Example 3.6.2 : If regular grammar G is given by,

$S \rightarrow as \mid a$ find M accepting L(G)

Solution :

Let q_0 correspond to S and q_f is final state.

$$M = (\{q_0, q_f\}, \{a\}, \delta, q_0, \{q_f\})$$

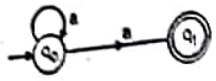


Fig. P. 3.6.2

Exercise

Q. 1. Construct a regular expression corresponding to the state diagram described by Fig. 1

Fig. 1

Q. 2. Find regular expression :

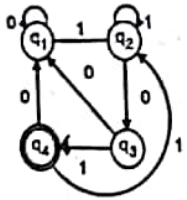


Fig. 2

Q. 3 Construct DFA with reduced state equivalent to regular expression,

$10 + (0 + 11)\,0^*1$.

□□□

CHAPTER 4

Context Free Languages

University Prescribed Syllabus

Context Free Languages : Context-free Languages, Derivation Tree, Ambiguity of Grammar, CFG simplification, Normal Forms, Pumping Lemma for CFG.

Syllabus Topic : Context Free Languages

4.1 Context Free Languages

Write a note on context free languages.

- A grammar is a set of rules for putting strings together and so corresponds to a language.
- A type 2 grammar is called as **context-free grammar** (as A can be replaced by α in any context).
- A language generated by context free grammar is called **a type 2 language** or **a context free language.**

Definition of Context Free Grammar (CFG)

Context Free Grammar consisting a finite set of grammar rules is a quadruple (N, Σ, P, S) where,

N = A set of non-terminal symbols.

Σ = A set of terminals; where, $N \cap T$ = NULL (Set N intersection set T is always equal to NULL)

P = Set of production rules.

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

P : N (NUT)*, i.e. left hand side of production rule P, does have any right context or left context.

S = A starting symbol.

Note : For non-terminals use uppercase letters.

For example

(1)

Start Terminal

↓ ↓

S → as → Non-terminal

S → ε

The grammar : ({A}, {a, b, c}, P,)

Compare this to quadruple (N, T, P, S)

Here,

N → {A}

T → {a, b, c}

∴ P : A → aA

A → a b c

(2) The grammar : {{S, F}, {0, 1}, P, S}

P :

S → 00S | 11F

F → 00F | ε

(3) Construct a context free Grammar G generating all integers (with sign).

∴ G = (N, T, P, S)

N = {S, (sign), (digit), (integer)}

T = {0, 1, 2, ..., 9, +, – }

P consists of S → < sign> <integer>

<sign> → + 1 –

<integer> → <digit> <integer> | <digit>

<digit> → 0|1|2 . . . . |9

L(G) = Set of all integer

e.g. Derivation of – 17 can be obtained as follows :

S ⇒ <sign> <integer>

⇒ – <integer>

⇒ – <digit> <integer>

⇒ – 1 <digit>

⇒ – 17

(4) Production

S → as

S → ε

Derivation for aaaa is,

S ⇒ as

⇒ aas

⇒ aaas

⇒ aaaas

⇒ aaaas

= aaaa

(5) Production :

S → SS

S → a

S → ε

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

Derivation of aa :

$S \Rightarrow SS$

$\Rightarrow SSS$

$\Rightarrow SSa$

$\Rightarrow SSSa$

$\Rightarrow SASA$

$\Rightarrow \varepsilon ASA$

$\Rightarrow \varepsilon a \varepsilon a$

$\Rightarrow aa$

Syllabus Topic : Derivation Tree

4.1.1 Derivation Tree

What is a derivation tree ? Explain it with example.

- Derivation can be represented using tree. Such trees are called **derivation trees.**
- Derivation tree is also called as parse tree.
- Derivation tree is an ordered rooted tree that graphically represents the semantic information of a string derived from a context free grammar.

Representation

- **Root vertex :** Start symbol indicates root vertex.
- **Vertex :** All non-terminal symbols are vertices.
- **Leaves :** All terminal symbols or ε are leaves.

For example

If $S \rightarrow x_1 x_2 \ldots . x_n$ is a production rule in CFG, then parse tree will be as shown in Fig. 4.1.2 .

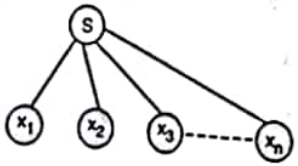


Fig. 4.1.2

Definition

For CFG, G = (N, Σ, P, S) is a **derivation tree** satisfying following condition :

1. Each interior node is labeled by variable in N.
2. The root has label S.
3. Each vertex is labeled by either variable, terminal or ε.
4. A leaf labeled by ε must be the only child of parent.
5. If interior node labeled by A with children labeled by $x_1, x_2, \ldots, x_n$ (from left), then $A \rightarrow x_1, x_2 \ldots . x_n$ must be rule.

Example 4.1.1 : Let G = ({S, A}, {a, b}, P, S)

where, P consists of :

$S \rightarrow aAS \mid a \mid SS$

$A \rightarrow SbA \mid ba$

Draw a derivation tree.

Solution :

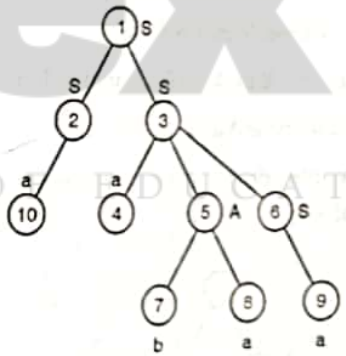


Fig. P. 4.1.1

Explanation

- Vertex S is a root vertex.
- Vertices 4 to 6 are child of 3, written from left using S → aAS production in P.
- Vertices 7 and 8 are child of 5 written from left using A → ba production in P.
- Vertex 5 is internal vertex labeled as A, which is a variable.

E-next THE NEXT LEVEL OF EDUCATION https://E-next.in

Ordering of leaves from left

1. Successors of root (means child of root) are ordered from left by definition.
2. Vertices at level 1 are ordered from left, if v_1 and v_2 are any two vertices at level 1 and v_1 is to the left of v_2, then we can say v_1 is to the left of any child of v_2.
3. Any child of v_1 is to left of v_2 and to the left of any child of v_2. Thus, we get left to right ordering of vertices at level 2.

- Repeating the process upto level n, where n is height of tree.
- Note that ordering of all vertices is from left.

E.g. See the ordering of vertices shown in Fig. 4.1.3

1. Child of root 1 are 2 and 3 order from left
2. Child of 2 namely 10 is a left of any child of 3.
3. The child of 3 ordered from 4-5-6.
4. The vertices at level 2 in the left to right ordering are 10-4-5-6.
5. The vertex 4 is to the left of 6.
6. The child of 5 ordered from left are 7-8, so 4 is to the left of 7. Similarly 8 is to the left of 9.

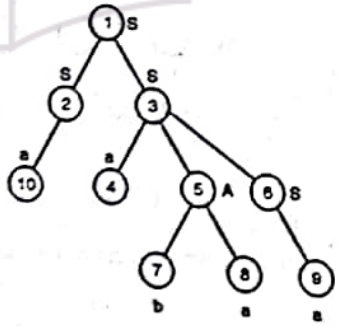


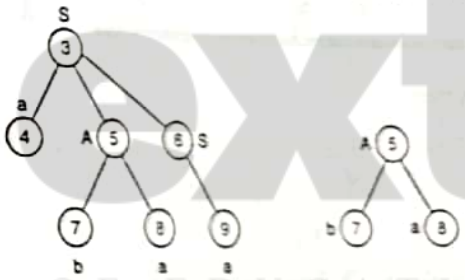
Fig. 4.1.3

- Thus, order from left is 10-4-7-8-9. If we are ordering left to right, direction is always anticlockwise direction.
- Result of derivation tree is concatenation of the label of leaves without repetition in left to right ordering.
- For Fig. 4.1.3, the result of derivation tree is aabaa.

Definition

A subtree of derivation tree 'T' is a tree,

- Whose root is some vertex v of T.
- Whose vertices are descendants of vertex v together with their labels.
- Whose edges are those connecting the descendants of vertex V.



(a) Subtree (b) 'A' subtree

Fig. 4.1.4

Example 4.1.2 : Consider the grammar G whose productions are

$S \rightarrow aASla$

$A \rightarrow SbA|SS|ba$

Show that $S \Rightarrow aabbaa$ and construct a derivation tree.

Solution :

$S \Rightarrow a\underline{A}S$ applying $A \rightarrow SbA$

$\Rightarrow a\underline{S}bAS$ applying $S \rightarrow a$

$\Rightarrow aab\underline{A}S$ applying $A \rightarrow ba$

$\Rightarrow aabba\underline{S}$ applying $S \rightarrow a$

$\Rightarrow aabbaa$

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

Derivation tree :

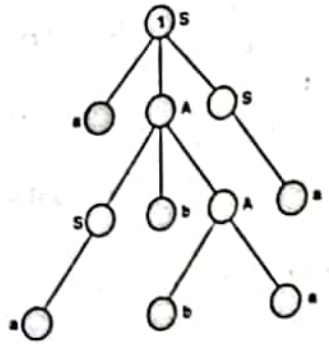


Fig. P. 4.1.2

Derivation trees can be represented in two ways :

A.	Left derivation tree	B.	Right derivation tree

A. Left derivation tree

Left derivation tree is obtained by applying production to the leftmost variable in each step.

For example,

For generating string aabaa from grammar

$S \rightarrow aAS \mid aSS \mid \varepsilon$

$A \rightarrow SbA \mid ba$

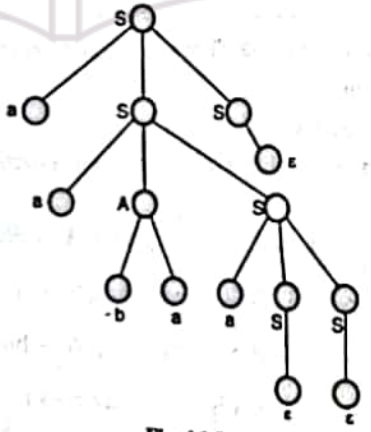


Fig. 4.1.5

B. Right derivation tree

A right derivation tree is obtained by applying production to the right most variable in each step :

For example,

For generating string aabaa from grammar

$S \rightarrow aAS \mid aSS \mid \varepsilon$

$A \rightarrow SbA \mid ba$

Fig. 4.1.6

Example 4.1.3 : For generating string 00110101 form grammar,

$S \rightarrow 0B \mid 1A$

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

$B \rightarrow 1 \mid 1s \mid 0BB$

Find (i) Left most derivation (ii) Right most derivation

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

Solution :

Leftmost derivation

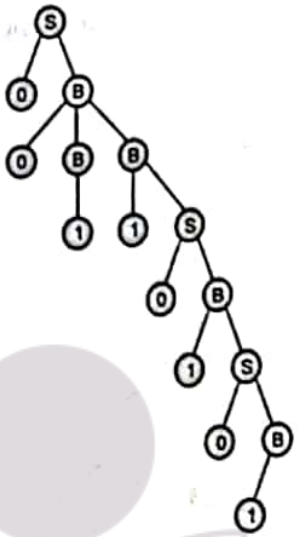


Fig. P. 4.1.3(a)

Rightmost derivation

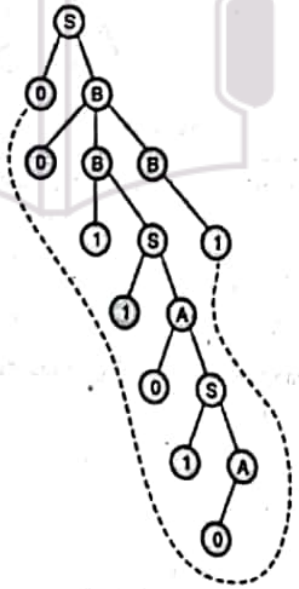


Fig. P. 4.1.3(b)

Syllabus Topic : Ambiguity of Grammar

4.2 Ambiguity in Context Free Grammars

- A CFG is **ambiguous** if one or more terminal strings have multiple left most derivations from the start symbol.
- Deterministic context free grammars are always unambiguous.

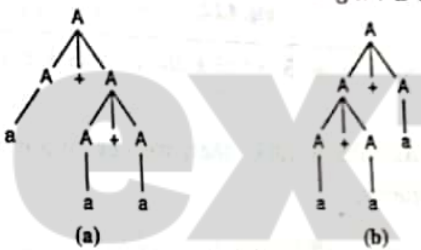
Examples

(i) $A \rightarrow \varepsilon$

$B \rightarrow \varepsilon$

(ii) $A \rightarrow A \mid \varepsilon$

(iii) $A \rightarrow A + A \mid A - A \mid a$ is ambiguous for string a + a + a.



(a) (b)

Fig. 4.2.1

For example

G = {{S}, {a + b, + , *}, P, S} where,

P consists of ;

$S \rightarrow S + S$

$S \rightarrow S * S$

$S \rightarrow a \mid b$

The string a + a * b can be generated as,

Ist way	IInd way
$S \rightarrow S + S$	$S \rightarrow \underline{S} * S$
$\rightarrow a + S$	$\rightarrow S + S * S$
$\rightarrow a + S * S$	$\rightarrow a + S * S$

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

Ist way	IInd way
→ a + a * S	→ a + a * S
→ a + a * b	→ a + a * b

Thus the grammar is ambiguous grammar.

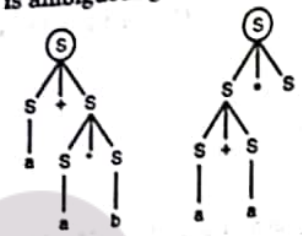


Fig. 4.2.2

Example 4.2.1 : If G is the grammar, S → SbSIa, show that G is ambiguous.

Solution :

Consider string abababa ∈ L(G), then we get two derivation trees for string. Thus G is ambiguous,

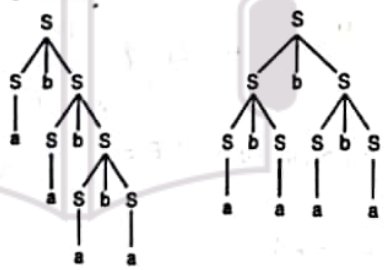


Fig. P. 4.2.1 : Two derivation trees of abababa

exam

Syllabus Topic : CFG Simplification

4.3 Simplification of Context Free Grammars

Write a note on CFG simplification.

- Simplification of CFG means reduction of CFG.
- In CFG, sometimes all production rules and symbols are not needed for the derivation of strings.

- Besides this, there may also be some NULL productions and UNIT production.
- Elimination of these productions and symbols is called simplification of CFG.
- Simplification consists of the following steps :

1. Reduction of CFG
2. Removal of unit productions
3. Removal of Null production

4.3.1 Reduction of CFG

Phase 1 : Derivation of an equivalent grammar G′ from CFG, G such that each variable derives some terminal strings.

Derivation procedure

Step I : Include all symbols w_1 that derives some terminal and initialize i = 1.

Step II : Include symbols w_{i+1} that derives w_i.

Step III : Increment i and repeat step 2 until $w_{i+1} = w_i$.

Step IV : Include all production rules that have w_i in it.

Phase 2 : Derivation of equivalent grammar G″ from the CFG, G′ such that each symbol appears in a sentential form.

Derivation procedure

Step I : Include the start symbol in y1 and initialize i = 1.

Step II : Include all symbols yi + 1, that can be derived from yi and include all production rules that have been applied.

Step III : Increment i and repeat step 2, unit $y_{i+1} = y_i$.

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

By using phase 1 and phase 2, we obtain reduced CFG

Solved Examples

Example 4.3.1 : Find a reduced grammar equivalent to the grammar G, having production rules.

$P : S \rightarrow AclB.\ A \rightarrow a,\ C \rightarrow clBc,\ E \rightarrow aAle$

Solution :

Lets proceed with phase I

Terminal symbols : $-T = \{a, c, e\}$

Set w_1 includes all symbols which are derived from terminal symbols :

$$w_1 = \{A, C, E\}$$

Set w_2 includes all symbols that can derived the symbols present in w_1

$$w_2 = \{A, C, E, S\}$$

Set w_3 includes all symbols that can derived the symbols present in w_2

$$w_3 = \{A, C, E, S\}$$

New Grammar $G' = \{(A, C, E, S), \{a, c, e\}, P, (S)\}$

G = {NT, T, P, S}

P = Production rule

NT = Non-terminal

T = Terminal

S = Start symbol

$P : S \rightarrow AC,\ A \rightarrow a,\ C \rightarrow c\ E \rightarrow aA/e$

Obtain phase II

1. y_1 includes start symbol :

$$y_1 = \{S\}$$

2. y_2 includes all symbols that can be derived from start symbols.

$$y_2 = \{S, A, C\}$$

3. y_3 includes all symbols that can be derived from y_2.

$$y_3 = \{S, A, C, a, c\}$$

$$y_4 = \{S, A, C, a, c\}$$

$$a'' = \{(A, C, S), \{a, c\}, P, \{S\}\}$$

$$P : S \rightarrow AC,\ A \rightarrow a,\ C \rightarrow c$$

Example 4.3.2 : Let $G = (V_n, \Sigma, P, S)$ be given by the production $S \rightarrow AB,\ A \rightarrow a,\ B \rightarrow b,\ B \rightarrow C,\ E \rightarrow c.$ Find G′ such that every variable in G′ derives some terminal string.

Solution :

First obtain phase I

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

$$w_1 = \{A, B, E\}$$

$$w_2 = \{A, B, E, S\}$$

$$w_3 = \{A, B, E, S\}$$

$$G' = \{(A, B, E, S), (a, b, c), P, S\}$$

$$P : S \rightarrow AB, A \rightarrow a, B \rightarrow b, E \rightarrow c$$

Phase II

$$y_1 = \{S\}$$

$$y_2 = \{S, A, B\}$$

$$y_3 = \{S, A, B, a, b\}$$

$$y_4 = \{S, A, B, a, b\}$$

$$G'' = \{(S, A, B), \{a, b\}, P(S)\}$$

$$P : S \rightarrow AB, A \rightarrow a, B \rightarrow b$$

Thus, we obtained new production rule P.

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

Example 4.3.3 : Find reduced grammar equivalent to the grammar G whose productions are, $S \rightarrow AB \mid CA$, $B \rightarrow BC \mid AB$, $A \rightarrow a$, $C \rightarrow aB \mid b$

Solution :

Phase I

Let proceed with phase I.

Terminals symbols : T = {a, b}

Set w_1 includes all symbol which are derived from terminal symbols.

$$w_1 = \{A, C\}$$

Set w_2 includes all symbols derived from the symbols present in w_1

$$w_2 = \{A, C, S\}$$

Set w_3 includes all symbols that are derived from the symbols present in w_2,

$$w_3 = \{A, C, S\}$$

Now grammar G′ = {(A, C, S), (a, b), P, (S)}

Grammar, G = {NT, T, P, S}

New production rule

P : S → AC, A → a, C → b

Phase 2

$$y_1 = \{S\}$$

$$y_2 = \{S, A, C\}$$

$$y_3 = \{S, A, C, a, b\}$$

$$y_4 = \{S, A, C, a, b\}$$

$$G'' = \{(A, C, S), \{a, b\} P, S\}$$

P : S → AC, A → a, C → b

4.3.2 Removal of Unit Productions

Any production rule of the form $A \rightarrow B$ where, A, B, $\in$ non-terminals is called **unit production.**

Procedure for Removal

Step I : To remove $A \rightarrow B$; add production $A \rightarrow X$ to the grammar rule whenever

$B \rightarrow X$ occurs in the grammar [$X \in$ Terminal × can be uni]

Step II : Delete $A \rightarrow B$ from the grammar.

Step III : Repeat from step I until all unit productions are removed.

Example 4.3.3 : Remove unit productions from the grammar whose production rule is given by,

$P : S \rightarrow XY, X \rightarrow a, Y \rightarrow Zb, Z \rightarrow m, M \rightarrow N, N \rightarrow a$

Solution :

Unit productions in our grammar are :

$Y \rightarrow Z, Z \rightarrow M, M \rightarrow N$ (all are unit terminals)

- Since, $N \rightarrow a$, we add $M \rightarrow a$

  $P : S \rightarrow XY, X \rightarrow a, Y \rightarrow z \mid b \rightarrow Z \rightarrow M,$

  $M \rightarrow a, N \rightarrow a$

- Since, $M \rightarrow a$, we add $Z \rightarrow a$

  $P : S \rightarrow XY, X \rightarrow a, Y \rightarrow Z \mid b, Z \rightarrow a$

  $M \rightarrow a, N \rightarrow a$

- Since, $Y \rightarrow Z \mid b$, we add $Y \rightarrow a$

  $P : S \rightarrow XY, X \rightarrow a, Y \rightarrow a \mid b, Z \rightarrow a$

  $M \rightarrow a, N \rightarrow a$

Here, Z, M and N are unreachable symbols because start symbol $S \rightarrow XY$, from S we get X and Y.

So remove the unreachable symbols,

$P : S \rightarrow XY, X \rightarrow a, Y \rightarrow a \mid b$

This is our final production rule.

E-next THE NEXT LEVEL OF EDUCATION https://E-next.in

4.3.3 Removal of NULL Production

In CFG, non-terminal symbol 'A' is a nullable variable, if there is a production, $A \to \varepsilon$ or there is derivation that starts at 'A' and lead to ε (like A, ... → E).

Procedure for Removal

Step I : To remove $A \to \varepsilon$, look for all productions whose right side contains A.

Step II : Replace each occurrences of 'A' in each of their productions with ε.

Step III : Add the resultant productions to the grammar.

Example 4.3.4 : Remove NULL production from the following grammar :
$S \to ABAC, A \to aA|\varepsilon, B \to bB|\varepsilon, C \to c$

Solution :

First find out NULL production, $A \to \varepsilon$, $B \to \varepsilon$,

these two null productions we have to remove.

(a) To eliminate $A \to \varepsilon$

(i) $S \to ABAC$

(ii) Replace each occurrences of A in each of these production with ε.

$S \to ABAC$

$S \to ABC|BAC|BC$

$A \to aA$

$A \to a$

New production :

P : $S \to ABAC \mid ABC \mid BAC \mid BC$

$A \to aA \mid a$

$B \to bB|\varepsilon$

$C \to c$

(b) To eliminate $B \to \varepsilon$,

We have to check new production where we find B on right side of any production that replace with ε.

$S \to ABAC \mid ABC \mid BAC \mid BC$

Replace B with ε

We get,

$S \to AAC \mid AC \mid C$

$B \to b$

New production :

$S \to ABAC \mid ABC \mid BAC \mid BC \mid AAC \mid AC \mid C$

$A \to aA|a$

$B \to bB|b$

$C \to c$

This is our result.

Example 4.3.5 : Consider the grammar G where production are $S \to AS \mid AB$, $A \to \varepsilon$, $B \to \varepsilon$, $D \to b$. Construct a grammar G_1 without null productions generating.

Solution :

First find out NULL production,

$A \to \varepsilon$, $B \to \varepsilon$

(a) To eliminate $A \to \varepsilon$

$S \to aS \mid AB$

Replace A in each of these production with ε.

$\therefore S \to aS \mid AB$

$\to aS \mid B$

New production

P : $S \to aS \mid AB \mid B$

E-next NEXT LEVEL OF EDUCATION

https://E-next.in

$B \rightarrow \varepsilon$

$D \rightarrow b$

(b) To eliminate $B \rightarrow \varepsilon$

Replace B in each of these productions with ε

$S \rightarrow aS \mid AB$

$S \rightarrow aS \mid A$

New production :

$P : S \rightarrow aS \mid AB \mid A$

$D \rightarrow b$

$P : S \rightarrow aS, S \rightarrow AB; S \rightarrow A, D \rightarrow b$

Syllabus Topic : Normal Forms

4.4 Normal Forms

☞ **Write a note on Normal forms.**

- The process of removal of duplication of production is called **normalized CFG.**
- In CFG, the RHS of a production can be any string of variable and terminals.
- When production in grammar G satisfies certain restriction, then G is said to be a normal forms.
- There are two normal forms :
  - (i) Chomsky Normal Form (CNF)
  - (ii) Greibach Normal Form (GNF)

4.4.1 Chomsky Normal Form(CNF)

- In Chomsky Normal Form (CNF), we have restriction on the length of RHS, which is elements in RHS should either be two variables or terminal.

A CFG is in Chomstry Normal Form if the productions are in the following forms :

$A \rightarrow a$

$A \rightarrow BC$

Where A, B and C are Non-terminals and A is a terminal.

Convert a given CFG to Chomsky Normal Form

Step I : If the start symbol S occurs on some right side, create a new start symbol S′ and new production.

$S' \rightarrow S$

Step 2 : Remove NULL productions

(we already discussed in previous section)

Step 3 : Remove unit production

(we already discussed in previous section)

Step 4 : Replace each production $A \rightarrow B_1 \ldots . B_n$ where $n > 2$ with $A \rightarrow B_1 C$ where $C \rightarrow B_2 \ldots . B_n$

Repeat this step for all productions having two or more symbols on right side.

Step 5 : If right side of any production is in the form $A \rightarrow aB$, where a is 'a' terminal and A and B are non terminals, then the production is replaced by $A \rightarrow XB$ and $X \rightarrow a$

Repeat this step for every production which is of the form $A \rightarrow aB$.

For example

Convert the following CFG to CNF :

$P : S \rightarrow AS\,A \mid aB, \quad A \rightarrow B \mid S, \quad B \rightarrow b \mid \varepsilon$

Step 1 : Start symbol S occurs on right side in production

$A \rightarrow B \mid S$ and $S \rightarrow ASA$

We add a new state S′ and $S' \rightarrow S$ is added to production.

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

$P : S' \to S,\ S \to ASA|aB,\ A \to B|S,\ B \to b|\varepsilon$

Step 2 : Remove NULL production :

$B \to \varepsilon, A \to \varepsilon$

After removing $B \to \varepsilon$,

$P : S' \to S, S \to AS\,A \mid aB \mid a,\ A \to B|S|\varepsilon, B \to b$

After removing $A \to \varepsilon$,

$P : S' \to S, S \to AS\,A \mid aB \mid a \mid AS \mid SA \mid S, A \to B|S, B \to b$

Step 3 : Remove unit production :

Unit production in our problem :

$S \to S, S' \to S, A \to B$ and $A \to S$

After removing $S \to S$,

$P : S' \to S, S \to ASA \mid aB \mid a \mid AS \mid SA$

$A \to B|S, B \to b$

After removing $S' \to S$,

$P : S' \to ASA \mid aB \mid a \mid AS \mid SA$

$S' \to ASA \mid aB \mid A \mid AS \mid SA$

$A \to B|S,\ B \to b$

After removing $A \to B$,

$P : S' \to ASA \mid aB \mid a \mid AS \mid SA$

$S \to ASA \mid aB \mid a \mid AS \mid SA$

$A \to b/S, B \to b$

After removing $A \to S$,

$P : S' \to ASA \mid aB \mid a \mid AS \mid SA$

$S \to ASA \mid aB \mid a \mid AS \mid SA$

$A \to b \mid ASA \mid aB \mid a \mid AS \mid SA$

$B \to b$

Step 4 : Now find out the productions that has more than two variables in RHS.

$S' \to ASA$, $S \to ASA$ and $A \to ASA$

After removing these we get :

SA will be replaced by X.

$S' \to AX \mid aB \mid a \mid AS \mid SA$

$S \to AX \mid aB \mid a \mid AS \mid SA$

$A \to b \mid AX \mid aB \mid a \mid AS \mid SA$

$B \to b$

$X \to SA$

Step 5 : Now change the productions

$S' \to aB$, $S \to aB$, and $A \to aB$

a is replaced with variable Y and add production $Y \to a$

Finally we get :

$P : S' \to AX \mid YB \mid a \mid AS \mid SA$

$S \to AX \mid YB \mid a \mid AS \mid SA$

$A \to b \mid AX \mid YB \mid a \mid AS \mid SA$

$B \to b$

$X \to SA$

$Y \to a$

This is our result, which is the required Chomsky Normal Form for the given CFG.

4.4.2 Greibach Normal Form(GNF)

- CFG is in GNF, if its productions are in form of :

$A \to a\alpha$

$\downarrow \quad \downarrow \to$ String variable or ε

terminals

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

For example

$S \rightarrow aAB \mid bBAC$

$A \rightarrow a$

$B \rightarrow b$

$C \rightarrow cC$

These all productions are in GNF

Conversion of CFG to GNF

Step 1 : Check whether the given CFG is a simplified grammar i.e. it should not have null production, unit production or useless symbol.

Step 2 : Check whether the CFG is already converted into CNF.

Step 3 : Change the names of non-terminal symbols to A_i.

For example

$S \rightarrow XA \mid BB$

$B \rightarrow b \mid SB$

$X \rightarrow b$

$A \rightarrow a$

Replace,

S with A_1

X with A_2

A with A_3

B with A_4

After replacing this, we get,

Production P :

$A_1 \rightarrow A_2A_3 \mid A_4A_4$

$A_4 \rightarrow b \mid A_1A_4$

$A_2 \rightarrow a$

$A_3 \rightarrow a$

Step 4 : Modify the rules so that non terminals are in ascending order.

If $A_i \rightarrow A_{jx}$ is a rule, then $i < j$

and ($i > j$ or $i = j$) are not acceptable.

Check production, line by line

(1) $A_1 \rightarrow A_2 A_3$

$i = 1, j = 2$

$i < j$, this production is acceptable.

(2) $A_1 \rightarrow A_4 A_4$

$i = 1, j = 4$

$i < j$ this production is acceptable.

(3) $A_4 \rightarrow A_1 A_4$

$i = 4, j = 1$

$i > j$ which is not acceptable.

So it is not in GNF, we have to convert it into GNF.

Replace value of A_1 in A_4 ($A_1 \rightarrow A_2 A_3$)

$A_4 \rightarrow b \mid A_2 A_3 A_4$

Now which $A_4 \rightarrow A_2 A_3 A_4 \mid A_4 A_4 A_4$

Here, $i = 4, j = 2$

$i > j$ which is not acceptable

so replace A_2 with b.

$A_4 \rightarrow b \mid b A_3 A_4 \mid A_4 A_4 A_4$

$\therefore A_4 \rightarrow A_4 A_4 A_4$

Here, $i = 4, j = 4$

$i = j$ in which we find left recursion.

Now remove left recursion :

when ($A_i = A_j$)

$A_4 \rightarrow A_4 A_4 A_4$

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

Introduce a new variable and write a whole production with z variable.

$A_4 \rightarrow b \mid b\,A_3\,A_4 \mid bZ \mid b\,A_3A_4\,Z$

$Z \rightarrow A_4\,A_4\,Z \mid A_4\,A_4$

Now grammar is :

$A_1 \rightarrow A_2\,A_3 \mid A_4\,A_4$

$A_4 \rightarrow b \mid b\,A_3\,A_4 \mid bZ \mid bZ$

$Z \rightarrow A_4A_4 \mid A_4\,A_4\,Z$

$A_2 \rightarrow b$

$A_3 \rightarrow a$

Repeat procedure again :

Substitute the value of A_4 and A_2

$A_1 \rightarrow bA_3 \mid b\,A_3\,A_4\,A_4 \mid b\,A_4 \mid b\,A_3\,A_4\,Z\,A_4 \mid bZ\,A_4$

$A_4 \rightarrow b\,A_3\,A_4 \mid b \mid b\,A_3\,A_4\,Z \mid bZ$

$Z \rightarrow bA_3\,A_4\,A_4 \mid bA_4 \mid bA_3\,A_4Z\,A_4 \mid bZ\,A_4 \mid bA_3\,A_4\,A_4\,Z \mid bA_4\,Z \mid bA_3\,A_4ZA_4Z \mid bZA_4Z$

$A_2 \rightarrow b$

$A_3 \rightarrow a$

This is the result of our example.

Exam

Syllabus Topic : Pumping Lemma for CFG

4.5 Pumping Lemma for CFG

- In formal language theory, the pumping lemma for Context Free Language (CFL) also known as the 'Bar-Hill-Lemma'. This lemma gives a property shared by CFL's.
- The pumping lemma for context-free languages describes property that all context-free languages are guaranteed to have.
- All strings in languages must have a length of at least p, where p is called pumping length which is constant.

- The pumping lemma gives us a technique to show that certain languages are not context free.
- The process of 'pumping up' additional copies of strings is what gives the pumping lemma its name

It was first proved by Dona Scott and Michael Rabin in 1959 and rediscovered shortly by Elishamil and Micha Perles in 1961 as a simpulticaton pumping lemma for CFL.

4.5.1 Definition of Pumping Lemma for CFL

Let L be a context free language, then there is a constant n which depends only on language L. Such that there exists a string $z \in L$ and $|z| \geq n$. (length of z is $\geq n$) where,

$$z = uvwxy$$

Conditions

1. $|vwx| \leq n$
2. $|vx| \geq 1$
3. For All $i \geq 0$, $uv^iwx^iy \in L$.

Example :

Using pumping lemma, prove the following language is not CFL.

$$L = 0^n\,1^n\,2^n \mid n \geq 1$$

$$L = \{012, 001122, 000111222 \ldots\ldots\}$$

Z =	0	0	1	1	2	2
	u	v		w	x	y

Length of string n = 6

Case (i)

$$|vw\,x| \leq n$$

$$|vwx| = 2 + 1 + 1 = 4$$

$$4 \leq 6$$

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

Case (ii)

$$|vx| \geq 1$$

$$|vx| = 2 + 1 = 3$$

$$3 \geq 1$$

Case (iii)

$$uv^i wx^i y \in L$$

$$i = 2$$

where u = 0, v = 0 1, w = 1, x = 2 and y = 2.

Using these value make a string $uv^i wx^i y$,

which is 0 0 1 0 1 1 2 2 2

0 0 1 0 1 1 2 2 2 does not belong to our languages L.

$\therefore$ 0 0 1 0 1 1 2 2 2 $\notin$ L.

By pumping lemma, we proved that the given language is not a context free language.

Example 4.5.1 : Consider a context free grammar G with the following productions,

S → ASA | B

B → aCb | bCa

C → ACA | A

A → a | b

(i) What are the variable and terminals of G

(ii) Give 3 strings of length 7 in L (G)

Solution :

(i) Variable and terminals of G

V_N = {S, A, B, C}

Terminals = Σ = {a, b}

(ii) Strings of length 7 in given L (G)

S ⇒ AASAA ⇒ AABAA ⇒ AAaCbAA

⇒ AAaAbAA ⇒ ababbab

Exercise

Q. 1. Find reduced grammar equivalent to the grammar G whose productions are

S → AB | CA, B → BC|AB, A → a

C → aB|b

Ans. : New production rule : P : S → CA, A → a, C → b

Q. 2 Construct a reduced grammar equivalent to grammar.

S → aAa, A → sb | bCC | DaA

C → abb|DD, E → aC, D → aDA

Ans. : New production rule : P : S → aAa,

A → sb | bCC,

C → abb

Q. 3 Find the following grammar G to CNF G is,

S → aAD, A → aB | bAB, B → b, D → d

Q. 4 Find a grammar in Chomsky Normal Form equivalent to S → aAbB, A → aAa, B → bBb.

Q. 5 Reduce the following grammar to CNF

S → ASA|bA, A → B|S, B → C

Show that CFG, is ambiguous grammer with production ;

S → SS | (S) | ε

Q. 6 Find derivation tree of a * b + a * b given that a * b + a * b is in L(G), where

G is given by S → S + S | S * S

S → a|b

Q. 7 A context free grammar G has following production.

S → 0S0 | 1S1 | A

A → 2B3

B → 2B3|3 Describe the language generated by parameters.

E-next THE NEXT LEVEL OF EDUCATION

https://E-next.in

Q. 8 Consider the following productions :

$S \rightarrow aB \mid bA$

$A \rightarrow as \mid bAAla$

$B \rightarrow bs \mid aBB \mid b$

For the string aaabbabbba find

1. left most derivation
2. right most derivation
3. parse tree

Q. 9 Show that grammar is ambiguous

G : $S \rightarrow alabSblaAb$

$A \rightarrow bSlaAAb$

Q. 10 Show that grammar is ambiguous

G : $S \rightarrow aBlab$

$A \rightarrow aAB \mid a$

$B \rightarrow ABblb$

□□□

CHAPTER 5

Turing Machines

University Prescribed Syllabus

Turing Machines : Turing Machine Definition, Representations, Acceptability by Turing Machines, Designing and Description of Turing Machines, Turing Machine Construction, Variants of Turing Machine

Linear Bound Automata : The Linear Bound Automata Model, Linear Bound Automata and Languages

Pushdown Automata : Definitions, Acceptance by PDA, PDA and CFG

E-next THE NEXT LEVEL OF EDUCATION

5.1 Introduction to Turing Machine

- In 1936, Mr. Alan Turing introduced advanced mathematical model for modern digital computer which was known as Turing machine. Turing machine is advanced machine of FA and PDA.
- This simple mathematical model has no difference between input and output set. This mathematical model can be constructed to accept a given language or to carry out some algorithm.
- This model sometimes uses it's own output as input for further computation.
- This machine or model can select current location and also decides location of memory by moving left or right. This mathematical model known as Turing machine has become an effective procedure.

https://E-next.in