Minimization of CFG

→ To make context free Grammar easier and efficient we must minimize CFG.

→ To minimise CFG

(i) To Remove ϵ - productions

(ii) Remove Unit Productions.

(iii) Eliminate useless symbols.

Removal of ∈-null Productions:

→ First we find the set V_n of all nullable variables of "G" using the following steps.

1) For all productions $A \to \epsilon$ place "A" into V_n.

2) Repeat this step until no furthur variables are added to V_n.

3) For all productions

$B \to A_1, A_2 \cdots A_n$, If $A_1, A_2 \cdots A_n$ are alre-ady in V_n then place B to V_n.

4) After finding nullable variable set we can get the resultant productions (which does'nt contain ∈ productions) as follows.

→ Include the production into the resu-ltant grammar such that it does'nt contain nullable variables.

→ For each production in "P" except null productions we put into the resultant grammar that production as well as all those generated by replacing nullable variable with ∈ in all possible combinations.

Eliminate null productions from the following.

$S \to aS \mid AB$

$A \to \epsilon$

$B \to \epsilon$

$D \to b$

First we find nullable variable set.

$V_n = \{A, B\}$ ($A \to \epsilon$, $B \to \epsilon$ (since they are the productions in P))

Consider the productions $S \to AB$ as "S" is deriving into 2 nullable variables A, B which are present in V_n so add "S" to V_n.

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

In the given grammar we cannot add any more variable to V_n

so the final nullable variable set

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

The resultant productions which doesn't contain ϵ productions can be obtained as follows.

The production $D \to b$ doesn't have any nullable variables, hence it is directly included into the resultant grammar.

→ For all the remaining production except null productions replace nullable variables with ε in all possible combination.

S→aS gives S→aS, S→a

S→AB gives S→AB, S→A, S→B.

The resultant grammar which doesn't contain ε productions are

S→ aS|a|AB|A|B

D→b

Eliminate null productions from the following grammar

S→ AaB|aaB

A→ε

B→bbA|ε

The nullable variable set to A

$V_n = \{A, B\}$

The productions are

S→ AaB gives S→ AaB

S→ aB, S→a

S→aaB gives S→ aaB, S→aa

S→ bbA gives B→bbA, B→bb

The final productions are

S→ AaB|Aa|aB|a|aaB|aa

B → bbA | bb.

Removal of Unit Productions

→ A production is of the form A → D where A, B are variables is called unit production

→ These unit productions are undesirable for CFG.

→ To remove unit productions from CFG, we use substition method.

→ This method can be applied only to CFG which doesn't contain null productions.

Eliminate Unit productions from the following Grammar.

S → Aa | B

B → A | bb

A → a | bc | B.

As there are no null productions in the given grammar we can directly apply the substitution method for removal of unit production.

The unit production are

S → B

B → A.

A → B

B → B is substituted by B → bb

S → B → A is substituted by S → a|bc

B → A is substituted by B → a|bc

A → B || A → bb

final resultant grammar which doesn't contain unit productions are

S → Aa | bb | a | bc

B → a | bc | bb

A → a | bc | bb.

Removal of useless symbols:-

To identify and eliminate the useless symbols we should apply the following theorems.

Theorem 1:- If G is a context free grammar we can find an equivalent grammar G' such that each variable in G' derives some terminal string.

→ Let $G = \{V, T, P, S\}$ we can define

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

Construction of V':-

We define a set $W_i \subseteq V$ by recursion

$$W_1 = \{A \in V \mid \exists\ A \to w \text{ where } w \in T^*\}$$

$$W_{i+1} = \{W_i \cup \{A \in V \mid \exists\ A \to \alpha \text{ with } \alpha \in (W_i \cup T)^*\}$$

This procedure is repeated until no more variables are added to the set.

$W_k = V'$ for some value of k

The useless variables are $V - V'$.

Construction of P':-

$$P' = \{A \to \alpha \mid A, \alpha \in (V' \cup T)^*\}$$

① Let CFG is $G = \{S \to AB$
$A \to a$
$B \to b \mid CC$
$C \to C\}$

Find G' such that every variable in G' must derive some terminal string.

Sol: $\omega_1 = \{A, B, \epsilon\}$

$\omega_2 = \omega_1 \cup \{S_1\} = \{A, B, \epsilon, S\}$

$\omega_3 = \omega_2 \cup \{\phi\}$

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

$P' = \{ S \to AB$

$A \to a$

$B \to b$

$E \to c \}$

We can construct $G' = (V', T, P', S)$ as follows.

Now, $V' = \{A, B, \epsilon, S\}$ and

$P' = \{ S \to AB$

$A \to a$

$B \to b$

$\epsilon \to c \}$

As "ϵ" is not deriving any terminal it is eliminated & the production $B \to CC$ is also eliminated.

2. For Theorem 2:

For every CFG $G = (V, T, P, S)$ we can construct an equivalent Grammar G'.

$G' = (V', P', T', S)$ such that every symbol should appear in some sentential

form.

@ Construction of W_i for $i \geq 1$

$W_1 = \{S\}$

$W_{i+1} = W_i \cup \{X \in (V \cup T) \mid \exists\ A \to \alpha$ where

$A \in W_i$ and α contains symbol $X\}$

→ We repeat this process until no more new elements are added to the set & we call it as W_k.

Consider CFG $G = (\{S, A, B, C\}, \{a, b, c\}, P, S)$ where P consists of

$S \to AB$

$A \to a$

$B \to b$

$\epsilon \to c$

Elemenate useless symbols which are not present in the sentential form.

$W_1 = \{S\}$

$W_2 = W_1 \cup \{X \in V \cup T \mid \exists\ A \to \alpha$

$A \in W_1$ & α contains $X\}$

$= W_1 \cup \{A, B\}$

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

$W_3 = W_2 \cup \{X \in V \cup T \mid \exists\ A \to \alpha$

$A \in W_2$ & α contains $X\}$

$W_3 = W_2 \cup \{a, b\}$

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

$W_4 = W_3 \cup \{X \in V \cup T \mid \exists\ A \to \alpha, A \in W_2,$ X contains $\alpha\}$

$W_4 = W_3 \cup \{\phi\}$

Only S, A, B, a, b are present in the sentential form. The remaining symbols C, c are not present in the sentential form.

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

→ Construct minimized or Reduced Grammar for the following CFG

$S \to aAa$

$A \to Sb \mid bCC \mid DaA$

$C \to abb \mid DD$

$E \to ac$

$D \to aDA$

As there are no null & unit productions directly we will identify and remove the useless symbols as follows.

→ Apply theorem 1 i.e every variable must derive some terminal string.

$W_1 = \{C\}$ (∵ $C \to abb$ is a production (which derives only terminals))

$W_2 = W_1 \cup \{A, \epsilon\}$ (Since A, ϵ are deriving the terminal or the element present in W_1)

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

$W_3 = W_2 \cup \{S\}$

$W_3 = \{C, A, \epsilon, S\}$

$W_4 = W_3 \cup \{\phi\}$

The variable D is not deriving any terminal string so, it can be eliminated.

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

where P' consists $S \to aAa$

$A \to Sb \mid bCC$

$C \to abb$

$\epsilon \to ac$

→ Now, theorem 2 for the above resultant grammar.

$W_1 = \{S\}$

$W_2 = W_1 \cup \{X \in V \cup T \mid \exists A \to \alpha, A \to W_1, \alpha$ contains $X\}$

$W_1 \cup \{a, A\}$

$W_2 = \{S, a, A\}$

$W_3 = W_2 \cup \{b, C\}$

$W_3 = \{S, a, A, b, C\}$

$W_4 = W_3 \cup \{x \in V \cup T \mid \exists A \to \alpha$ where $A \to W_3, \alpha$ contains $x\}$

Only the e is not present in any sentential form. So, it can be eliminated. So our the final or reduced minimal grammar is

16/2/17 NORMAL FORMS:

In a CFG the right hand side of the production can be any string of variables and terminals. When the productions satisfy certain restrictions then that grammar is said to be in a normal form. Two widely used normal forms for CFG are

(i) Chomsky normal form (CNF)

(ii) Greibach Normal form (GNF)

Chomsky normal form:

Any CFG in which all productions is of the form $A \rightarrow BC$ (or) $A \rightarrow a$ where A, B, C are variables and a is a terminal is said to be in CNF. i.e

Reduction to CNF:

→ All the productions which are not in the req form can be reduced as follows.

$A \rightarrow w_1 w_2 - - - - w_n$

with some terminals on the right hand side.

→ If W_i is a terminal say a_i, add a new variable C_{ai} & $C_{ai} \to a_i$ to the

→ Add $C_{ai} \to a_i$ for the productions.

Repeat same for all terminals.

→ $A \to A_1 A_2 \cdots A_m$ where $m \geq 3$ can be reduced as $A \to A_1 C_1$

$$C_1 \to A_2 C_2$$

$$C_2 \to A_3 C_3$$

$$\vdots$$

Find an equivalent grammar in CNF for the following grammar

$G = (\{S, A, B\}, \{a, b\}, P, S)$ where P consists of $S \to bA \mid aB$

$A \to bAA \mid as \mid a$

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

→ only $A \to a$ & $B \to b$ is in the req form

All other productions should be reduced into required form.

$S \to bA$ is splitted into

$S \to C_bA$.

$C_b \to b$

The splitting production $S \to C_bA$

$C_b \to b$ are in the req form so

$S \to ba$ is replaced by $S \to C_bA$, $C_b \to b$

→ A → bAA is splitted into

$A \to C_b AA \quad C_b \to b$

$A \to C_b C_1$ (Again splitting into)

$C_1 \to AA$.

$A \to bAA$ is replaced by

$A \to C_b C_1$; $C_1 \to AA$; $C_b \to b$

→ $B \to aBB$.

$B \to C_a BB \quad C_a \to a$

$B \to C_a C_2$

$C_2 \to BB$.

$B \to aBB$ is replaced by

$B \to C_a C_2$; $C_2 \to BB$, $C_a \to a$.

→ $A \to aS$

$A \to C_a S$

$C_a \to a$

→ $B \to bS$

$B \to C_b S$

$C_b \to b$

→ $S \to aB$

$S \to C_a B$, $C_a \to a$

The final productions of CFG in CNF are:

$A \to a$
$B \to b$
$S \to C_b A$
$C_b \to b$
$S \to C_a B$
$C_a \to a$

$A \to C_b C_1$
$C_1 \to AA$
$C_b \to b$
$B \to C_b C_2$
$C_2 \to BB$
$C_a \to a$

$A \to C_a S$
$C_a \to a$
$B \to C_b S$
$C_b \to b$.

Convert the following grammar into CNF.

$S \to ABa$
$A \to aab$
$B \to Ac$

No real normal forms r present

$S \to ABa$ | $A \to aab$ | $B \to Ac$
$S \to ABC_a$ | $A \to C_a C_a b$ | $B \to AC_c$
$C_a \to a$ | $C_c \to c$
$A \to C_a C_a C_b$
$C_b \to b$
$A \to C_a C_2$
$C_2 \to C_a b$

Greibach Normal form (GNF):-

In GNF constraints are placed on the positions in which terminals & variables can appear.

→ CFG is said to be in GNF if all productions are of the form.

$$A \to a\alpha$$

or

$$A \to a$$

where $\alpha \in V^*$

→ To convert given grammer into GNF the following lemmas are used

Lemma 1:-

Let $A \to B\alpha$ is a production in P & let $B \Rightarrow B_1 | B_2 | \dots | B_n$ & the set of all B productions. Then $A \to B\alpha$ is replaced by

$$A \to B_1\alpha | B_2\alpha | B_3\alpha | \dots | B_n\alpha$$

Lemma 2:-

Let $A \to A\alpha_1 | A\alpha_2 | \dots | A\alpha_n$. are the set of A productions for which A is the left most symbol of the RHS. and $A \to B_1 | B_2 | \dots | B_m$. are the remaining A productions. Then all these

productions can be replaced by done.

$$\left.\begin{array}{l} A \to B_i \\ A \to B_i Z \end{array}\right\} \text{where } 1 \le i \le n.$$

$$\left.\begin{array}{l} Z \to \alpha_i \\ Z \to \alpha_i Z \end{array}\right\} \text{where } 1 \le i \le n$$

Procedure to convert Grammar into GNF:-

→ Grammar should be in CNF.

→ Rename all variables as same variable name but with different subscripts

for ex:

→ S, A, B are variables then rename as A_1, A_2, A_3 respectively.

→ Choose a production such that left hand side variables subscript is greater than the RHS starting variable subscript. Then apply lemma 1 & lemma 2 according to the productions.

→ Repeat applying of lemma1 and lemma2 for all the productions till the grammar comes into GNF.

Prob convert the following grammar into GNF.

$S \to AB$

$A \to Bs/b$

$B \to SA/a$.

Sol: As the given is already in CNF directly we can rename the variables.

Rename S, A, B as A_1, A_2, A_3.

Now productions can be rewritten as

$A_1 \to A_2 A_3$ —①

$A_2 \to A_3 A_1 / b$ —②

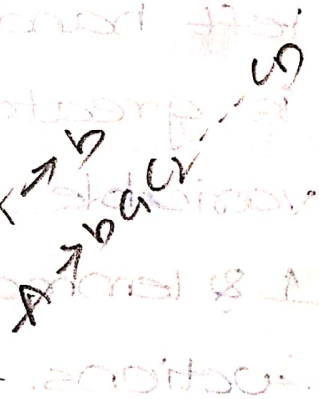
$A_3 \to A_1 A_2 / a$ —③

Choose a production $A_3 \to A_1 A_2 / a$ since its Left HS variable is > than RHS starting variable subscript

$A_3 \to A_1 A_2 / a$

$A_3 \to A_2 A_3 A_2 / a$

$A_3 \to A_3 A_1 A_3 A_2 / bA_3 A_2 / a$



Categorize into 2 productions according to lemma 2

$A_3 \to A_3 A_1 A_3 A_2 \Rightarrow \alpha = A_1 A_3 A_2$

$A_3 \to bA_3 A_2 / a$

$B_1 = bA_3A_3'$

To convert into req GNF

$A_3 \rightarrow bA_3A_2 \mid a$

$A_3 \rightarrow bA_3A_2Z \mid aZ$

$Z \rightarrow A_1A_3A_2$

$Z = A_1A_3A_2Z$.

Now all the A_3 productions are in the req. form.

Observing A_3 productions & (2) we can apply lemma 2

$A_2 \rightarrow bA_3A_2A_1 \mid aA_1 \mid bA_3A_2ZA_1 \mid aZA_1 \mid b$

All A_2 productions are in the req. form

Observing A_2 productions & (1) we can apply lemma (1)

$A_1 \rightarrow bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3A_2ZA_1A_3 \mid aZA_1A_3 \mid bA_3$

Now all A_1 productions are in the req form.

By observing A_1 productions & 2 productions we can apply lemma 1

$Z \rightarrow bA_3A_2A_1A_3A_3A_2 \mid aA_1A_3A_3A_2 \mid bA_3A_2ZA_1A_3A_3A_2 \mid aZA_1A_3A_3A_2 \mid bA_3A_3A_2$.

$Z \rightarrow bA_3A_2A_1A_3A_3A_2Z \mid aA_1A_3A_3A_2Z \mid$

$bA_3A_2ZA_1A_3A_3A_2Z \mid aZA_1A_3A_3A_2Z \mid bA_3A_3A_2Z.$

Now all "Z" productions are in the req form

The final resultant productions in GNF are

$A_1 - S$

$A_2 - A$

$A_3 - B$

$S \rightarrow bBASB \mid aSB \mid bBAZSB \mid aZSB \mid bB$

$A \rightarrow bBAS \mid aS \mid bBAZS \mid aZS \mid b$

$A_3 \rightarrow bBA \mid a \mid bBAZ \mid aZ.$

$Z \rightarrow bBASBBA \mid aSBBA \mid bBAZSBBA \mid aZSBBA \mid$
$bBBA.$

$Z \rightarrow bBASBBAZ \mid aSBBAZ \mid bBAZSBBAZ \mid$
$aZSBBAZ \mid bBBAZ.$

Pumping lemma for context free languages:-

Let L be a context free language, for some natural number 'n', every $z \in L$ with $|z| \geq n$ can be decomposed as "uvwxy" such that $|vx| \geq 1$

$$|vwx| \leq n$$

then according to pumping lemma $\forall$ $k \geq 0$ the generated strings $uv^k wx^k y \in L$

s.t the language $L = \{a^p \mid p \text{ is prime number}\}$ is not context free.

$L = \{aa, aaa, aaaaa, \ldots\}$

Considering $n = 5$

$\underbrace{aa}_{u}\underbrace{a}_{v}\underbrace{a}_{w}\underbrace{a}_{x} = \text{string}$

$y = \epsilon$

$|vxy| = 2 \geq 1$

$|vwx| = 3 \leq 5$

for diff values of k.

$k=1$, $uv^1wx^1y = aaaaa\epsilon = aaaaa \in L$ - Regular

$k=2$, $aaaaaaa\epsilon \in L$

K=3, aaaaaaaaaae $\notin L$

for K=3, the generated string is not regular.

Applications of Context Free Grammar.

→ In the implementation of YACC parser generator.

YACC (Yet Another compiler compiler)

→ In XML structured definitions

→ To define new programming language syntax rules.

Note: In a context Free grammar without "ε" generating & whose productions are of the form $A \to ax$ where A is any variable 'a' is terminal.

$\alpha \in (V+T)^*$ or $\alpha \in (V \cup T)^*$ is said to be in GNF (Greibach normal form).

→ In CFG without "ε" generating & whose productions are of the form $A \to BC$ or $A \to a$ where A, B, C ∈ V and a is terminal is said to be in Chomsky Normal Form.

→ Grammar is a generating device where as automata is a acceptor device.