CompSci 334 - Mathematical Foundations of CS
Dr. S. Rodger
Section: Transforming Grammars (Ch. 6) (handout)

Methods for Transforming Grammars (Read Ch 6 in Linz Book)

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

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

Then can construct G' from G by deleting

$$A \rightarrow x_1 B x_2$$

from P and adding to it

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

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

Example:

S → aBa   becomes
B → aS | a

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

Example Previous expression grammar was left recursive.

$$
\begin{array}{l}
\mathrm{E} \rightarrow \mathrm{E{+}T} \mid \mathrm{T} \\
\mathrm{T} \rightarrow \mathrm{T{*}F} \mid \mathrm{F} \\
\mathrm{F} \rightarrow \mathrm{I} \mid (\mathrm{E}) \\
\mathrm{I} \rightarrow \mathrm{a} \mid \mathrm{b}
\end{array}
$$

A top-down parser would want to derive the leftmost terminal as soon as possible. But in the left recursive grammar above, in order to derive a sentential form that has the leftmost terminal, we have to derive a sentential form that has other terminals in it.

Derivation of a+b+a+a is:

$\mathrm{E} \Rightarrow \mathrm{E{+}T} \Rightarrow \mathrm{E{+}T{+}T} \Rightarrow \mathrm{E{+}T{+}T{+}T} \stackrel{*}{\Rightarrow} \mathrm{a{+}T{+}T{+}T}$

We will eliminate the left recursion so that we can derive a sentential form with the leftmost terminal and no other terminals.

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:

$$
\begin{array}{l}
\mathrm{A} \rightarrow \mathrm{A}x_1 \mid \mathrm{A}x_2 \mid \ldots \mid \mathrm{A}x_n \\
\mathrm{A} \rightarrow y_1|y_2|\ldots|y_m
\end{array}
$$

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

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

$$
\begin{array}{l}
\mathrm{A} \rightarrow y_i|y_i\mathrm{Z},\ \mathrm{i}=1,2,\ldots,\mathrm{m} \\
\mathrm{Z} \rightarrow x_i|x_i\mathrm{Z},\ \mathrm{i}=1,2,\ldots,\mathrm{n}
\end{array}
$$

Example:

$\mathrm{E} \rightarrow \mathrm{E{+}T|T}$ becomes

$\mathrm{T} \rightarrow \mathrm{T{*}F|F}$ becomes

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

Useless productions

S → aB | bA
A → aA
B → Sa
C → cBc | a

What can you say about this grammar?

Theorem (useless productions) Let G be a CFG. Then ∃ 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={Variables that can derive strings of terminals}

1. $V_1=\emptyset$
2. Repeat until no more variables added
   - For every A∈V with $A \rightarrow x_1 x_2 \ldots x_n$, $x_i \in (T^* \cup V_1)$, 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 → xBy, draw A→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 → aB | bA
A → aA
B → Sa | b
C → cBc | a
D → bCb
E → Aa | 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 | $\exists$ production A$\rightarrow\lambda$ $\}$
2. Repeat until no more additions
   - if B$\rightarrow$A$_1$A$_2$...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$\rightarrow$ $x_1 x_2 \ldots x_m$ $\in$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. |rhs| ≥ 1 into P'.

Example:

S $\rightarrow$ Ab
A $\rightarrow$ BCB | Aa
B $\rightarrow$ b | λ
C $\rightarrow$ cC | λ

Definition Unit Production

$$A \rightarrow B$$

where A,B $\in$V.

Consider removing unit productions:

Suppose we have

A $\rightarrow$ B   becomes
B $\rightarrow$ a | ab

But what if we have

A $\rightarrow$ B   becomes
B $\rightarrow$ C
C $\rightarrow$ A

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 $\overset{*}{\Rightarrow}$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$\overset{*}{\Rightarrow}$B s.t. B$\rightarrow y_1|y_2|\ldots y_n \in$ P', put A$\rightarrow y_1|y_2|\ldots y_n \in$ P'

Example:

S → AB
A → B
B → C | Bb
C → A | c | Da
D → 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

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

where A,B,C∈V and a∈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 → CBcd
B → b
C → Cc | e

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

$$A \rightarrow ax$$

where a∈T and x∈V^*

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

Proof:

1. Rewrite grammar in CNF.
2. Relabel Variables A_1, A_2, … A_n
3. Eliminate left recursion and use substitution to get all productions into the form:

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

   where a∈T, x_i ∈V^*, and Z_i are new variables introduced for left recursion.

4. All productions with A_n are in the correct form, $A_n \rightarrow \mathrm{a}x_n$. 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.