Assignment 1:
Grammars and Induction

15-312: Foundations of Programming Languages
Daniel Spoonhower (spoons+@cs.cmu.edu)

Out: Thursday, August 28, 2003
Due: Thursday, September 4, 2003 (1:30 pm)
50 points total

Welcome to 15-312! This assignment focuses on context-free grammars and inductive proofs. It is due September 4th at the start of lecture. You are encouraged, but not required, to typeset your answers; if you write them out by hand, write legibly. If I can’t read it, I can’t give credit for it.

Please make sure you understand the policy on collaboration; refer to http://www.cs.cmu.edu/~fp/courses/312/assignments.html

1 Grammars (35 points)

Consider the grammar $G$ over the alphabet $\Sigma = \{\text{int, list, } \rightarrow, (, )\}$ with nonterminals tycon and type:

\[
\text{tycon ::= int | tycon list | ( type)} \\
\text{type ::= tycon | type } \rightarrow \text{ type}
\]

Question 1.1 (5 points).

The first production for tycon can be written in rule notation as

\[
\text{int tycon}
\]

Write grammar $G$ in rule notation.
Grammar $G$ is flawed: it does not pin down the associativity of $\rightarrow$. For example, there are two different derivations of the string $\text{int} \rightarrow \text{int} \rightarrow \text{int}$. We can fix the ambiguity by changing the second production of type from

$$\text{type} ::= \text{type} \rightarrow \text{type}$$

to

$$\text{type} ::= \text{tycon} \rightarrow \text{type}$$

Making this change results in the grammar $G'$ below. To avoid confusion, we rename tycon to tycon' and type to type'.

$$\text{tycon'} ::= \text{int} | \text{tycon'} \text{list} | (\text{type'})$$

$$\text{type'} ::= \text{tycon'} | \text{tycon'} \rightarrow \text{type'}$$

**Question 1.2 (5 points).**

Write grammar $G'$ in rule notation.

Note, however, that we have not shown that this new grammar $G'$ is equivalent to $G$, that is, that the languages of type and type' are the same: $L(\text{type}) = L(\text{type'})$. This can be proved in two steps: first prove $L(\text{type'}) \subseteq L(\text{type})$, then prove $L(\text{type}) \subseteq L(\text{type'})$.

**Question 1.3 (10 points).**

Prove $L(\text{type'}) \subseteq L(\text{type})$ by proving

If $s \text{ type'}$ then $s \text{ type}$

by induction. If you need to generalize the induction hypothesis, be sure to clearly state your generalized induction hypothesis. If you need any lemmas, state them explicitly and prove them.

**Question 1.4 (15 points).**

Prove $L(\text{type}) \subseteq L(\text{type'})$ by proving

If $s \text{ type}$ then $s \text{ type'}$

As in the previous question, clearly state any generalized induction hypothesis and prove any lemmas you need.
2 Propositional logic (15 points)

In this question we will look at a subset of Propositional Logic. Our universe of terms consists of an infinite number of nullary operators $P_0, P_1, \ldots, P_n$ (“propositional variables”) and the binary operator $\Rightarrow$ (“implication”). We define the sets prop and thm over this universe:

\[
\begin{align*}
&P_1 \text{ prop} \quad \text{Var} \\
&\frac{A \text{ prop} \quad B \text{ prop}}{A \Rightarrow (B \Rightarrow A) \text{ thm} \quad K} \\
&\frac{A \text{ prop} \quad B \text{ prop}}{A \Rightarrow B \text{ prop} \quad \text{Imp}} \\
&\frac{A \text{ prop} \quad B \text{ prop} \quad C \text{ prop}}{(A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C) \text{ thm} \quad S} \\
&\frac{A \Rightarrow B \text{ thm} \quad A \text{ thm}}{B \text{ thm} \quad \text{App}}
\end{align*}
\]

Truth Value. If we have assignments (to true or false) for all of the propositional variables in a proposition, its truth value (either true or false) can be computed recursively using the following familiar truth table for $\Rightarrow$:

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Truth Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>false $\Rightarrow$ false</td>
<td>true</td>
</tr>
<tr>
<td>false $\Rightarrow$ true</td>
<td>true</td>
</tr>
<tr>
<td>true $\Rightarrow$ false</td>
<td>false</td>
</tr>
<tr>
<td>true $\Rightarrow$ true</td>
<td>true</td>
</tr>
</tbody>
</table>

Tautology. A proposition is a tautology iff for every assignment of truth values (true, false) to the propositional variables $P_0, \ldots, P_n$, the truth value of the proposition is true.

Question 2.1 (15 points).

Prove, using rule induction, that if $A$ thm then $A$ is a tautology.

Question 2.2 (EXTRA CREDIT).

Find a proposition $A$ that is a tautology, but not a theorem (that is, the judgment $A$ thm cannot be derived). You do not need to prove that it is not a theorem!