

Section: Transforming grammars (Ch. 6)

Methods for Transforming Grammars

We will consider CFL without λ . It would be easy to add λ to any grammar by adding a new start symbol S_0 ,

$$S_0 \rightarrow S \mid \lambda$$

Theorem (Substitution) Let G be a CFG. Suppose G contains

$$A \rightarrow x_1 B x_2$$

where A and B are different variables, and B has the productions

$$B \rightarrow y_1 | y_2 | \dots | y_n$$

Then can construct G' from G by deleting

$$A \rightarrow x_1 B x_2$$

from P and adding to it

$$A \rightarrow x_1 y_1 x_2 | x_1 y_2 x_2 | \dots | x_1 y_n x_2$$

Then, $L(G) = L(G')$.

Example:

$$S \rightarrow aBa$$

$$B \rightarrow aS \mid a$$

becomes

$S \rightarrow aaSa \mid aaa$
 $B \rightarrow aS \mid a$

useless rules

Definition: A production of the form $A \rightarrow Ax$, $A \in V$, $x \in (V \cup T)^*$ is *left recursive*.

Example Previous expression grammar was left recursive.

$$\mathbf{E} \rightarrow \mathbf{E} + \mathbf{T} \mid \mathbf{T}$$

$$\mathbf{T} \rightarrow \mathbf{T} * \mathbf{F} \mid \mathbf{F}$$

$$\mathbf{F} \rightarrow \mathbf{I} \mid (\mathbf{E})$$

$$\mathbf{I} \rightarrow \mathbf{a} \mid \mathbf{b}$$

Derivation of $\mathbf{a} + \mathbf{b} + \mathbf{a} + \mathbf{a}$ is:

$$\mathbf{E} \Rightarrow \mathbf{E} + \mathbf{T} \Rightarrow \mathbf{E} + \mathbf{T} + \mathbf{T} \Rightarrow \mathbf{E} + \mathbf{T} + \mathbf{T} + \mathbf{T}$$

$$\xRightarrow{*} \mathbf{a} + \mathbf{T} + \mathbf{T} + \mathbf{T}$$



Theorem (Removing Left recursion)

Let $G=(V,T,S,P)$ be a CFG. Divide productions for variable A into left-recursive and non left-recursive productions:

$$\begin{aligned} A &\rightarrow Ax_1 \mid Ax_2 \mid \dots \mid Ax_n \\ A &\rightarrow y_1 \mid y_2 \mid \dots \mid y_m \end{aligned}$$

where x_i, y_i are in $(V \cup T)^*$.

Then $G'=(V \cup \{Z\}, T, S, P')$ and P' replaces rules of form above by

$$\begin{aligned} A &\rightarrow y_i \mid y_i Z, \quad i=1,2,\dots,m \\ Z &\rightarrow x_i \mid x_i Z, \quad i=1,2,\dots,n \end{aligned}$$

Left recur

$y_1 \dots x_3 x_2 x_1$

$y_1 \dots x_3 x_2 x_1$

right recurs

Example:

$$E \rightarrow E+T \mid T$$

becomes

$$E \rightarrow T \mid TZ$$

$$Z \rightarrow +T \mid +TZ$$

$$T \rightarrow T*F \mid F$$

becomes

$$T \rightarrow F \mid FY$$

$$Y \rightarrow *F \mid *FY$$

other
rules

$$F \rightarrow I$$

$$I \rightarrow a \mid b$$

Now, Derivation of a+b+a+a is:

$$E \rightarrow TZ \rightarrow FZ \rightarrow IZ \rightarrow aZ$$

$$\rightarrow a+.TZ$$

we didn't see any
other terminals

Useless productions

$S \rightarrow aB \mid bA$

$A \rightarrow aA$

$B \rightarrow Sa$

$C \rightarrow cBc \mid a$

everything
useless
need to
get C

What can you say about this grammar?

useless variables
useless rules

Theorem (useless productions) Let G be a CFG. Then $\exists G'$ that does not contain any useless variables or productions s.t. $L(G)=L(G')$.

To Remove Useless Productions:

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

I. Compute $V_1=\{\text{Variables that can derive strings of terminals}\}$

1. $V_1=\emptyset$

2. Repeat until no more variables added

- For every $A \in V$ with $A \rightarrow x_1x_2 \dots x_n, x_i \in (T^* \cup V_1)$, add A to V_1

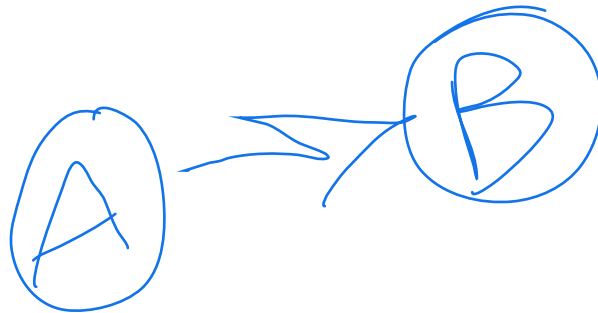
3. $P_1 =$ all productions in P with symbols in $(V_1 \cup T)^*$

Then $G_1=(V_1,T,S,P_1)$ has no variables that can't derive strings.

II. Draw Variable Dependency Graph

For $A \rightarrow xBy$, draw $A \rightarrow B$.

Remove productions for V if there is no path from S to V in the dependency graph. Resulting Grammar G' is s.t. $L(G)=L(G')$ and G' has no useless productions.



Example:

$S \rightarrow aB \mid \cancel{bA}$

$\cancel{A \rightarrow aA}$

$B \rightarrow Sa \mid b$

$C \rightarrow cBc \mid a$

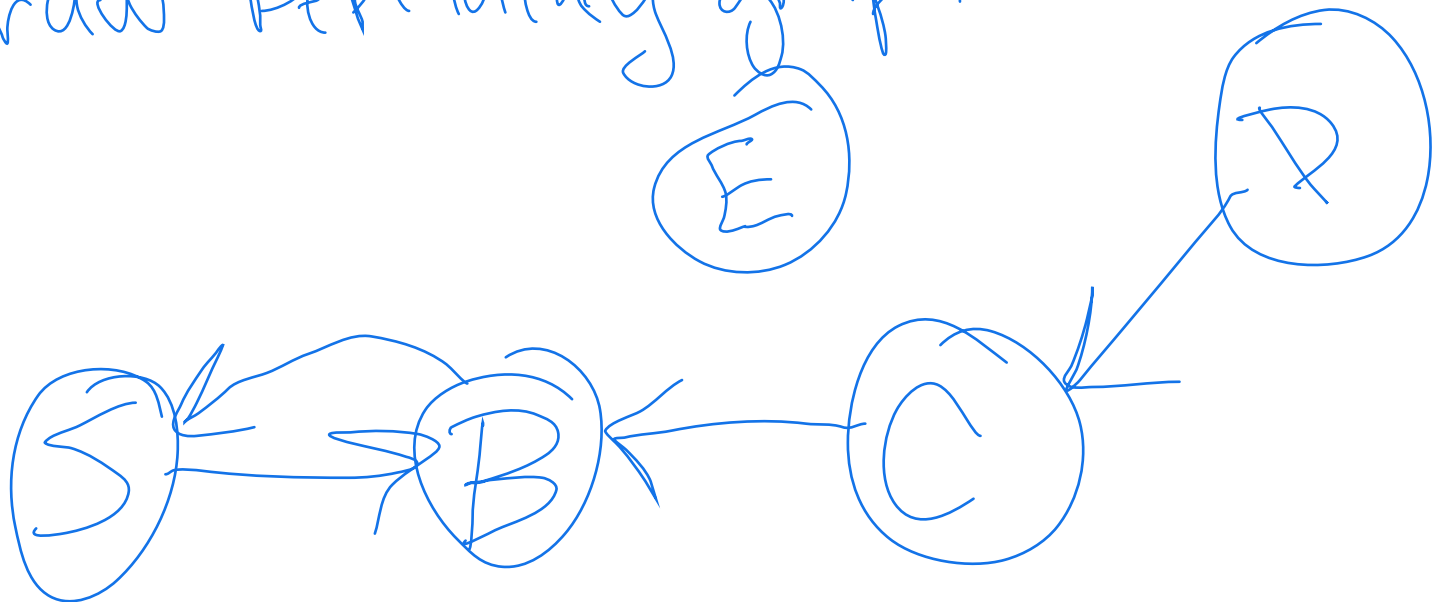
$D \rightarrow bCb$

$E \rightarrow \cancel{Aa} \mid b$

$S \rightarrow aB$
 $B \rightarrow Sa \mid b$

$V_1 = \{B, C, E, D, S\}$ not A

Draw Dependency graph



From S, can reach B

Theorem (remove λ productions) Let G be a CFG with λ not in $L(G)$. Then \exists a CFG G' having no λ -productions s.t. $L(G)=L(G')$.

To Remove λ -productions

1. Let $V_n = \{A \mid \exists \text{ production } A \rightarrow \lambda \}$
2. Repeat until no more additions
 - if $B \rightarrow A_1 A_2 \dots A_m$ and $A_i \in V_n$ for all i , then put B in V_n
3. Construct G' with productions P' s.t.
 - If $A \rightarrow x_1 x_2 \dots x_m \in P$, $m \geq 1$, then put all productions formed when x_j is replaced by λ (for all $x_j \in V_n$) s.t. $|\text{rhs}| \geq 1$ into P' .

Example:

$$S \rightarrow Ab$$

$$A \rightarrow BCB \mid Aa$$

$$B \rightarrow b \mid \lambda$$

$$C \rightarrow cC \mid \lambda$$

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

$$S \rightarrow Ab \mid b$$

$$A \rightarrow BCB \mid BC \mid CB \mid BB \mid B \mid C$$

$$A \rightarrow Aa \mid a$$

$$B \rightarrow b$$

$$C \rightarrow cC \mid c$$

Definition Unit Production

$$A \rightarrow B$$

where $A, B \in V$.

Consider removing unit productions:

Suppose we have

$A \rightarrow B$ becomes

$$A \rightarrow a | ab$$

$B \rightarrow a | ab$

But what if we have

$A \rightarrow B$ becomes

$B \rightarrow C$

$C \rightarrow A$

$$\begin{array}{l} A \rightarrow C \\ B \rightarrow A \\ C \rightarrow B \end{array} \quad Ah!$$

Stopped here

Theorem (Remove unit productions)
 Let $G=(V,T,S,P)$ be a CFG without λ -productions. Then \exists CFG $G'=(V',T',S,P')$ that does not have any unit-productions and $L(G)=L(G')$.

To Remove Unit Productions:

1. Find for each A, all B s.t. $A \xRightarrow{*} B$
 (Draw a dependency graph)
2. Construct $G'=(V',T',S,P')$ by
 - (a) Put all non-unit productions in P'
 - (b) For all $A \xRightarrow{*} B$ s.t. $B \rightarrow y_1|y_2|\dots|y_n \in P'$, put $A \rightarrow y_1|y_2|\dots|y_n \in P'$

Example:

S \rightarrow **AB**

A \rightarrow **B**

B \rightarrow **C** | **Bb**

C \rightarrow **A** | **c** | **Da**

D \rightarrow **A**

Theorem Let L be a CFL that does not contain λ . Then \exists a CFG for L that does not have any useless productions, λ -productions, or unit-productions.

Proof

1. Remove λ -productions
2. Remove unit-productions
3. Remove useless productions

Note order is very important.
Removing λ -productions can create unit-productions! QED.

Definition: A CFG is in Chomsky Normal Form (CNF) if all productions are of the form

$$A \rightarrow BC \quad \text{or} \quad A \rightarrow a$$

where $A, B, C \in V$ and $a \in T$.

Theorem: Any CFG G with λ not in $L(G)$ has an equivalent grammar in CNF.

Proof:

1. Remove λ -productions, unit productions, and useless productions.
2. For every rhs of length > 1 , replace each terminal x_i by a new variable C_j and add the production $C_j \rightarrow x_i$.
3. Replace every rhs of length > 2 by a series of productions, each with rhs of length 2. QED.

Example:

$S \rightarrow CBcd$

$B \rightarrow b$

$C \rightarrow Cc \mid e$

Definition: A CFG is in Greibach normal form (GNF) if all productions have the form

$$A \rightarrow ax$$

where $a \in T$ and $x \in V^*$

Theorem For every CFG G with λ not in $L(G)$, \exists a grammar in GNF.

Proof:

1. Rewrite grammar in CNF.
2. Relabel Variables A_1, A_2, \dots, A_n

3. Eliminate left recursion and use substitution to get all productions into the form:

$$\begin{aligned}A_i &\rightarrow A_j x_j, j > i \\Z_i &\rightarrow A_j x_j, j \leq n \\A_i &\rightarrow \mathbf{a}x_i\end{aligned}$$

where $\mathbf{a} \in \mathbf{T}$, $x_i \in \mathbf{V}^*$, and Z_i are new variables introduced for left recursion.

4. All productions with A_n are in the correct form, $A_n \rightarrow \mathbf{a}x_n$. Use these productions as substitutions to get A_{n-1} productions in the correct form. Repeat with A_{n-2} , A_{n-3} , etc until all productions are in the correct form.