PFPL Supplement: Comparing \texttt{fix} and \texttt{self}

Robert Harper

August 29, 2019

General recursion, \texttt{fix} : $\tau \text{ is } e$, is only sensible in a by-name dynamics for \textbf{PCF}, because it steps to \([\text{fix} \, x : \tau \text{ is } e / x]e\), which substitutes a non-value for the variable $x$ in the expression $e$. In a by-value dynamics general recursion is, for this reason, not sensible and must be replaced by type-specific forms of self-reference, such as the self-referential function form \texttt{fun}\{\tau_1; \tau_2\}(f.x.e), which is postulated to be a value of type $\tau_1 \rightarrow \tau_2$. Such a function is a value that is unrolled on application, substituting the recursive function value itself for the recursive variable, $f$, in the body of the function.

In \textbf{FPC} an alternative account of self-reference is provided by the recursive type $\tau \text{ self}$, the type of self-referential values \texttt{self} : $x$ is $e$. Within $e$ the self-reference, $x$, must be unrolled, writing \texttt{unroll}(x), to unroll the recursion and access the underlying expression $e$. More precisely,

$$\texttt{unroll}(\texttt{self} \, x \text{ is } e) \mapsto [\texttt{self} \, x \text{ is } e/x]e,$$

which makes sense in either a by-name or a by-value dynamics.

Using \texttt{self} types the recursive factorial function has type \((\text{nat} \rightarrow \text{nat}) \, \text{self}\), which reveals in its type that it is self-referential. To call such a function, either externally or internally within its definition, it is necessary to first \texttt{unroll} the self-reference and apply it to an argument, which may be either a value or a computation, depending on the dynamics. In any case the self-referential variable is only ever replaced by a value.

Curiously, in \textbf{FPC} it is possible to define \texttt{fix} from \texttt{self}, obtaining the expected dynamics, even under a by-value interpretation! The idea is that the “self” type $\tau \, \texttt{self}$ forces the recursively-defined computation to unroll the recursion before using it. Specifically, define $\hat{e}$ to be $[\texttt{unroll}(x)/x]e$, and then define \texttt{fix} : $\tau \text{ is } e$ to be the expression \texttt{unroll}(\texttt{self} \, x \, \text{is} \, \hat{e})$. Then observe the following transition:

$$\texttt{fix} \, x : \tau \text{ is } e = \texttt{unroll}(\texttt{self} \, x \, \text{is} \, \hat{e})$$
$$\quad \quad \quad \quad \mapsto [\texttt{self} \, x \, \text{is} \, \hat{e} / x]\hat{e}$$
$$\quad \quad \quad \quad = [\texttt{self} \, x \, \text{is} \, \hat{e} / x][\texttt{unroll}(x) / x]e$$
$$\quad \quad \quad \quad = [\texttt{unroll}(\texttt{self} \, x \, \text{is} \, \hat{e}) / x]e$$
$$\quad \quad \quad \quad = [\texttt{fix} \, x : \tau \text{ is } e / x]e.$$

The penultimate line summarizes the result of the iterated substitution; it does not arise in the dynamics itself as a substitution of a non-value for a variable, which would be disallowed in the by-value case.
References