May 8, 2016

1 Notation

- We use μ to refer to a heap. A heap is treated as a function from memory addresses to variables.
- $\mu\{a \mapsto b\}$ refers to a new function μ' defined as:

$$\mu'(x) = \begin{Bmatrix} \mu(x), & x \neq a \\ b, & x = a \end{Bmatrix}$$

- For an expression e, $e[e_1/x_1, ..., e_n/x_n]$ refers to e where every free occurrence of x_i in e is replaced with e_i.
- If f is a function $f(x)\uparrow$ and $f(x)\downarrow$ are boolean-valued postfix operators meaning that $f(x)$ is undefined or defined, respectively. $\uparrow$ alone is taken to mean the 'undefined' value.
- $\varnothing$ is overloaded to represent the single instance of the special empty type, aka `Unit`.
- A configuration is a pair $\langle \mu, e \rangle$.
- $\langle \mu, e \rangle \longrightarrow \langle \mu', e' \rangle$ if, after one reduction step on e in heap μ, the program ends in heap μ', ready to execute e'.
- A program consisting of body e begins execution in the configuration $\langle \lambda x.\uparrow, e \rangle$.
- Terms which are values are members of the set $V = \{x \Rightarrow \overline{\sigma = e}, r\}$
- A valid program terminates in a finite number of reductions in the configuration $\langle \mu, v \rangle$ for some $v \in V$.
- $\langle \mu_1, e_1 \rangle \longrightarrow_* \langle \mu_2, v \rangle$ if $\langle \mu_2, v \rangle$ is the terminating configuration after repeatedly applying reduction rules (assuming that reduction rules are congruent and that $\langle \mu_1, e_1 \rangle$ terminates).

2 Dynamic Semantics

This first section introduces a basic dynamic semantics that has no notion of an effect.

$$\boxed{\langle \mu, e, \varepsilon \rangle \simeq \langle \mu, e, \varepsilon \rangle}$$

$$\frac{\mu(x) = y}{\langle \mu, x \rangle \longrightarrow \langle \mu, y \rangle} \text{ (E-VAR)} \qquad \frac{\mu(l)\uparrow}{\langle \mu, \texttt{new x} \Rightarrow \overline{\sigma = e} \rangle \longrightarrow \langle \mu, \{l \mapsto \texttt{new x} \Rightarrow \overline{\sigma = e}\}, l \rangle} \text{ (E-NEW}_\sigma\text{)}$$

$$\frac{\begin{array}{c}\langle \mu_1, e_1 \rangle \longrightarrow_* \langle \mu_2, l \rangle \qquad \langle \mu_2, e_2 \rangle \longrightarrow_* \langle \mu_3, v \rangle \\ \mu_3() = x \Rightarrow \overline{\sigma = e} \qquad \texttt{def m}(y : \tau_1) : \tau_2 \texttt{ with } \varepsilon \in \overline{\sigma = e}\end{array}}{\langle \mu_1, e_1.m(e_2) \rangle \longrightarrow \langle \mu_3, e[l/x, v/y] \rangle} \text{ (E-METHCALL)}$$

$$\frac{\langle \mu_1, e_1 \rangle \longrightarrow_* \langle \mu_2, r \rangle \quad \langle \mu_2, e_2 \rangle \longrightarrow_* \langle \mu_3, v \rangle}{\langle \mu, e_1.\pi(e_2) \rangle \longrightarrow \langle , \varnothing \rangle} \text{ (E-OPERCALL)}$$

3 Dynamic Semantics With Effects

We amend the definition of configuration. A configuration is a triple $\langle \mu, e, \varepsilon \rangle$, where ε is an accumulated set of effects (i.e. pairs from $R \times \Pi$). The code e has the effect (r, π) if, when $\langle \lambda x.\uparrow, e, \varnothing \rangle \longrightarrow_* \langle \mu, e', \varepsilon \rangle$, we have $(r, m) \in \varepsilon$.

This definition could be overstrict. A non-terminating program (or a program which does not terminate in a value) still has effects during execution. However, it might be a useful simplification to narrow our focus to only consider those programs which terminate.

$$\boxed{\langle \mu, e, \varepsilon \rangle \simeq \langle \mu, e, \varepsilon \rangle}$$

$$\frac{\mu(x) = y}{\langle \mu, x, \varepsilon \rangle \longrightarrow \langle \mu, y, \varepsilon \rangle}\ (\textsc{E-Var}) \qquad \frac{\mu(l) \uparrow}{\langle \mu, \texttt{new x} \Rightarrow \overline{\sigma = e}, \varepsilon \rangle \longrightarrow \langle \mu, \{l \mapsto \texttt{new x} \Rightarrow \overline{\sigma = e}, l, \varepsilon \rangle}\ (\textsc{E-New}_\sigma)$$

$$\frac{\begin{array}{c}\langle \mu_1, e_1, \varepsilon_1 \rangle \longrightarrow_* \langle \mu_2, l, \varepsilon_2 \rangle \qquad \langle \mu_2, e_2, \varepsilon_2 \rangle \longrightarrow_* \langle \mu_3, v, \varepsilon_3 \rangle \\ \mu_3(l) = x \Rightarrow \overline{\sigma = e} \qquad \texttt{def m}(y : \tau_1) : \tau_2 \ \texttt{with}\ \varepsilon \in \overline{\sigma = e}\end{array}}{\langle \mu_1, e_1.m(e_2), \varepsilon_1 \rangle \longrightarrow \langle \mu_3, e[l/x, v/y], \varepsilon_3 \rangle}\ (\textsc{E-MethCall})$$

$$\frac{\langle \mu_1, e_1, \varepsilon_1 \rangle \longrightarrow_* \langle \mu_2, r, \varepsilon_2 \rangle \quad \langle \mu_2, e_2, \varepsilon_2 \rangle \longrightarrow_* \langle \mu_3, v, \varepsilon_3 \rangle}{e_1.\pi(e_2) \longrightarrow \langle \mu_3, \varnothing, \varepsilon_3 \cup \{(r, \pi)\} \rangle}\ (\textsc{E-Oper})$$