Opal Semantics

Alex Renda

September 2017

1 Description

Tiny turing-incomplete language based on IMP with sets:

2 Grammar

$\langle \mathit{Var} \rangle$ ::= Any identifier

$\langle \mathit{SExp} \rangle ::= \varnothing$
| $\langle \mathit{Var} \rangle$
| ($\langle \mathit{SExp} \rangle$. $\langle \mathit{SExp} \rangle$)

$\langle \mathit{Bool} \rangle ::= \langle \mathit{SExp} \rangle = \langle \mathit{SExp} \rangle$
| $\langle \mathit{SExp} \rangle \in \langle \mathit{SExp} \rangle$
| $\langle \mathit{Bool} \rangle \wedge \langle \mathit{Bool} \rangle$
| $\langle \mathit{Bool} \rangle \vee \langle \mathit{Bool} \rangle$
| `true`
| `false`

$\langle \mathit{Com} \rangle$ ::= `skip`
| `if` $\langle \mathit{Bool} \rangle$ `then` $\langle \mathit{Com} \rangle$ `else` $\langle \mathit{Com} \rangle$
| $\langle \mathit{Var} \rangle := \langle \mathit{SExp} \rangle$
| $\langle \mathit{Com} \rangle$; $\langle \mathit{Com} \rangle$

3 Small Step Semantics

$$\begin{array}{lcc} \textsc{Env} & : & \langle \mathit{Var} \rangle \to \langle \mathit{SExp} \rangle \\ \to_B & : & (\langle \mathit{Bool} \rangle \times \textsc{Env}) \to \langle \mathit{Bool} \rangle \\ \to_C & : & ((\langle \mathit{Com} \rangle \times \textsc{Env}) \to (\langle \mathit{Com} \rangle, \textsc{Env}) \end{array}$$

3.1 Boolean Relation

Should we "typecheck" at the evaluation stage? i.e. should we make sure that all sets are 100% evaluable even if we can determine truth without it?

3.1.1 Equality

$$\frac{\sigma(v) = s_1}{\langle v = s_2, \sigma\rangle \to_B s_1 = s_2}\ \textsc{Eq-VarL}$$

$$\frac{\sigma(v) = s_2}{\langle s_1 = v, \sigma\rangle \to_B s_1 = s_2}\ \textsc{Eq-VarR}$$

$$\frac{}{\langle \varnothing = \varnothing, \sigma\rangle \to_B \texttt{true}}\ \textsc{Eq-Empty}$$

$$\frac{}{\langle (s_1.s_2) = \varnothing, \sigma\rangle \to_B \texttt{false}}\ \textsc{Eq-NotL}$$

$$\frac{}{\langle \varnothing = (s_3.s_4), \sigma\rangle \to_B \texttt{false}}\ \textsc{Eq-NotR}$$

$$\frac{}{\langle (s_1.s_2) = (s_1'.s_2'), \sigma\rangle \to_B (s_1 = s_1') \wedge (s_2 = s_2')}\ \textsc{Eq-Prop}$$

3.1.2 Membership

$$\frac{\sigma(v) = s_2}{\langle s_1 \in v, \sigma\rangle \to_B s_1 \in s_2}\ \textsc{Mem-Var}$$

$$\frac{}{\langle s \in \varnothing, \sigma\rangle \to_B \texttt{false}}\ \textsc{Mem-Empty}$$

$$\frac{}{\langle s_1 \in (s_1'.s_2), \sigma\rangle \to_B (s_1 = s_1' \vee s_1 \in s_2)}\ \textsc{Mem-Check}$$

3.1.3 Conjunction

$$\frac{\langle b_1, \sigma\rangle \to_B b_1'}{\langle b_1 \wedge b_2, \sigma\rangle \to_B b_1' \wedge b_2}\ \textsc{Conj-Prop}$$

$$\frac{}{\langle \texttt{true} \wedge b_2, \sigma\rangle \to_B b_2}\ \textsc{Conj-True}$$

$$\frac{}{\langle \texttt{false} \wedge b_2, \sigma\rangle \to_B \texttt{false}}\ \textsc{Conj-False}$$

3.1.4 Disjunction

$$\frac{\langle b_1, \sigma\rangle \to_B b_1'}{\langle b_1 \vee b_2, \sigma\rangle \to_B b_1' \vee b_2}\ \textsc{Disj-Prop}$$

$$\frac{}{\langle \texttt{true} \vee b_2, \sigma\rangle \to_B \texttt{true}}\ \textsc{Disj-True}$$

$$\frac{}{\langle \texttt{false} \vee b_2, \sigma\rangle \to_B b_2}\ \textsc{Disj-False}$$

3.2 Command Relation

3.2.1 If

$$\frac{\langle b, \sigma \rangle \to_B b'}{\langle \texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2, \sigma \rangle \to_C \langle \texttt{if } b' \texttt{ then } c_1 \texttt{ else } c_2, \sigma \rangle} \textsc{If-Prop}$$

$$\frac{}{\langle \texttt{if true then } c_1 \texttt{ else } c_2, \sigma \rangle \to_C \langle c_1, \sigma \rangle} \textsc{If-True}$$

$$\frac{}{\langle \texttt{if false then } c_1 \texttt{ else } c_2, \sigma \rangle \to_C \langle c_2, \sigma \rangle} \textsc{If-False}$$

3.2.2 Sequence

$$\frac{\langle c_1, \sigma \rangle \to_C \langle c_1', \sigma' \rangle}{\langle c_1; c_2, \sigma \rangle \to_C \langle c_1'; c_2, \sigma' \rangle} \textsc{Seq-Prop}$$

$$\frac{}{\langle \texttt{skip}; c_2, \sigma \rangle \to_C \langle c_2, \sigma \rangle} \textsc{Seq-Skip}$$

3.2.3 Assignment

$$\frac{}{\langle v := s, \sigma \rangle \to_C \langle \texttt{skip}, \sigma[v \mapsto s] \rangle} \textsc{Assign}$$