Principles for software composition 2020/21
05 - HOFL

[**Ex. 1**] Determine the type of the HOFL term

$$t \stackrel{\text{def}}{=} \textbf{rec}\ x.\ ((\lambda y.\ \textbf{if}\ y\ \textbf{then}\ 0\ \textbf{else}\ 0)\ x).$$

Then compute its (lazy) canonical form.

[**Ex. 2**] Determine the type of the HOFL term

$$map \stackrel{\text{def}}{=} \lambda f.\ \lambda x.\ ((f\ \textbf{fst}(x)), (f\ \textbf{snd}(x)))$$

Then, compute the (lazy) canonical forms of the terms

$$t_1 \stackrel{\text{def}}{=} map\ (\lambda z.\ 2 \times z)\ (1, 2) \qquad\qquad t_2 \stackrel{\text{def}}{=} \textbf{fst}\ (\ map\ (\lambda z.\ 2 \times z)\ (1, 2)\ )$$

[**Ex. 3**] Let $(D, \sqsubseteq_D)$ be a CPO and $f : D \to D$ be a continuous function. Prove that the set of fixpoints of f is itself a CPO (ordered by $\sqsubseteq_D$).

[**Ex. 4**] (Test for convergence) We would like to modify the denotational semantics of HOFL assigning to the construct

$$\textbf{if}\ t\ \textbf{then}\ t_0\ \textbf{else}\ t_1$$

- the semantics of t_1 if the semantics of t is $\bot_{\mathbb{Z}_\bot}$, and
- the semantics of t_0 otherwise.

Is it possible? If not, why?

[**Ex. 5**] (Strict conditional) Modify the operational semantics of HOFL by taking the following rules for conditionals:

$$\frac{t \to 0 \quad t_0 \to c_0 \quad t_1 \to c_1}{\textbf{if}\ t\ \textbf{then}\ t_0\ \textbf{else}\ t_1 \ \to\ c_0} \qquad\qquad \frac{t \to n \quad n \neq 0 \quad t_0 \to c_0 \ \ t_1 \to c_1}{\textbf{if}\ t\ \textbf{then}\ t_0\ \textbf{else}\ t_1 \ \to\ c_1}.$$

Without changing the denotational semantics, prove that:

1. for any term t and canonical form c, we have $t \to c \ \Rightarrow\ \forall \rho.\ [\![t]\!]\, \rho = [\![c]\!]\, \rho$;
2. in general $t{\Downarrow} \not\Rightarrow t{\downarrow}$ (exhibit a counterexample).

[**Ex. 6**] Determine the type of the HOFL term

$$t \stackrel{\text{def}}{=} \textbf{rec}\ f.\ (\ \lambda x.1\ ,\ \textbf{fst}(f)\ 0\ )$$

Then, compute the (lazy) denotational semantics of t.