PicoML Semantics

Max Kopinsky (modified from Elsa Gunter)

February 2021

1 Types

1.1 Typing rules

$$\frac{}{\Gamma \vdash c : \mathit{instantiate}(\mathit{signature}(c)) \mid \emptyset}\ \textsc{T-Const}$$

$$\frac{}{\Gamma \vdash x : \mathit{instantiate}(\Gamma(x)) \mid \emptyset}\ \textsc{T-Var}$$

$$\frac{\Gamma \vdash e_1 : \beta \mid \phi_1 \qquad \Gamma \vdash e_2 : \tau_1 \mid \phi_2 \qquad \Gamma \vdash e_3 : \tau_2 \mid \phi_3}{\Gamma \vdash \texttt{if}\ e_1\ \texttt{then}\ e_2\ \texttt{else}\ e_3 : \tau_1 \mid \{\beta \sim \texttt{bool}, \tau_1 \sim \tau_2\} \cup \phi_1 \cup \phi_2 \cup \phi_3}\ \textsc{T-If}$$

1.2 Unification

To solve all the unification constraints we generate, we use a *solve* function. The solver operates on a set of constraints to solve and returns a substitution that solves them.

α refers to a type variable, τ refers to any monotype.

$$\frac{\mathit{solve}\{\dots\} \Rightarrow \sigma}{\mathit{solve}\{\tau \sim \tau, \dots\} \Rightarrow \sigma}\ \textsc{Delete} \qquad \frac{\mathit{solve}\{\alpha \sim \tau, \dots\} \Rightarrow \sigma}{\mathit{solve}\{\tau \sim \alpha, \dots\} \Rightarrow \sigma}\ \textsc{Orient}$$

$$\frac{\mathit{solve}\{\tau_1 \sim \tau_1', \dots, \tau_n \sim \tau_n', \dots\} \Rightarrow \sigma}{\mathit{solve}\{T\ \tau_1 \dots \tau_n \sim T\ \tau_1' \dots \tau_n', \dots\} \Rightarrow \sigma}\ \textsc{Decompose}$$

$$\frac{\mathit{solve}\ [\tau/\alpha]\{\dots\} \Rightarrow \sigma}{\mathit{solve}\{\alpha \sim \tau, \dots\} \Rightarrow \sigma[\alpha \mapsto \sigma(\tau)]}\ \textsc{Eliminate},\ \alpha \notin \mathit{freeVars}(\tau)$$

In the last rule, we replace α with τ in all of the remaining constraints and solve them, getting σ. The remaining constraints might refer to free variables of τ, so we make sure to also apply the substitution σ to τ in the final mapping.

We take special care to disallow Eliminate if α is itself a free variable of τ. What would the constraint $a \sim [a]$ mean?

2 Expressions

$$\frac{}{< c, \sigma >\Downarrow_v c}\ \textsc{E-Const} \qquad \frac{}{< u, \sigma >\Downarrow_v v}\ \textsc{E-Var},\ \text{if } u := v \in \sigma$$

$$\frac{< e_1, \sigma >\Downarrow_v \texttt{true} \qquad < e_2, \sigma >\Downarrow_v v}{< \texttt{if}\ e_1\ \texttt{then}\ e_2\ \texttt{else}\ e_3, \sigma >\Downarrow_v v}\ \textsc{E-If1}$$

$$\frac{< e_1, \sigma >\Downarrow_v \texttt{false} \qquad < e_3, \sigma >\Downarrow_v v}{< \texttt{if}\ e_1\ \texttt{then}\ e_2\ \texttt{else}\ e_3, \sigma >\Downarrow_v v}\ \textsc{E-If2}$$

3 Statements

$$\frac{\Gamma \vdash e : \tau \qquad < e, \sigma >\Downarrow_v v}{< e;;, \sigma, \Gamma >\Downarrow (v, \sigma, \Gamma)}\ \textsc{S-Anon}$$

$$\frac{\Gamma \vdash e : \tau \qquad < e, \sigma >\Downarrow_v v}{< \texttt{let}\ x = e;;, \sigma, \Gamma >\Downarrow (v, \sigma[x := v], \Gamma \cup \{x : \tau\})}\ \textsc{S-Let}$$