1 Real PCF

We give below the denotational and operational semantics for Real PCF. The types are as follows:

\[
\sigma, \tau, \ldots ::= \text{unit} \\
| I \\
| \sigma \to \tau
\]

\(\mathbb{S} = \{\bot, \top\}\) with \(\bot < \top\). \(\mathcal{I} = [0, 1]\) with the usual order.

\[
\llbracket \text{unit} \rrbracket = \mathbb{S} \quad \llbracket I \rrbracket = \mathcal{I} \quad \llbracket \sigma \to \tau \rrbracket = [\llbracket \sigma \rrbracket \to \llbracket \tau \rrbracket].
\]

\[
\llbracket x \rrbracket \rho = \rho(x) \\
\llbracket uv \rrbracket \rho = \llbracket u \rrbracket \rho(\llbracket v \rrbracket \rho)
\]

\([\text{fn } x_{\sigma}.u] \rho = (V \in [\sigma] \mapsto [u](\rho[x_{\sigma} \mapsto V]))\)

\[
\llbracket \text{letrec } x_{\sigma} = u \text{ in } v \rrbracket \rho = [v](\rho[x_{\sigma} \mapsto [\text{rec } x_{\sigma} = u] \rho])
\]

\[
\llbracket \text{rec } x_{\sigma} = u \rrbracket \rho = \text{lfp}(V \in [\sigma] \mapsto [u](\rho[x_{\sigma} \mapsto V]))
\]

\[
\llbracket 0.u \rrbracket \rho = \text{add}_0(\llbracket u \rrbracket \rho) \\
\llbracket t_1.0.u \rrbracket \rho = \text{rem}_0(\llbracket u \rrbracket \rho)
\]

\[
\llbracket \text{pif } u \text{ then } v \text{ else } w : \tau \rrbracket \rho = \begin{cases} 
\llbracket v \rrbracket \rho, & \text{if } \llbracket u \rrbracket \rho = \top \\
\llbracket v \rrbracket \rho \land \llbracket w \rrbracket \rho, & \text{if } \llbracket u \rrbracket \rho = \bot
\end{cases}
\]

\[
\llbracket u > 1/2 \rrbracket \rho = \begin{cases} 
\top, & \text{if } \llbracket u \rrbracket \rho > 1/2 \\
\bot, & \text{otherwise}
\end{cases}
\]

\[
\llbracket * \rrbracket \rho = \top
\]

where \(V \in X \mapsto f(V)\) denotes the function which to all \(V\) in \(X\) associates \(f(V)\), and where:

\[
\text{add}_0(a) = a/2 \quad \text{add}_1(a) = (a + 1)/2
\]

\[
\text{rem}_0(a) = \min(2a, 1) \quad \text{rem}_1(a) = \max(2a - 1, 0)
\]
Contexts (type constraints omitted):

\[ C ::= _ | Cv | t_1 \circ C | t_1 \circ C | t_1 \circ C | C > 1/2 | C > 0 | \text{pif } C \text{ then } v \text{ else } w | \text{pif } u \text{ then } C \text{ else } w | \text{pif } u \text{ then } v \text{ else } C \]

Operational semantics. We only apply a rule under a context \( C \) of the above form, i.e., \( u \rightarrow v \) if and only if \( u = C[\ell] \) and \( v = C[r] \), where \( C \) is a context (the types being respected), and \( \ell \rightarrow r \) is one of the rules below.

\[
\begin{align*}
\text{(fn } x_\sigma.u)v & \rightarrow u[x_\sigma := v] \\
\text{letrec } x_\sigma = u \text{ in } v & \rightarrow v[x_\sigma := \text{letrec } x_\sigma = u \text{ in } u] \\
\text{t}_{1_a}(a.u) & \rightarrow u \quad (a \in \{0,1\}) \\
\text{t}_{1_0}(1.u) & \rightarrow \bot \\
\text{t}_{1_1}(0.u) & \rightarrow 0 \\
d.1/u > 1/2 & \rightarrow u > 0 \\
d.1/u > 0 & \rightarrow * \\
d.0/u > 0 & \rightarrow u > 0 \\
\text{pif } * \text{ then } v \text{ else } w & \rightarrow v \\
\text{pif } u \text{ then } v \text{ else } * & \rightarrow v \\
\text{pif } u \text{ then } 0.v \text{ else } 1.w & \rightarrow 0.v \\
\text{pif } u \text{ then } a.v \text{ else } a.w & \rightarrow a.(\text{pif } u \text{ then } v \text{ else } w) \\
& \quad (a \in \{0,1\})
\end{align*}
\]

Exercise 1:

Recall that for all \( u : \tau, [u] \) is a well-defined function, Scott-continuous from \( Env \overset{\text{def}}{=} \prod_{x_{\text{variable}},[\sigma]} \) to \([\tau]\).

1. Show that the construction \( u > 0 \) of Real PCF is redundant. Explicitly propose a definition of an expression Real PCF \( \text{nonzero} \), of type \( I \rightarrow \text{unit} \), which does not use the expression of the form \( u > 0 \), and whose semantics \( [\text{nonzero}] \rho \) is the function to which \( 0 \) associates \( \bot \) and to all \( a \in I \) non-zero associates \( \top \). Prove this assertion.

2. Show that the rule tagged with \( (\alpha) \) of the operational semantics is correct, in the sense that \( [\text{pif } u \text{ then } v \text{ else } *]^{\rho} = [v]^{\rho} \) for all \( \rho \in Env \).

3. We consider a Real PCF program of the form \( \text{letrec } x_\sigma = u \text{ in } v \), of type \( \text{unit} \). Show that if \( [\text{letrec } x_\sigma = u \text{ in }]^{\rho} \neq \bot \), then there is an integer \( n \in \mathbb{N} \) such that

\[
[\text{letrec } x_\sigma = u \text{ in }]^{\rho} = g(f^n(\bot)),
\]

where we use the abbreviations \( g(V) = [v](\rho[x_\sigma \rightarrow V]) \) and \( f(V) = [u](\rho[x_\sigma \rightarrow V]) \).

(The \( \bot \) in argument of \( f^n \) is that of \([\sigma]\).) This expresses that a recursive definition (of \( x_\sigma \)) used in a terminating computation \( v \) of type \text{unit} will be "expanded" only \( n \) times.
4. Why does the argument from the previous question not work if \texttt{letrec } x_\sigma = u \texttt{ in } v \texttt{ is of type I?}

5. Recall that \( \hat{0} \overset{\text{def}}{=} \texttt{letrec } x_I = 0.x_I \texttt{ in } x_I \). Show that there does not exist a derivation in the operational semantics for

\[ t_1 (\texttt{pif } \hat{0} > 1/2 \texttt{ then } 1.\hat{0} \texttt{ else } 0.1.1.\hat{0}) > 1/2 \rightarrow^* \star. \]

We can set \( Z \overset{\text{def}}{=} \texttt{letrec } x_I = 0.x_I \texttt{ in } 0.x_I \).

6. What can we conclude for the adequacy of the type unit? Justify.

7. Any suggestions to complete the operational semantics?

**Exercise 2:**

We now assume that a same Real PCF\(^-\) variable is always labeled with the same type: if we see \( x_\sigma \) and \( x_\tau \), then \( \sigma = \tau \). This amounts to saying that the name \( x \) of the variables is sufficient to distinguish them.

We consider the Real PCF\(^-\) language, which is just Real PCF\(^-\) but without any type index. For example, \( \texttt{fn } x.u \) and \( \texttt{letrec } x = u \texttt{ in } v \) are the expressions Real PCF\(^-\) corresponding to \( \texttt{fn } x_\sigma.u \) and \( \texttt{letrec } x_\sigma = u \texttt{ in } v \), respectively.

Formally, let \( E \) denote the type erasure function, defined by

\[ E(\texttt{letrec } x_\sigma = u \texttt{ in } v) \overset{\text{def}}{=} \texttt{letrec } x = E(u) \texttt{ in } E(v), \quad E(\texttt{fn } x_\sigma.u) \overset{\text{def}}{=} \texttt{fn } x.E(u), \text{ etc.} \]

We will say that a Real PCF\(^-\) expression \( u \) is typable, of type \( \tau \), if and only if there exists a Real PCF\(^-\) expression \( u' \), of type \( \tau \), such that \( E(u) = u \).

1. Are all Real PCF\(^-\) expressions typable? Justify.

2. Is the type of a Real PCF\(^-\) typable expression unique? Justify.