Problems

These problems will be discussed at seminar 4. Programs in Haskell are not required to compile. The exercises in Nielson have been renumbered. My references are to the 1999 edition.

---

1 Use the unification algorithm to unify `append(cons(1, empty), cons(2, cons(3, empty)), L)` with `append(cons(X, Xs), Ys, cons(X, Zs))`. Indicate the arguments to `unify` at each recursive call and the value of the substitution σ after each call.

2 Use the resolution algorithm to find a proof of `male(oskar)` using the initial example in the lecture notes on Prolog.

3 Consider replacing the first rule in Table 2.4 with

$$\frac{\langle D_V, \sigma\rangle \rightarrow_D \sigma'}{\langle \texttt{var x := a}; D_V, \sigma\rangle \rightarrow_D \sigma'[x \mapsto \mathcal{A}[\![\texttt{a}]\!]\sigma]}$$

Would this change the semantics? Either prove that it doesn't or exhibit an example where the semantics differ.

4 (Exercise 2.34) We shall extend **While** with the statement `random(`x`)` and the idea is that its execution will change the value of x to be any positive natural number. Extend the natural semantics and the structural operational semantics to express this. Discuss whether this statement is superfluous when **While** is also extended with the `or` statement.

5 (Exercise 2.35) Consider an extension of **While** that in addition to the `par` statement also contains the construct `protect` S `end`. The idea is that the statement S has to be executed as an atomic entity so that, for example, `x:=1 par protect (x:=2; x:=x+2) end` only has two possible outcomes, **1** and **4**. Extend the structural operational semantics to express this.

6 (Exercise 2.45) Modify the syntax of `While` with blocks and procedures so that procedures take two *call by value* parameters, i.e. the kind of parameters used by Java for the primitive types.

$D_P ::=$ `proc` $p(x_1, x_2)$ `is` $S; D_P \mid \epsilon$

$S ::= \ldots \mid$ `call` $p(a_1, a_2)$

Procedure environments will now be elements of

$$\mathbf{Env}_P = \mathbf{Pname} \hookrightarrow (\mathbf{Var} \times \mathbf{Var} \times \mathbf{Stm} \times \mathbf{Env}_V \times \mathbf{Env}_P)$$

Use static name scopes. You may assume that procedures are not recursive. You have to provide new rules for procedure declarations and procedure calls.