Normalized Core Language

suhorng

November 23, 2014

1 Introduction

2 Syntax

$$
\begin{array}{lcl}
v & ::= & () \\
  & | & n \in \mathbb{N} \\
  & | & b \in \mathbb{B} \\
  & | & x
\end{array}
\qquad
\begin{array}{lcl}
f & ::= & \lambda x.e \\
  & | & v\ v \\
  & | & ref\ v \\
  & | & !v \\
  & | & v := v
\end{array}
\qquad
\begin{array}{lcl}
e & ::= & v \\
  & | & \mathbf{let}\ x = v\ \mathbf{in}\ e \\
  & | & \mathbf{let}\ x = f\ \mathbf{in}\ e \\
  & | & \mathbf{if}\ v\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2
\end{array}
$$

3 Transforming to NCL

4 An Abstract Machine

$$
\begin{aligned}
\text{Addr} &\equiv (\text{left abstract}) \\
\text{Val} &\equiv \langle\mathbf{unit}\rangle \mid \langle\mathbf{int}\ \mathbb{Z}\rangle \mid \langle\mathbf{bool}\ \mathbb{B}\rangle \mid \langle\mathbf{ref}\ \text{Addr}\rangle \mid \langle\mathbf{cl}\ \text{Var}, \text{Env}, \text{Var}, e\rangle \\
\Gamma \in \text{Env} &\equiv \text{Var} \to \text{Addr} \\
\sigma \in \text{Store} &\equiv \text{Addr} \to \text{Value} \\
\kappa \in \text{Kont} &\equiv \langle\mathbf{halt}\rangle \mid \langle\mathbf{frame}\ \text{Env}, \text{Var}, e, \text{Kont}\rangle
\end{aligned}
$$

4.1 computation

Let $\kappa' = \langle \textbf{frame } \Gamma, y, e', \kappa \rangle$.

- $\langle \Gamma, \sigma, (\lambda x.e), y, e', \kappa \rangle \longmapsto \langle \sigma, v, \kappa' \rangle$

  - $v = \langle \textbf{cl } y, \Gamma, x, e \rangle$

- $\langle \Gamma, \sigma, (v_1 \ v_2), y, e', \kappa \rangle \longmapsto \langle \Gamma'', \sigma', e'', \kappa' \rangle$

  - $\langle \textbf{cl } \eta, \Gamma', x', e'' \rangle = \gamma(\Gamma, \sigma, v_1)$
  - $l \notin \text{dom}(\sigma)$
  - $\Gamma'' = \Gamma'[x' := l]$
  - $\sigma' = \sigma[l := \gamma(\Gamma, \sigma, v_2)]$

- $\langle \Gamma, \sigma, (\texttt{ref } v), y, e', \kappa \rangle \longmapsto \langle \sigma', \langle \textbf{ref } l \rangle, \kappa' \rangle$

  - $l \notin \text{dom}(\sigma)$
  - $\sigma' = \sigma[l := \gamma(\Gamma, \sigma, v)]$

- $\langle \Gamma, \sigma, (\texttt{!}v), y, e', \kappa \rangle \longmapsto \langle \sigma, \sigma(l), \kappa' \rangle$

  - $\langle \textbf{ref } l \rangle = \gamma(\Gamma, \sigma, v_1)$ and $l \in \text{dom}(\sigma)$

- $\langle \Gamma, \sigma, (v_1 \ \texttt{:=} \ v_2), y, e', \kappa \rangle \longmapsto \langle \sigma', \langle \textbf{unit} \rangle, \kappa' \rangle$

  - $\langle \textbf{ref } l \rangle = \gamma(\Gamma, \sigma, v_1)$ and $l \in \text{dom}(\sigma)$
  - $\sigma' = \sigma[l := \gamma(\Gamma, \sigma, v_2)]$

4.2 expressions

- $\langle \Gamma, \sigma, v, \kappa \rangle \longmapsto \langle \sigma, \gamma(\Gamma, \sigma, v), \kappa \rangle$

- $\langle \Gamma, \sigma, (\texttt{let } x = v \texttt{ in } e), \kappa \rangle \longmapsto \langle \sigma, \gamma(\Gamma, \sigma, v), \kappa' \rangle$

  - $\kappa' = \langle \textbf{frame } \Gamma, x, e, \kappa \rangle$.

- $\langle \Gamma, \sigma, (\texttt{let } x = f \texttt{ in } e), \kappa \rangle \longmapsto \langle \Gamma, \sigma, f, x, e, \kappa \rangle$

- $\langle \Gamma, \sigma, (\texttt{if } v \texttt{ then } e_1 \texttt{ else } e_2), \kappa \rangle \longmapsto \langle \Gamma, \sigma, e, \kappa \rangle$

  - $e = \begin{cases} e_1, & \text{if } \gamma(\Gamma, \sigma, v) = \langle \textbf{bool } \textit{true} \rangle \\ e_2, & \text{if } \gamma(\Gamma, \sigma, v) = \langle \textbf{bool } \textit{false} \rangle \end{cases}$

4.3 values

- $\langle \sigma, v, \langle \textbf{frame } \Gamma, x, e, \kappa \rangle \rangle \longmapsto \langle \Gamma', \sigma', e, \kappa \rangle$

  - $l \notin \text{dom}(\sigma)$
  - $\Gamma' = \Gamma[x := l]$
  - $\sigma' = \sigma[l := v]$

The function $\gamma : \text{Env} \times \text{Store} \times v \to \text{Val}$ is defined by

- $\gamma(\Gamma, \sigma, (\texttt{)}) = \langle \textbf{unit} \rangle$

- $\gamma(\Gamma, \sigma, n) = \langle \textbf{int } n \rangle$

- $\gamma(\Gamma, \sigma, b) = \langle \textbf{bool } b \rangle$

- $\gamma(\Gamma, \sigma, x) = \sigma(\Gamma(x))$