Elements of Programming Languages
Tutorial 7: Small-step semantics and type soundness
Week 9 (November 14–18, 2016)

Exercises marked $\star$ are more advanced. Please try all unstarred exercises before the tutorial meeting.

1. **Comparing large-step and small-step derivations**

   Write both large-step and small-step derivations for the following expressions. For the small-step derivations, construct the derivations of each $e \mapsto e'$ step explicitly.

   (a) $(\lambda x.x + 1)\ 42$

   (b) $(\lambda x.\texttt{if}\ x == 1\ \texttt{then}\ 2\ \texttt{else}\ x + 1)\ 42$

2. **Small-step derivations that go wrong**

   For each of the following expressions, show the small-step evaluation leading to the point where evaluation becomes stuck due to a dynamic type error. (There is no need to show the derivations of each step.)

   (a) $((\lambda x.\lambda y.\texttt{let}\ z = x + y\ \texttt{in}\ z + 1)\ 42)\ \texttt{true}$

   (b) $(\lambda x.\texttt{if}\ x\ \texttt{then}\ x + 1\ \texttt{else}\ x + 2)\ \texttt{true}$

3. **Small-step rules for $\mathsf{L}_{\mathsf{Data}}$**

   Recall that we defined the semantics for $\mathsf{L}_{\mathsf{Data}}$ using big-step rules, as follows:

   $\boxed{e \Downarrow v}$

   $$\frac{e_1 \Downarrow v_1 \quad e_2 \Downarrow v_2}{(e_1, e_2) \Downarrow (v_1, v_2)} \qquad \frac{e \Downarrow (v_1, v_2)}{\texttt{fst}\ e \Downarrow v_1} \qquad \frac{e \Downarrow (v_1, v_2)}{\texttt{snd}\ e \Downarrow v_2}$$

   $$\frac{e \Downarrow v}{\texttt{left}(e) \Downarrow \texttt{left}(v)} \qquad \frac{e \Downarrow \texttt{left}(v_1) \quad e_1[v_1/x] \Downarrow v}{\texttt{case}\ e\ \texttt{of}\ \{\texttt{left}(x) \Rightarrow e_1\ ;\ \texttt{right}(y) \Rightarrow e_2\} \Downarrow v}$$

   $$\frac{e \Downarrow v}{\texttt{right}(e) \Downarrow \texttt{right}(v)} \qquad \frac{e \Downarrow \texttt{right}(v_2) \quad e_2[v_2/x] \Downarrow v}{\texttt{case}\ e\ \texttt{of}\ \{\texttt{left}(x) \Rightarrow e_1\ ;\ \texttt{right}(y) \Rightarrow e_2\} \Downarrow v}$$

   (a) For each construct, write out equivalent small-step rules. Are there any design choices in translating the big-step rules to small-step rules?

   (b) ($\star$) Construct small-step derivations reducing the following expressions to values:

      i. $(\lambda p.(\texttt{snd}\ p, \texttt{fst}\ p + 2))\ (17, 42)$

      ii. $(\lambda x.\texttt{case}\ x\ \texttt{of}\ \{\texttt{left}(y).y + 1\ ;\ \texttt{right}(z).\ z\})\ (\texttt{left}(42))$

4. ($\star$) **Type soundness for nondeterminism**

   This question builds on the *nondeterministic choice* construct mentioned in an earlier tutorial, with the following typing rules:

   $\boxed{\Gamma \vdash e : \tau}$

   $$\frac{\Gamma \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1 \Box e_2 : \tau}$$

   and small-step evaluation rules:

   $\boxed{e \mapsto e'}$

   $$\frac{}{e_1 \Box e_2 \mapsto e_1} \qquad \frac{}{e_1 \Box e_2 \mapsto e_2}$$

   (a) State the *preservation* property. Outline how we could prove the cases of preservation for nondeterministic expressions.

   (b) State the *progress* property. Outline how we could prove the cases of progress for nondeterministic expressions.