Exercise Compiler Construction (4)

Hans de Nivelle

Due: 4 November 2009

1. Consider the language (Σ, R, S), defined by $\Sigma = \{\ '(', ')'\ \}$, $R = \{S \rightarrow SS,\ \ S \rightarrow (S),\ \ S \rightarrow \epsilon$.

   - Give derivations for

     $$'(()()())',\ '((()))',\ '((()()))'.$$

   - Show that the language is ambiguous. (There are words that have more than one derivation.)
   - Repair the grammar, so that it is not ambiguous anymore, but still accepts the same set of words.

2. (a) In the programming language Lisp, everything is a list. The empty list has form ( ) or **nil**. Non-empty lists have form (L_1), $(L_1\ L_2)$, $(L_1\ L_2\ L_3)$, etc. Give a complete grammar for Lisp. The elements of a list can be atoms, numbers, or by themselves lists. (You may ignore the existence of dotted pairs and arrays.)

   (b) Give a grammar for the language of Prolog-style lists. Lists have form

   $$[\ ],\ [L],\ [L_1, L_2],\ [L_1, L_2, L_3],\ \text{etc.}$$

   The elements of the lists can by themselves be lists again.

   (c) Consider the language consisting of functional expressions of form c, and $f(t_1, \ldots, t_n)$, with $n > 0$, and $t_1, \ldots, t_n$ functional expressions by themselves. Give a grammar for this language.

   (d) Give a grammar for propositional formulas consisting of propositional variables, and the following operators:

operator	meaning	binding strength
!	negation	600, prefix
&	and	500, leftassoc
|	or	400, leftassoc
$\rightarrow$	implication	300, rightassoc
$\leftrightarrow$	equivalence	200, nonassoc.

   In addition to the operators, the language must allow parentheses.

3. (a) Give the attribute functions for the grammar of Task 2a. Take into account that $(L_1 \cdots L_n)$ denotes $\mathbf{cons}(L_1, \mathbf{cons}(L_2, \ldots \mathbf{nil}))$. Give a derivation for the list

   $$(\mathbf{car}\ (\mathbf{quote}\ (1\ 2\ 3))).$$

   (b) Give the attribute functions for the grammar in Task 2d.