1 Denotational Semantics for REC

So far the most interesting thing we have given a denotational semantics for is the while loop. What about functions? We now have enough machinery to capture some of their semantics, even for mutually recursive functions. We show how to give a semantics for the language REC [Win93, Chp. 9].

1.1 REC Syntax

$$\begin{array}{rcl} p & ::= & \mathsf{let}\ d\ \mathsf{in}\ e \\ d & ::= & f(x_1,\ldots,x_n) = e \ \mid\ f(x_1,\ldots,x_n) = e\ \mathsf{and}\ d \\ e & ::= & n \ \mid\ x \ \mid\ e_1 \oplus e_2 \ \mid\ \mathsf{let}\ x = e_1\ \mathsf{in}\ e_2 \ \mid\ \mathsf{ifp}\ e_0\ \mathsf{then}\ e_1\ \mathsf{else}\ e_2 \ \mid\ f_i(e_1,\ldots,e_{a_i}) \end{array}$$

The expressions d are function declarations. The functions can be mutually recursive. It is reasonable to expect that under most semantics, $\mathsf{let}\ f(x) = f(x)\ \mathsf{in}\ f(0)$ will loop infinitely, but $\mathsf{let}\ f(x) = f(x)\ \mathsf{in}\ 0$ will halt and return 0.

For example,

$$\begin{array}{l} \mathsf{let}\ f_1(n,m) = \mathsf{ifp}\ m^2 - n\ \mathsf{then}\ 1\ \mathsf{else}\ (n\ \mathsf{mod}\ m)\cdot f_1(n,m+1) \\ \mathsf{and}\ f_2(n) = \mathsf{ifp}\ f_1(n,2)\ \mathsf{then}\ n\ \mathsf{else}\ f_2(n+1) \\ \mathsf{in}\ f_2(1000) \end{array}$$

In this REC program, $f_2(n)$ finds the first prime number $p \geq n$. The value of $n\ \mathsf{mod}\ m$ is positive iff m does not divide n.

1.2 CBV Denotational Semantics for REC

The meaning function is $[\![e]\!] \in \mathit{FEnv} \to \mathit{Env} \to \mathbb{Z}_\perp$, where Env and FEnv denote the sets of variable environments and function environments, respectively, as used in REC.

$$\begin{array}{rcl} \rho & \in & \mathit{Env} = \mathit{Var} \to \mathbb{Z} \\ \varphi & \in & \mathit{FEnv} = (\mathbb{Z}^{a_1} \to \mathbb{Z}_\perp) \times \cdots \times (\mathbb{Z}^{a_n} \to \mathbb{Z}_\perp) \end{array}$$

Here Var is a countable set of variables, $\mathbb{Z}$ is the set of integers, which are the values that can be bound to a variable in an environment, and $\mathbb{Z}^m = \underbrace{\mathbb{Z} \times \mathbb{Z} \times \cdots \times \mathbb{Z}}_{m \text{ times}}$.

$$
\begin{aligned}
[\![n]\!]\varphi\rho &\triangleq n \\
[\![x]\!]\varphi\rho &\triangleq \rho(x) \\
[\![e_1 \oplus e_2]\!]\varphi\rho &\triangleq \mathit{let}\ v_1 \in \mathbb{Z} = [\![e_1]\!]\varphi\rho\ \mathit{in} \\
&\ \mathit{let}\ v_2 \in \mathbb{Z} = [\![e_2]\!]\varphi\rho\ \mathit{in} \\
&\quad v_1 \oplus v_2 \\
&= [\![e_1]\!]\varphi\rho\ \oplus_\perp\ [\![e_2]\!]\varphi\rho \\
[\![\mathsf{let}\ x = e_1\ \mathsf{in}\ e_2]\!]\varphi\rho &\triangleq \mathit{let}\ y \in \mathbb{Z} = [\![e_1]\!]\varphi\rho\ \mathit{in} \\
&\quad [\![e_2]\!]\varphi\rho[y/x] \\
[\![\mathsf{ifp}\ e_0\ \mathsf{then}\ e_1\ \mathsf{else}\ e_2]\!]\varphi\rho &\triangleq \mathit{let}\ v_0 \in \mathbb{Z} = [\![e_0]\!]\varphi\rho\ \mathit{in} \\
&\ \mathit{if}\ v_0 > 0\ \mathit{then}\ [\![e_1]\!]\varphi\rho\ \mathit{else}\ [\![e_2]\!]\varphi\rho \\
[\![f_i(e_1, \ldots, e_{a_i})]\!]\varphi\rho &\triangleq \mathit{let}\ v_1 \in \mathbb{Z} = [\![e_1]\!]\varphi\rho\ \mathit{in} \\
&\ \vdots \\
&\ \mathit{let}\ v_{a_i} \in \mathbb{Z} = [\![e_{a_i}]\!]\varphi\rho\ \mathit{in} \\
&\ (\pi_i\,\varphi)(v_1, \ldots, v_{a_i})
\end{aligned}
$$

The meaning of a program $\mathsf{let}\ d\ \mathsf{in}\ e$ is

$$[\![\mathsf{let}\ d\ \mathsf{in}\ e]\!] \triangleq [\![e]\!]\varphi\rho_0,$$

where ρ_0 is some initial environment containing default values for the variables (say 0), and if the function declarations d are

$$f_1(x_1, \ldots, x_{a_1}) = e_1\ \mathsf{and}\ \ldots\ \mathsf{and}\ f_n(x_1, \ldots, x_{a_n}) = e_n,$$

then

$$
\begin{aligned}
\varphi = \mathit{fix}\ \lambda\psi \in \mathit{FEnv}.\,(&\lambda v_1 \in \mathbb{Z}, \ldots, v_{a_1} \in \mathbb{Z}.\,[\![e_1]\!]\,\psi\,\rho_0[v_1/x_1, \ldots, v_{a_1}/x_{a_1}], \\
&\vdots \\
&\lambda v_1 \in \mathbb{Z}, \ldots, v_{a_n} \in \mathbb{Z}.\,[\![e_n]\!]\,\psi\,\rho_0[v_1/x_1, \ldots, v_{a_n}/x_{a_n}]),
\end{aligned}
$$

or more accurately,

$$
\begin{aligned}
\varphi = \mathit{fix}\ \lambda\psi \in \mathit{FEnv}.\,(&\lambda v \in \mathbb{Z}^{a_1}.\,[\![e_1]\!]\,\psi\,\rho_0[\pi_1(v)/x_1, \ldots, \pi_{a_1}(v)/x_{a_1}], \\
&\vdots \\
&\lambda v \in \mathbb{Z}^{a_n}.\,[\![e_n]\!]\,\psi\,\rho_0[\pi_1(v)/x_1, \ldots, \pi_{a_n}(v)/x_{a_n}]).
\end{aligned}
$$

For this fixpoint to exist, we need to know that FEnv a pointed CPO and that the function $\mathit{FEnv} \to \mathit{FEnv}$ to which we are applying fix is continuous. The domain FEnv is a product, and a product is a pointed CPO when each factor is a pointed CPO. Each factor $\mathbb{Z}^{a_i} \to \mathbb{Z}_\perp$ is a pointed CPO, since a function is a pointed CPO when the codomain of that function is a pointed CPO, and $\mathbb{Z}_\perp$ is a pointed CPO. Therefore, FEnv is a pointed CPO.

The function $\tau : \mathit{FEnv} \to \mathit{FEnv}$ to which we are applying fix is continuous, because it can be written using the metalanguage. Here is the argument. We illustrate with $n = 2$ and $a_1 = a_2 = 1$ for simplicity, thus we assume the declaration d is

$$f_1(x) = e_1\ \mathsf{and}\ f_2(x) = e_2.$$

Then

$$\varphi \;=\; \mathit{fix}\, \lambda\psi \in \mathit{FEnv}.\, (\lambda v \in \mathbb{Z}.\, [\![e_1]\!]\, \psi\, \rho_0[v/x],\; \lambda v \in \mathbb{Z}.\, [\![e_2]\!]\, \psi\, \rho_0[v/x]).$$

This gives the least fixpoint of the operator

$$\tau \;=\; \lambda\psi \in \mathit{FEnv}.\, (\lambda v \in \mathbb{Z}.\, [\![e_1]\!]\, \psi\, \rho_0[v/x],\; \lambda v \in \mathbb{Z}.\, [\![e_2]\!]\, \psi\, \rho_0[v/x]),$$

provided we can show that τ is continuous. We can write

$$\begin{aligned}
\tau &= \lambda\psi \in \mathit{FEnv}.\, (\lambda v \in \mathbb{Z}.\, [\![e_1]\!]\, \psi\, \rho_0[v/x],\; \lambda v \in \mathbb{Z}.\, [\![e_2]\!]\, \psi\, \rho_0[v/x]) \\
&= \lambda\psi \in \mathit{FEnv}.\, (\tau_1(\psi),\, \tau_2(\psi)) \\
&= \lambda\psi \in \mathit{FEnv}.\, \langle \tau_1,\, \tau_2 \rangle\, (\psi) \\
&= \langle \tau_1,\, \tau_2 \rangle,
\end{aligned}$$

where $\tau_i : \mathit{FEnv} \to \mathit{FEnv}$ is

$$\tau_i \;=\; \lambda\psi \in \mathit{FEnv}.\, \lambda v \in \mathbb{Z}.\, [\![e_i]\!]\, \psi\, \rho_0[v/x].$$

Because $\langle \tau_1,\, \tau_2 \rangle$ is continuous iff τ_1 and τ_2 are, it suffices to show that each τ_i is continuous. Now we can write τ_i in our metalanguage.

$$\begin{aligned}
\tau_i &= \lambda\psi \in \mathit{FEnv}.\, \lambda v \in \mathbb{Z}.\, [\![e_i]\!]\, \psi\, \rho_0[v/x] \\
&= \lambda\psi \in \mathit{FEnv}.\, \lambda v \in \mathbb{Z}.\, [\![e_i]\!]\, \psi\, (\mathit{subst}\, \rho_0\, x\, v) \\
&= \lambda\psi \in \mathit{FEnv}.\, \lambda v \in \mathbb{Z}.\, ([\![e_i]\!]\, \psi)\, ((\mathit{subst}\, \rho_0\, x)\, v) \\
&= \lambda\psi \in \mathit{FEnv}.\, \lambda v \in \mathbb{Z}.\, (([\![e_i]\!]\, \psi) \circ (\mathit{subst}\, \rho_0\, x))\, v \\
&= \lambda\psi \in \mathit{FEnv}.\, (([\![e_i]\!]\, \psi) \circ (\mathit{subst}\, \rho_0\, x)) \\
&= \lambda\psi \in \mathit{FEnv}.\, \mathit{compose}\, ([\![e_i]\!]\, \psi,\, \mathit{subst}\, \rho_0\, x) \\
&= \lambda\psi \in \mathit{FEnv}.\, \mathit{compose}\, ([\![e_i]\!]\, \psi,\, \mathit{const}\, (\mathit{subst}\, \rho_0\, x)\, \psi) \\
&= \lambda\psi \in \mathit{FEnv}.\, \mathit{compose}\, (\langle [\![e_i]\!],\, \mathit{const}\, (\mathit{subst}\, \rho_0\, x) \rangle\, \psi) \\
&= \lambda\psi \in \mathit{FEnv}.\, (\mathit{compose} \circ \langle [\![e_i]\!],\, \mathit{const}\, (\mathit{subst}\, \rho_0\, x) \rangle)\, \psi \\
&= \mathit{compose} \circ \langle [\![e_i]\!],\, \mathit{const}\, (\mathit{subst}\, \rho_0\, x) \rangle \\
&= \mathit{compose}\, (\mathit{compose},\, \langle [\![e_i]\!],\, \mathit{const}\, (\mathit{subst}\, \rho_0\, x) \rangle).
\end{aligned}$$

Now we can argue that τ_i is continuous. The composition of two continuous functions is continuous, so it suffices to know that *compose* and $\langle [\![e_i]\!],\, \mathit{const}\, (\mathit{subst}\, \rho_0\, x) \rangle$ are continuous. We argued last time that *compose* is continuous. To show $\langle [\![e_i]\!],\, \mathit{const}\, (\mathit{subst}\, \rho_0\, x) \rangle$ is continuous as a function, it suffices to show that both $[\![e_i]\!]$ and $\mathit{const}\, (\mathit{subst}\, \rho_0\, x)$ are continuous as functions. The former is continuous by the induction hypothesis (structural induction on e). The latter is a constant function on a discrete domain and is therefore continuous.

1.3 CBN Denotational Semantics

The denotational semantics for CBN is the same as for CBV with two exceptions:

$$\begin{aligned}
[\![\mathsf{let}\; x = e_1\; \mathsf{in}\; e_2]\!]\, \varphi\, \rho &\;\triangleq\; [\![e_2]\!]\, \varphi\, \rho[[\![e_1]\!]\, \varphi\, \rho / x] \\
[\![f_i(e_1, \ldots, e_{a_i})]\!]\, \varphi\, \rho &\;\triangleq\; (\pi_i\, \varphi)([\![e_1]\!]\, \varphi\, \rho, \ldots, [\![e_{a_i}]\!]\, \varphi\, \rho).
\end{aligned}$$

We must extend $\mathit{Env} = \mathit{Var} \to \mathbb{Z}_\perp$ and $\mathit{FEnv} = (\mathbb{Z}_\perp^{a_1} \to \mathbb{Z}_\perp) \times \cdots \times (\mathbb{Z}_\perp^{a_n} \to \mathbb{Z}_\perp)$.

References

[Win93] Glynn Winskel. *The Formal Semantics of Programming Languages*. MIT Press, 1993.