# Programmation 1

## TD n°12

8 décembre 2020

# 1 Real PCF$^-$

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

$$\begin{aligned} \sigma, \tau, \ldots ::= & \ \mathtt{unit} \\ | & \ \mathtt{I} \\ | & \ \sigma \to \tau \end{aligned}$$

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

$$[\![\mathtt{unit}]\!] = \mathbb{S} \qquad [\![\mathtt{I}]\!] = \mathcal{I} \qquad [\![\sigma \to \tau]\!] = [[\![\sigma]\!] \to [\![\tau]\!]].$$

$$\begin{aligned}
[\![x_\tau]\!]\rho &= \rho(x_\tau) \\
[\![uv]\!]\rho &= [\![u]\!]\rho([\![v]\!]\rho) \\
[\![\mathtt{fn}\ x_\sigma.u]\!]\rho &= (V \in [\![\sigma]\!] \mapsto [\![u]\!](\rho[x_\sigma \mapsto V])) \\
[\![\mathtt{letrec}\ x_\sigma = u\ \mathtt{in}\ v]\!]\rho &= [\![v]\!](\rho[x_\sigma \mapsto [\![\mathtt{rec}\ x_\sigma = u]\!]\rho]) \\
[\![\mathtt{rec}\ x_\sigma = u]\!]\rho &= \mathrm{lfp}(V \in [\![\sigma]\!] \mapsto [\![u]\!](\rho[x_\sigma \mapsto V])) \\
[\![0.u]\!]\rho &= add_0([\![u]\!]\rho) & [\![1.u]\!]\rho = add_1([\![u]\!]\rho) \\
[\![\mathtt{tl}_0 u]\!]\rho &= rem_0([\![u]\!]\rho) & [\![\mathtt{tl}_1 u]\!]\rho = rem_1([\![u]\!]\rho) \\
[\![\mathtt{pif}\ u\ \mathtt{then}\ v\ \mathtt{else}\ w : \tau]\!]\rho &= \begin{cases} [\![v]\!]\rho, & \text{if } [\![u]\!]\rho = \top \\ [\![v]\!]\rho \wedge [\![w]\!]\rho, & \text{if } [\![u]\!]\rho = \bot \end{cases} \\
[\![u > \mathtt{1/2}]\!]\rho &= \begin{cases} \top, & \text{if } [\![u]\!]\rho > 1/2 \\ \bot, & \text{otherwise} \end{cases} & [\![u > \mathtt{0}]\!]\rho = \begin{cases} \top, & \text{if } [\![u]\!]\rho > 0 \\ \bot, & \text{otherwise} \end{cases} \\
[\![*]\!]\rho &= \top,
\end{aligned}$$

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

$$\begin{aligned} add_0(a) &= a/2 & add_1(a) &= (a+1)/2 \\ rem_0(a) &= \min(2a, 1) & rem_1(a) &= \max(2a-1, 0) \end{aligned}$$

Contexts (type constraints omitted) :

$$
\begin{array}{l}
\mathcal{C} ::= \_ \\
\quad \mid \mathcal{C} v \\
\quad \mid \texttt{tl}_0 \mathcal{C} \\
\quad \mid \texttt{tl}_0 \mathcal{C} \\
\quad \mid \texttt{tl}_1 \mathcal{C} \\
\quad \mid \mathcal{C} > \texttt{1/2} \\
\quad \mid \mathcal{C} > \texttt{0} \\
\quad \mid \texttt{pif}\ \mathcal{C}\ \texttt{then}\ v\ \texttt{else}\ w \\
\quad \mid \texttt{pif}\ u\ \texttt{then}\ \mathcal{C}\ \texttt{else}\ w \\
\quad \mid \texttt{pif}\ u\ \texttt{then}\ v\ \texttt{else}\ \mathcal{C}
\end{array}
$$

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

$$
\begin{array}{rll}
(\texttt{fn}\ x_\sigma.u)v & \to u[x_\sigma := v] & \\
\texttt{letrec}\ x_\sigma = u\ \texttt{in}\ v & \to v[x_\sigma := \texttt{letrec}\ x_\sigma = u\ \texttt{in}\ v] & \\
\texttt{tl}_a(a.u) & \to u & (a \in \{0,1\}) \\
\texttt{tl}_0(1.u) & \to \dot{1} & \\
\texttt{tl}_1(0.u) & \to \dot{0} & \\
(1.u) > \texttt{1/2} & \to u > 0 & \\
(1.u) > 0 & \to * & \\
(0.u) > 0 & \to u > 0 & \\
\texttt{pif}\ *\ \texttt{then}\ v\ \texttt{else}\ w & \to v & \\
\texttt{pif}\ u\ \texttt{then}\ v\ \texttt{else}\ * & \to v & (\alpha) \\
\texttt{pif}\ u\ \texttt{then}\ 0.v\ \texttt{else}\ 1.w & \to 0.v & \\
\texttt{pif}\ u\ \texttt{then}\ a.v\ \texttt{else}\ a.w & \to a.(\texttt{pif}\ u\ \texttt{then}\ v\ \texttt{else}\ w) & (a \in \{0,1\})
\end{array}
$$

**Exercise 1 :**

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

1. Show that the construction $u > 0$ of Real PCF$^-$ is redundant. Explicitly propose a definition of an expression Real PCF$^-$ **nonzero**, of type $\texttt{I} \to \texttt{unit}$, which does not use the expression of the form $u > 0$, and whose semantics $[\![\textbf{nonzero}]\!]\rho$ is the function to which 0 associates $\bot$ and to all $a \in \mathcal{I}$ 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 $[\![\texttt{pif}\ u\ \texttt{then}\ v\ \texttt{else}\ *]\!]\rho = [\![v]\!]\rho$ for all $\rho \in Env$.
3. We consider a Real PCF$^-$ program of the form $\texttt{letrec}\ x_\sigma = u\ \texttt{in}\ v$, of type `unit` . Show that if $[\![\texttt{letrec}\ x_\sigma = u\ \texttt{in}\ ]\!]\rho \neq \bot$,then there is an integer $n \in \mathbb{N}$ such that

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

where we use the abbreviations $g(V) = [\![v]\!](\rho[x_\sigma \mapsto V])$ and $f(V) = [\![u]\!](\rho[x_\sigma \mapsto V])$. (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 `unit` will be "expanded" only $n$ times .

4. Why does the argument from the previous question not work if $\texttt{letrec}\ x_\sigma = u\ \texttt{in}\ v$ is of type $\texttt{I}$ ?
5. Recall that $\dot{0} \stackrel{\text{def}}{=} \texttt{letrec}\ x_{\texttt{I}} = 0.x_{\texttt{I}}\ \texttt{in}\ x_{\texttt{I}}$. Show that there does not exist a derivation in the operational semantics for

$$\texttt{t1}_0(\texttt{pif}\ \dot{0} > \texttt{1/2}\ \texttt{then}\ 1.\dot{0}\ \texttt{else}\ 0.1.1.\dot{0}) > \texttt{1/2} \to^* *.$$

We can set $Z \stackrel{\text{def}}{=} \texttt{letrec}\ x_{\texttt{I}} = 0.x_{\texttt{I}}\ \texttt{in}\ 0.x_{\texttt{I}}$.

6. What can we conclude for the adequacy of the type $\texttt{unit}$ ? Justify.
7. Any suggestions to complete the operational semantics ?

**Solution:**

1.

$$\begin{aligned}\texttt{rec}\ \textbf{nonzero}\ &= \texttt{fn}\ m_{\texttt{I}}.\\ &\quad \texttt{pif}\ m > \texttt{1/2}\ \texttt{then}\ *\\ &\quad \texttt{else}\ \textbf{nonzero}(\texttt{t1}_0 m)\end{aligned}$$

Its semantics is the smallest fixed point of the function $F$ which to $\phi \in [\mathcal{I} \to \mathbb{S}]$ associates the function which to $a \in \mathcal{I}$ associates $\top$ if $a > 1/2$, $\varphi(\max(2a, 1)) = \varphi(2a)$ otherwise. (The $\wedge$ is trivial here.)

The iterations of Kleene are $\phi_0 = \perp$, then $\phi_1$ which associates $\top$ exactly with $a > 1/2$, then $\phi_2$ which associates $\top$ exactly with $a$ such that $a > 1/2$ or $2a > 1/2$ (i.e. $a > 1/4$).

By induction on $n$, we see that $\phi_n$ associates $\top$ exactly with $a > 1/2$. In effect, $\phi_{n+1}$ sends all $a > 1/2$ to $\top$, and all $aleq 1/2$ to $\phi_n(2a)$, that is to say to $\top$ if $2a > 1/2^n$ (i.e., $a > 1/2^{n+1}$) and to $\perp$ otherwise.

The smallest fixed point therefore always sends 0 to $\perp$, but any number $a > 0$ to $\top$ since there is an $n \in \mathbb{N}$ from which $a > 1/2^n$.

2. If $[\![u]\!]\rho > 1/2$, the left side is $[\![v]\!]\rho$ Otherwise, it is worth $[\![v]\!]\rho \wedge [\![*]\!]\rho = [\![v]\!]\rho$ since $[\![*]\!]\rho = \top$ is the largest element of $\mathbb{S}$ (and everything happens in $\mathbb{S}$ given the typing constraints).
3. By definition, $[\![\texttt{letrec}\ x_\sigma = u\ \texttt{in}\ v]\!]\rho = g(\text{lfp} f)$. Using Kleene's formula, and the Scott-continuity of $g$, this is $\sup_{n\in\mathbb{N}} g(f^n(\perp))$. The dcpo $[\![u]\!] = \mathbb{S}$ is flat, so this sup is reached for a certain $n$. Note that we must use the Scott-continuity of $g$. There is no $n \in \mathbb{N}$ such that $\texttt{fn}\ (\perp) = \text{lfp} f$ in general, as the next question shows.
4. $\texttt{I}$ does not have the ascending string property. For example, the definition of $\dot{1}$ produces such an infinite growing chain.
5. Expressions $1.\dot{0}$ and $0.1.1.\dot{0}$ are in normal form because 1._ and 0._ are not contexts. In fact, we can only start by rewriting $\dot{0} > 1/2$ in $Z > 1/2$, then in $0.Z > 1/2$, which gives $\texttt{tl}_0(\texttt{pif}\ 0.Z > 1/2\ \texttt{then}\ 1.\dot{0}\ \texttt{else}\ 0.1.1.\dot{0}) > 1/2$. But there is no longer any rule applicable to this expression.
6. It fails. Indeed, for any environment $\rho$, such as $[\![0.Z > 1/2]\!]\rho = add_0(0) = 0$,

$$\begin{aligned}[\![\texttt{t1}_0(\texttt{pif}\ 0.Z > 1/2\ \texttt{then}\ 1.\dot{0}\ \texttt{else}\ 0.1.1.\dot{0})]\!]\rho &= rem_0([\![1.\dot{0}]\!]\rho \wedge [\![0.1.1.\dot{0}]\!]\rho)\\ &= rem_0(add_1(0) \wedge add_0(add_1(add_1(0))))\\ &= rem_0(1/2 \wedge 3/8) = rem_0(3/8) = 3/4\end{aligned}$$

So $[\![\texttt{t1}_0(\texttt{pif}\ 0.Z > 1/2\ \texttt{then}\ 1.\dot{0}\ \texttt{else}\ 0.1.1.\dot{0}) > 1/2]\!]\rho = *$ , but we come to see that the operational semantics does not progress far enough to reach $*$.

7. We can already add the rules :

$$\begin{aligned}
\texttt{t1}_a(\texttt{pif } u \texttt{ then } v \texttt{ else } w) &\to \texttt{pif } u \texttt{ then } \texttt{t1}_a v \texttt{ else } \texttt{t1}_a w \\
(\texttt{pif } u \texttt{ then } v \texttt{ else } w) > 1/2 &\to \texttt{pif } u \texttt{ then } v > 1/2 \texttt{ else } w > 1/2 \\
(\texttt{pif } u \texttt{ then } v \texttt{ else } w) > 0 &\to \texttt{pif } u \texttt{ then } v > 0 \texttt{ else } w > 0
\end{aligned}$$

for $a \in \{0, 1\}$. The third form is not essential for the example, but we can see that it will be necessary in general. We could also think of adding rules like $(\texttt{pif } u \texttt{ then } v \texttt{ else } w)t \to \texttt{pif } u \texttt{ then } vt \texttt{ else } wt$, but this is not necessary, because with the adequation of type `unit` , the functions play little role.

**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 $E(\texttt{letrec } x_\sigma = u \texttt{ in } v) \stackrel{\text{def}}{=} \texttt{letrec } x = E(u) \texttt{ in } E(v)$, $E(\texttt{fn } x_\sigma.u) \stackrel{\text{def}}{=} \texttt{fn } x.E(u)$, 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.

**Solution:**

1. No, for example, $xx$ is not, neither is $* + \dot{0}$.
2. No, for example, $\texttt{fn } x.x$ has all types $\sigma \to \sigma$.