COMP 523: Language-based security
Assignment 4 (100 points total)

Prof. B. Pientka
McGill University

Due: Wednesday, Oct 6, 2010 at 2:35pm

1 Implementing proofs in Beluga (40 points)

(45pts): In HW 2, we extend the language for booleans and arithmetic expressions we have seen in class (see also Ch 3, CH 8 in Pierce) with an expression `leq` $t\ t'$ which allows us to check whether `t` is less than or equal to `t'`, and we proved that the rules were deterministic, that types were preserved and that we have progress.

10 points Implement small-step evaluation rules for `leq` $t\ t'$ in Beluga; extend the representation of the small-step rules in `small-step.bel`.

10 points Implement the proof of determinacy for the cases covering `leq` (see file `det.bel`).

20 points Add the typing rule for `leq` $t\ t'$ and implement the progress and type preservation proof (see file `tps.bel` for preservation proof and see file `progress.bel` for progress proof).).

2 Case-statement(60 points)

An alternative definition for numbers is as follows:

$$\begin{array}{lcl} \text{Terms } t & ::= & x \mid \mathsf{z} \mid \mathsf{succ}\ t \mid (\mathsf{case}\ t\ \mathsf{of}\ \mathsf{z} \Rightarrow t_1 \mid \mathsf{succ}\ x \Rightarrow t_2) \\ \text{Types } T & ::= & \text{NAT} \end{array}$$

Here we can analyze numbers using a case-expression where we pattern match against the possible shapes of numbers. So, if the subject t of the case-expression $\mathsf{case}\ t\ \mathsf{of}\ \mathsf{z} \Rightarrow t_1 \mid \mathsf{succ}\ x \Rightarrow t_2$ evaluates to z then we choose the first branch t_1. Otherwise t must evaluate to some value of the form $\mathsf{succ}\ v$. In this case we match $\mathsf{succ}\ x$ against $\mathsf{succ}\ v$ which will yield the instantiation of x to v. We then proceed to evaluate the second branch t_2 under this instantiation by applying the substitution $[v/x]$ to t_2. The evaluation for these terms can be then defined as follows:

$$\frac{}{\mathsf{z} \Downarrow \mathsf{z}} \qquad \frac{t \Downarrow v}{\mathsf{succ}\ t \Downarrow \mathsf{succ}\ v}$$

$$\frac{t \Downarrow \mathsf{z} \quad t_1 \Downarrow v}{\mathsf{case}\ t\ \mathsf{of}\ \mathsf{z} \Rightarrow t_1 \mid \mathsf{succ}\ x \Rightarrow t_2 \Downarrow v} \qquad \frac{t \Downarrow \mathsf{succ}\ v_2 \quad [v_2/x]t_2 \Downarrow v}{\mathsf{case}\ t\ \mathsf{of}\ \mathsf{z} \Rightarrow t_1 \mid \mathsf{succ}\ x \Rightarrow t_2 \Downarrow v}$$

1. (10pts) Assuming we also have functions, function application, and booleans, show how we can define functions for predecessor and iszero as abbreviations.

2. (10pts) Define the appropriate typing rule for the case-expression.

3. (10pts) Show that type preservation holds for this rule.

4. (15pts) Give the corresponding small-step evaluation rules.

5. (15 pts) Show progress holds for the small step semantics you propose.