when we put the value of A, then the production will bB, that will not give any result in the future because we don't have any value of B. Hence this production is useless.

$S \rightarrow a$
give the result it means this grammar accept the string a.
$A \rightarrow b$ doesn't have any existence in starting symbol S.
finally, the reduced grammar will be
$\boxed{S \rightarrow a}$

Page No.
Date

Q. Consider the following grammar & obtain equivalent grammar containing no useless grammar symbol.
$A \rightarrow xyz / Xyzz$
$X \rightarrow Xz / xYz$
$Y \rightarrow yYy / Xz$
$Z \rightarrow ZY/z$

Sol. $Z \rightarrow ZY/z$
is useless, it doesn't have any existence in derivation.
X, Y production are non-terminating production
So, $A \rightarrow xyz$

Q. Find a reduced grammar i.e equivalent to CFG given that: $S \rightarrow aC / SB$
$A \rightarrow bSCa$
$B \rightarrow aSB / bBC$
$C \rightarrow aBc / ad$

Sol. $S \rightarrow aC$
$C \rightarrow ad.$

Scanned by CamScanner

Page No.
Date

Removal of Unit Production

The production of type $A \to B$ is called unit production where A, B belongs to non-terminal.

Q. Consider the context free grammar G

$S \to AB$
$A \to a$
$B \to C/b$
$C \to D$
$D \to E$
$E \to a$

Sol. There are three unit production

$B \to C$
$C \to D$
$D \to E$
$E \to a$

So,

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

So, $\boxed{\begin{array}{l} S \to aB \\ B \to a/b \end{array}}$

Q. Consider the following ambiguous expression grammar.

$S \to A/bb$
$A \to B/b$
$B \to S/a$

Sol. $S \to bb/b/a$

Page No.
Date

Removal of Null production:

If a CFG, non-terminal variable or symbol A is nullable variable. If there is a production $A \to \epsilon$ or there is a derivation that starts at A and leads to ϵ.

Step 1: Find out all null productions & then find out all nullable variables (If it directly produces $A \to \epsilon$ or after some steps, it generates null then that variable is called nullable variable.

Step 2: After finding nullable variable, go to R.H.S of every variable and wherever that variable is present, write that with it or without it, which means that with A or without A and then eliminate $A \to \epsilon$.

Q. $S \to aSb / aAb / \epsilon$

Sol. $S \to ab / aSb / aAb$

So,

$S \to ab / aSb$

Q. Consider the following grammar

$S \to aA$
$A \to b/\epsilon$

Sol. $S \to aA/a$
$A \to b$

Q. Consider the following grammar:

$S \to ABAC$
$A \to aA/\epsilon$
$B \to bB/\epsilon$
$C \to c$

Sol. $S \to ABAC / BAC / AAC / ABC / AC / BC / C$
$A \to aA/a$
$B \to bB/b$
$C \to c$

Scanned by CamScanner

Q. Consider the following grammar & remove null productions:

(i) S → aSa
S → bSb / ε

(ii) S → a / Xb / aYa
X → Y / ε
Y → b / ε

Sol. (i) S → aSa / aa
S → bSb / bb

(ii) S → a / Xb / aYa / b / aa
X → b
Y → b

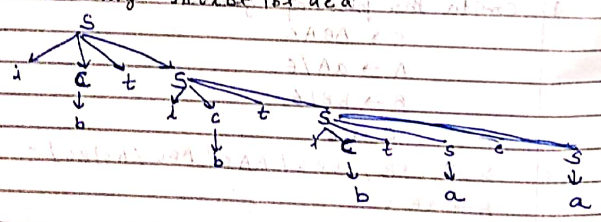
Q. Design a CFG for RE = (a+b)* bb (a+b)* which is free from null production.

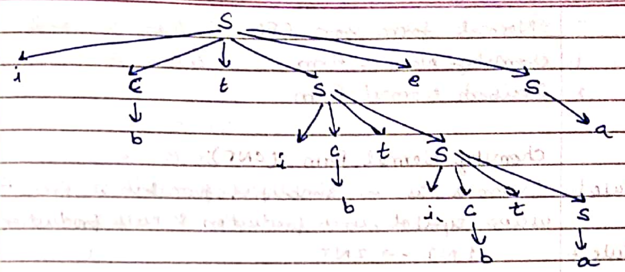
Sol. S → AbbA
A → aA / bA / ε
By removing null production.
S → AbbA / bbA / Abb / bb
A → aA / bA / a / b

Q. Check whether the given grammar is ambiguous or not? S → iCtS
S → iCtSeS
S → a
C → b

For string 'ibtibt ibt aea'

Sol.





Hence, we can more than one derivation tree, so given grammar is ambiguous.

Q. Eliminate the useless symbol from the grammar.
S → aA / a / Bb / cC
A → aB
B → a / Aa
C → cCD
D → ddd

Sol. Here C is non-terminating. So,
S → aA / a / Bb
A → aB
B → a / Aa

Page No.
Date

Normal form for CFG:

1. Chomsky Normal Form
2. Greibach Normal Form

Chomsky Normal Form (CNF):

Rule 1: Grammar is in simplified form & it is free from useless symbol, unit production & null production.

Rule 2: 1 NT → 2 NT

1 NT → 1 T

(non-terminal) (terminal)

Steps for converting CFG to CNF:

1) Eliminate start symbol from right hand side.
If start symbol S is at R.H.S. of any production in grammar, create a new production
$S_0 \rightarrow S$
2) Eliminate null production, useless production & unit production.
3) Eliminate terminals from R.H.S., if they exist with other terminals or non-terminals.
4) Eliminate R.H.S. with more than 2 non-terminals.

Q. Change the following grammar into CNF

$S \rightarrow bA / aB$

$A \rightarrow bAA / aS / a$

$B \rightarrow aBB / bS / a$

Sol. Let $X \rightarrow a$

$Y \rightarrow b$

Now $S \rightarrow YA / XB$

$A \rightarrow YAA / XS / a$

$B \rightarrow XBB / YS / a$

Page No.
Date

Let $M \rightarrow AA$

$N \rightarrow BB$

So, $S \rightarrow YA / XB$

$A \rightarrow YM / XS / a$

$B \rightarrow XN / YS / a$

$X \rightarrow a$

$Y \rightarrow b$

$M \rightarrow AA$

$N \rightarrow BB$

Q. Change the grammar into CNF

$S \rightarrow abSb / a / aAb$

$A \rightarrow bS / aAAb$

Sol. $X \rightarrow a$

$Y \rightarrow b$

Now,

$S \rightarrow XYSY / a / XAY$

$A \rightarrow YS / XAAY$

Let $M \rightarrow XA$

$S \rightarrow XYSY / a / MY$

$A \rightarrow YS / MAY$

Let $U \rightarrow XY$

$S \rightarrow USY / a / MY$

$A \rightarrow YS / MAY$

Let ~~MBD~~ $N \rightarrow MA$ & $T \rightarrow SY$

$S \rightarrow UT / a / MY$

$A \rightarrow YS / NY$

Q. Design a CFG for the $L = \{ a^{4n} / n \geq 1 \}$. Convert it in CNF form.

Sol. $S \to aaaaS / aaaa$

Let $X \to a$

$S \to XXXXS / XXXX$

Let $Y \to XX$

$S \to YYS / YY$

Let $Z \to YY$

So, $S \to ZS / YY$

$Y \to XX$

$X \to a$

Q. Change the following grammar into CNF

$S \to aAD$

$A \to aB / bAB$

$B \to b$

$D \to d$

Sol. Let $X \to a$

So $S \to XAD$

$A \to XB / BAB$

$B \to b$

$D \to d$

Let $Y \to AD$ & $Z \to AB$

$X \to a$

$S \to XY$

$A \to XB / BZ$

$B \to b$

$D \to d$

Q. Change the following grammar into CNF:

$S \to aAbB$

$A \to aA / a$

$B \to bB / b$

Sol. Let $X \to a$

$Y \to b$

So, $S \to XAYB$

$A \to XA / a$

$B \to YB / b$

Let $P \to XA$

$Q \to YB$

So,

$S \to PQ$

$A \to XA / a$

$B \to YB / b$

$X \to a$

$Y \to b$.

Greibach Normal form (GNF)

Rule 1: Grammar is in simplified form:
Rule 2: 1NT → 1T (NT)*

Eg: S → aA ✓
A → bBC ✓
B → b ✓
C → aBCa ✗

Q. Convert the grammar into GNF.
S → AB/BC
A → aB/bA/a ✓
B → bB/cC/b ✓
C → c ✓

Sol. In starting symbol S, put value of A
S → AB
S → BC
A → aB/bA/a
B → bB/cC/b
C → c

Take S → AB
Now, S → aBB / bAB / aB

Take S → BC
Now S → bBC / cCC / bC

Now,
S → aBB / bAB / aB / bBC / cCC / bC
A → aB / bA / a
B → bB / cC / b
C → c

Q. Convert a grammar into CFG:
S → abaSa / aba

Sol. Let X → a
Y → b
So, S → aYXSX / aYX
X → a
Y → b.

Q. Convert the following grammar into GNF:
S → aAS
S → a
A → SbA
A → SS
A → ba

Sol. Let X → a & Y → b
So, S → aAS / a
A → SbA
Put value of S
A → SYA
A → aASYA / aYA

A → SS
A → aASS / aS
A → bX

Lemma 1: If $G = (V_N, \Sigma, P, S)$ be a CFG and if

$A \to Aa$

& $A \to \beta_1 / \beta_2 / \dots / \beta_n$ belongs to production rules (P) of G

Then a new grammar

$G' = (V_N, \Sigma, P', S)$ can be constructed by replacing

$A \to \beta_1 / \beta_2 / \dots \beta_n$ in $A \to Aa$ which will produce

$A \to \beta_1 a / \beta_2 a \dots / \beta_n a$

Lemma 2: Let $G = (V_N, \Sigma, P, S)$ be a CFG and the set of A productions belongs to P be

$A \to A\alpha_1 / A\alpha_2 \dots / A\alpha_m$

$A \to \beta_1 / \beta_2 / \dots / \beta_n$.

Introduce a new non-terminal X.

$A \to \beta_i / \beta_i X$

$X \to \alpha_j / \alpha_j X$.

Scanned by CamScanner

Closure properties of CFL:

The Context free language are closed under some operations which means that after performing that particulars operation on those CFLs, the resultant language is CFL.

1. CFL are closed under union.
2. CFL are closed under concatenation
3. CFL are closed under kleen closure
4. CFL are not closed under complement
5. CFL are not closed under intersection.

Decision properties of CFL:

1) Emptiness:

There exists an algorithm which can determine whether or not the given CFG can generate any word at all.

Proof: The algorithm to determine whether or not a grammar generate any word is

1) Put the dot above every terminal symbol. if there is any null string, then put the dot over null as well. These symbol occur at the right hand side of production rule.

2) If all the symbol of R.H.S of production rule are doted, then dot the corresponding non-terminal which is at the L.H.S of the production throughout the grammar. Repeat this step as long as possible.

3) If the starting symbol get doted, then the ~~answe~~ answer is 'yes' otherwise 'No'.

Ex: $S \rightarrow PQ$
$P \rightarrow AP / AA$
$A \rightarrow a$
$Q \rightarrow BQ / BB$
$B \rightarrow b$

Step 1: $S \rightarrow PQ$
$P \rightarrow AP / AA$
$A \rightarrow \dot{a}$
$Q \rightarrow BQ / BB$
$B \rightarrow \dot{b}$

Step 2: $S \rightarrow PQ$
$P \rightarrow AP / AA$
$\dot{A} \rightarrow \dot{a}$
$Q \rightarrow BQ / BB$
$\dot{B} \rightarrow \dot{b}$

Step 3: $S \rightarrow PQ$
$P \rightarrow \dot{A}P / \dot{A}\dot{A}$
$\dot{A} \rightarrow \dot{a}$
$Q \rightarrow \dot{B}Q / \dot{B}\dot{B}$
$\dot{B} \rightarrow \dot{b}$

Step 4: $S \rightarrow \dot{P}\dot{Q}$
$\dot{P} \rightarrow \dot{A}\dot{P} / \dot{A}\dot{A}$
$\dot{A} \rightarrow \dot{a}$
$\dot{Q} \rightarrow \dot{B}\dot{Q} / \dot{B}\dot{B}$
$\dot{B} \rightarrow \dot{b}$

Step 5: $\dot{S} \rightarrow \dot{P}\dot{Q}$
$\dot{P} \rightarrow \dot{A}\dot{P} / \dot{A}\dot{A}$
$\dot{A} \rightarrow \dot{a}$
$\dot{Q} \rightarrow \dot{B}\dot{Q} / \dot{B}\dot{B}$
$\dot{B} \rightarrow \dot{b}$

2) finiteness:

Here exists an algorithm to determine whether the given CFG generates a finite or infinite language.

Proof: 1) Make the grammar simplify by removing unit production and useless productions.

2) Check whether there is any self-embedded non terminal or not by following steps:

(i) Underline some non-terminal say 'x' which is at L.H.S of production rule.

(ii) The doted x and underline x are two different symbols.

(iii) Put the dot over all the 'x' which are at R.H.S throughout the grammar.

(iv) Put the dot over any non-terminal at L.H.S whose R.H.S contain any doted symbol. This non terminal throughout the grammar.

(v) If the underline 'x' get dotted then 'x' is called self-embedded symbol otherwise not.

3) If the grammar contain any self-embedded non terminal then it generates infinite language, otherwise finite language.

Ex: $S \rightarrow PQa/b$
$P \rightarrow Xb$
$Q \rightarrow bPP$
$X \rightarrow aaa$

Step 1: $S \rightarrow PQa/b$
$P \rightarrow Xb$
$Q \rightarrow bPP$
$\underline{X} \rightarrow aaa$

Step 2: $S \rightarrow PQa/b$
$\dot{P} \rightarrow \dot{X}b$
$Q \rightarrow bPP$
$\underline{X} \rightarrow aaa$

Step 3: $S \rightarrow \dot{P}Q\,a/b$
$\dot{P} \rightarrow \dot{X}b$
$Q \rightarrow b\dot{P}\dot{P}$
$X \rightarrow aaa$

Step 4: $S \rightarrow \dot{P}\dot{Q}a/b$
$\dot{P} \rightarrow \dot{X}b$
$\dot{Q} \rightarrow b\dot{P}\dot{P}$
$\underline{X} \rightarrow aaa$

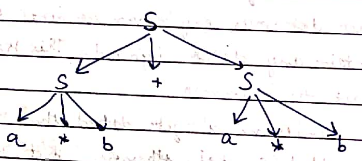
Step 5: $\dot{S} \rightarrow \dot{P}\dot{Q}\,a/b$
$\dot{P} \rightarrow \dot{X}b$
$\dot{Q} \rightarrow b\dot{P}\dot{P}$
$\underline{X} \rightarrow aaa$

Here x is not Self-embedded Symbol.
So, it is finite language.

3) Membership

Here exists an algorithm which tells whether a given string belongs to given grammar to determine this derivation tree is drawn.

Ex: $S \rightarrow S+S / S*S$
$S \rightarrow a/b$
Derive the string $a*b+a*b$



Pumping Lemma for context free language.

It is used to prove that a language is not context free. If A is a context free language then A has a pumping length |P| such that any string S where length of S is $|S| \geq P$, may be divided into 5 parts such that $S = u\,v\,x\,y\,z$ with following conditions must be true:

(i) uv^ixy^iz is in A for every $i \geq 0$
(ii) $|vy| > 0$
(iii) $|vxy| \leq P$

Q. Show that $L = \{a^n b^n c^n / n > 0\}$ is not CFL.

Sol. $L = \{abc, aabbcc, aaabbbccc, - - - \}$

Let $P = 3$

$L = a^3 b^3 c^3$

Case 1: $\underset{u}{aa}\,\underset{v}{ab}\,\underset{x}{b}\,\underset{y}{b}\,\underset{z}{ccc}$

for $i=2$; aa ab ab b bb ccc $\notin L$

Case 2: $\underset{v}{aaa}\,\underset{v}{b}\,\underset{x}{b}\,\underset{y}{b}\,\underset{z}{ccc}$

for $i=2$ aaa bbb bb ccc $\notin L$.

So, it is not context free language.

Undecidable Problems:

Undecidable problems about CFG are

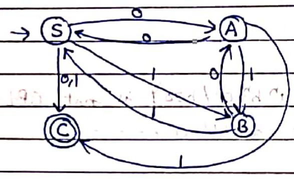
(i) whether or not two diff. CFL define same language.
(ii) whether given CFL is ambiguous or not.
(iii) whether the intersection of two CFL is context free language.
(iv) whether the complement of two CFL is CFL.
(v) There is no algorithm to answer these questions. So, these algorithms are known as undecidable problems.

Scanned by CamScanner

Conversion of Regular Grammar into Automata

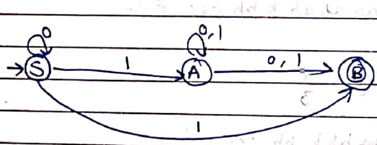
Q. Consider the following grammar
$S \rightarrow 0A / 1B / 0 / 1$
$A \rightarrow 0S / 1B / 1$
$B \rightarrow 0A / 1S$
Find automata for this grammar?

Sol:



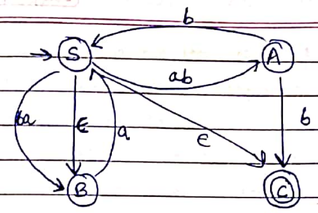
Q. Consider the automata for the following grammar.
$S \rightarrow 0S / 1A / 1$
$A \rightarrow 0A / 1A / 0 / 1$

Sol



Q. Consider the following grammar:
$S \rightarrow abA$
$S \rightarrow B$
$S \rightarrow baB$
$S \rightarrow \epsilon$
$A \rightarrow bS$
$B \rightarrow aS$
$A \rightarrow b$
Construct a NFA trace the transition of m to accept the string w 'abba'.

Sol.



$\rightarrow S \xrightarrow{ab} A \xrightarrow{b} S \xrightarrow{\epsilon} B \xrightarrow{a} S \xrightarrow{\epsilon} C^*$

Q. Design CFG for $\{L = 0^i 1^j \mid i, j \geq 1\}$

Sol.
$S_1 \rightarrow AB$
$A \rightarrow 0A / 00$ for $i > j$
$B \rightarrow 1B / 1$

$S_2 \rightarrow CD$ for $i = j$
$C \rightarrow 00C / 0$
$D \rightarrow 11BD / 1$