Assignment 3 SOLUTIONS **SE4I03 Fall 2002** **due on Oct. 30**

1. Find CFGs for the following languages:

   (a) $L_1 = \{w | \ w \text{ is } \textit{not} \text{ a palindrome } \}$

   If we only had rules $S \rightarrow aSa|bSb$, $S \rightarrow \varepsilon|a|b$, then we would only produce palindromes. What we want, is to interrupt this production at some point, with a string whose first and last symbols are not the same. Let D denote such a string, so D is defined as follows: $D \rightarrow aAb|bAa$, where A stands for any string at all. The "any string at all" can be given with $A \rightarrow \varepsilon|aA|bA$. Putting everything together, we have:

$$S \rightarrow aSa|bSb|D$$
$$D \rightarrow aAb|bAa$$
$$A \rightarrow \varepsilon|aA|bA$$

   (b) $L_2 = \{w | \text{ the nr. of 0s and 1s of } w \text{ is } \textit{not} \text{ the same } \}$

   In this question it is important to realize that $L_2 = L_{0>1} \cup L_{0<1}$, where $L_{0>1}$ denotes the set of strings with more 0s than 1s, and $L_{0<1}$ denotes the set of strings with more 1s than 0s.

   We first construct the CFG for $L_{0>1}$. Clearly $0 \in L_{0>1}$, and if $x \in L_{0>1}$, then $x0, 0x \in L_{0>1}$. This suggests the production $S \rightarrow 0|S0|0S$.

   We also need to be able to add 1s to our strings. But here we have to be careful; it is *not* true that if $x \in L_{0>1}$, then $x1, 1x \in L_{0>1}$! However, what *is* true, is that if $x \in L_{0>1}$, then so is $1xx, x1x, xx1 \in L_{0>1}$. Thus, $L_{0>1}$ can be given by:

$$S \rightarrow 0|S0|0S|1SS|S1S|SS1$$

   By the above arguments, we know that S produces only strings which are in $L_{0>1}$. How do we know that all the strings in $L_{0>1}$ can be produced with the above rule? We argue by induction on the length of the string:

   BC: if $|x| = 1$, then $x = 0$, so use $S \rightarrow 0$.

   IS: suppose the claim holds for all x, $|x| \leq n$ (i.e., if $x \in L_{0>1}$, and $|x| \leq n$, then x can be produced with the above rule). Let y be any string of length $n+1$ in $L_{0>1}$. If $y = 0^{n+1}$, then the above rule clearly produces y. Assume that y has at least one 1, so $y = u1v$. There are now 3 cases: $u = \varepsilon$, $v = \varepsilon$, or neither. In each case, we show that $S \rightarrow 1SS$, $S \rightarrow SS1$, or $S \rightarrow S1S$, respectively, can generate y. Working out the cases is left as an exercise.

   Replacing 0s by 1s in the rule, we obtain a rule for $L_{0<1}$. Putting everything together, we have:

$$S \rightarrow A|B$$
$$A \rightarrow 0|A0|0A|1AA|A1A|AA1$$
$$B \rightarrow 1|B1|1B|0BB|B0B|BB0$$

2. The rules are:

$$
\begin{aligned}
E &\to (E \wedge E) \\
E &\to (E \vee E) \\
E &\to \neg E \\
E &\to (E) \\
E &\to I \\
I &\to xJ \\
I &\to T|F \\
J &\to 1|0|J0|J1
\end{aligned}
$$

To see why the language consisting of boolean expressions is not regular, note that $(^n T)^n$ is a regular expression, so we can use the Pumping Lemma on the initial segment $(^n$ to unbalance the parenthesis.