CS164 - Left-recursion Elimination example (Weimer)

Consider the grammar:

$$\begin{array}{rcl} A & \to & B \,|\, a \,|\, CBD \\ B & \to & C \,|\, b \\ C & \to & A \,|\, c \\ D & \to & d \end{array}$$

Some strings in the language of this grammar are: a, cbd, etc. Notice that this grammar is not **immediately** left-recursive in that there is no single production $X \to X\alpha$. However, it is left-recursive because there are valid derivations of the form $A \to^* A\alpha$ (and $B \to^* B\beta$ and $C \to^* C\delta$). Let's demonstrate one: $A \to B \to C \to A$, so $A \to^* A$.

To warm up, let's compute $First()$ and $Follow()$ sets for this grammar.

$First(A)$ must contain $First(B)$, $\{a\}$ and $First(CBD)$. $First(B)$ must contain $First(C)$ and $\{b\}$. $First(C)$ must contain $First(A)$ and $\{c\}$. $First(CBD) = First(C)$ as before. So $First(A) = \{a, b, c\}$. Similarly, $First(B) = First(C) \bigcup \{b\} = \{a, b, c\}$ and $First(C) = First(A) \bigcup \{c\} = \{a, b, c\}$. $First(D) = \{d\}$.

To compute $Follow(A)$ we look for every occurrence of A on the right-hand side of a production. We find one in $C \to A$, but it that A is at the direct end of the production, we get that $Follow(A)$ includes $Follow(C)$. Now we look for C on the right-hand side of a production and find it in $A \to CBD$. This time there is something to the right of C, so we get that $Follow(C)$ contains $First(B)$. However, we also see a C on the right of $B \to C$, so $Follow(C)$ contains $Follow(B)$. Looking for B's on the right we find $A \to CBD$, so $Follow(B)$ contains $First(D) = \{d\}$l. So $Follow(A) = Follow(B) = Follow(C) = \{a, b, c, d\}$. Since D appears in the production $A \to CBD$, we have that $Follow(D)$ includes $Follow(A)$, so $Follow(D) = \{a, b, c, d\}$ as well.

OK, messy grammar. Now let's eliminate left-recursion. The first step is to make all left-recursion **immediate** by doing some substitutions. For example, since we have $A \to B \to C \to A$, we need to take the production $A \to B$ and replace it with $A \to C$ and $A \to b$. That gives us:

$$A \to C \,|\, b \,|\, a \,|\, CBD$$

But we're not done, since we can have $A \to C \to A$. So now we need to substitute in for C in that production. Let's do that once:

$$A \to A \,|\, c \,|\, b \,|\, a \,|\, CBD$$

To get here, we just removed the production $A \to C$ and replaced it by $A \to A$ and $A \to c$ (we got those two right-hand sides from the productions $C \to A$ and $C \to c$). Now we're almost done, just one more possible non-immediate left-recursion. Let's substitute it away:

$$A \to A \,|\, c \,|\, b \,|\, a \,|\, ABD \,|\, cBD$$

Huzzah! Now A has only immediate left-recursion. And actually, there is no other left-recursion left in the grammar now, since we can no longer derive $B \to^* B\beta$ or $C \to^* C\delta$. So we can leave the $B \to$ and $C \to$ and $D \to$ productions alone and concentrate on eliminating left-recursion from A.

First, let's group the productions into those that are left-recursive and those that are not:

$$\begin{array}{rcl} A & \to & A \,|\, ABD \\ & | & c \,|\, b \,|\, a \,|\, cBD \end{array}$$

Now imagine that you actually have this grammar before you. You can expand things for a long time by just using the first to productions: $A \to ABD \to ABDBD \to ABDBD$, etc. But eventually you have to settle down and use one of the other productions: $A \rightarrow ABD \rightarrow ABDBD \rightarrow cBDBD$ and then the chain stops. So we get the idea that A can eventually produce something that starts with c, b, a or cBD and ends with a list of BD's.

One other thing to note is that the production $A \rightarrow A$ itself is useless – it does not change the language of the grammar and can be safely dropped. So here's the revised left-recursive grammar:

$$\begin{array}{rcl} A & \rightarrow & ABD \\ & | & c \,|\, b \,|\, a \,|\, cBD \end{array}$$

Now let's break that down into A and A'. We reasoned above that an A goes to $c|b|a|cBD$ followed by a list of BD's. Let's make the first bit the A and make the list of BD's the A'.

$$\begin{array}{rcl} A & \rightarrow & cA' \,|\, bA' \,|\, aA' \,|\, cBDA' \\ A' & \rightarrow & \varepsilon \,|\, BDA' \end{array}$$

We're done. You can check and see that every production for A is of the form $A \rightarrow \alpha A'$ and that A' really does define a (possibly empty) list of BD's. Let's do one more example, just for fun. Consider the grammar:

$$\begin{array}{rcl} Q & \rightarrow & QED \,|\, q \\ E & \rightarrow & e \\ D & \rightarrow & NFA \,|\, d \\ N & \rightarrow & DFA \,|\, n \\ F & \rightarrow & f \\ A & \rightarrow & a \end{array}$$

This grammar is left recursive. In fact, it is immediately left-recursive in one place and non-immediately left-recursive in two places. First, let's substitute to get rid of the non-immediate left recursion. Consider the derivation: $D \rightarrow NFA \rightarrow DFAFA$. Since it ends up being left recursive, we must substitute. Take the production $D \rightarrow NFA$ and remove it. Then for every production $N \rightarrow \alpha_i$, add a production $D \rightarrow \alpha_i FA$. That gives us:

$$D \rightarrow DFAFA \,|\, nFA \,|\, d$$

Notice again that in one fell swoop we have eliminated the whole chain of non-immediate left-recursion: we can no longer derive $N \rightarrow^* N\beta$. So now our grammar looks like:

$$\begin{array}{rcl} Q & \rightarrow & QED \,|\, q \\ E & \rightarrow & e \\ D & \rightarrow & DFAFA \,|\, nFA \,|\, d \\ N & \rightarrow & DFA \,|\, n \\ F & \rightarrow & f \\ A & \rightarrow & a \end{array}$$

Now it's time to eliminate the immediate left recursion. Let's start with $Q \rightarrow QED|q$. Once again, we can derive strings like $Q \rightarrow QEDEDED$, but eventually we have to stop and use $Q \rightarrow q$. Taking Q to be the q bit and Q' to be the list of ED's, we get:

$$\begin{array}{rcl} Q & \rightarrow & qQ' \\ Q' & \rightarrow & \varepsilon \,|\, EDQ' \end{array}$$

Now let's look at $D \rightarrow DFAFA|nFA|d$. We see that we can make a huge list of $FAFA$' s using the first production but we eventually have to start with nFA or d. Let's make $D^{'}$ the list and D the first bit.

$$\begin{aligned} D &\rightarrow nFAD^{'} \,|\, dD^{'} \\ D^{'} &\rightarrow \varepsilon \,|\, FAFAD^{'} \end{aligned}$$

OK, that's all the left-recursion. The final grammar is:

$$\begin{aligned} Q &\rightarrow qQ^{'} \\ Q^{'} &\rightarrow \varepsilon \,|\, EDQ^{'} \\ E &\rightarrow e \\ D &\rightarrow nFAD^{'} \,|\, dD^{'} \\ D^{'} &\rightarrow \varepsilon \,|\, FAFAD^{'} \\ N &\rightarrow DFA \,|\, n \\ F &\rightarrow f \\ A &\rightarrow a \end{aligned}$$

Huzzah, we are done.