Environme

Dr. Liam O'Connor
University of Edinburgh LFCS
UNSW, Term 3 2020

Where we're at

- We refined the abstract M-Machine to a *C-Machine*, with explicit stacks

- Function a

$$\frac{}{(\texttt{Apply}\ \langle\langle f.x.\ e\rangle\rangle\ \square) \triangleright s \prec v \mapsto_C s \quad \ldots e))]}$$

- We're going to extend our C-Machine to replace substitutions with an environment, giving us a new *E-Machine*

Environments

Definition

An *environment* is a context containing equality assumptions about variables.
It can be viewed like a state that maps variables to their values, except that a variable's value d

$$\bullet\ \frac{}{\textbf{Env}} \qquad \frac{}{x = v, \eta\ \textbf{Env}}$$

We write $\eta(x)$ to indicate the leftmost value correspondin

Let's change our machine states to include an environment:

$$s \mid \eta \succ e \qquad s \mid \eta \prec v$$

First Attempt

First, we'll add a rule for consulting the environment if we encounter a free variable:

$$\frac{}{\vdash}$$

Then, we just need t

One broken attempt:

$$\frac{}{(\texttt{Apply}\ \langle\!\langle f.x.\ e\rangle\!\rangle\ \square) \triangleright s \mid \eta \prec v \ \mapsto_E\ s \mid \quad e}$$

We don't know when to remove the variables again!

Second Attempt

We will extend our stacks to allow us to save the old environment to it.

$$\frac{s\ \textbf{Stack} \quad \eta\ \textbf{Env}}{}$$

Calling a function:

$$\frac{}{(\texttt{Apply}\ \langle\!\langle f.x.\ e\rangle\!\rangle\ \Box) \triangleright s \mid \eta \prec v \ \mapsto_E\ \eta \triangleright \qquad \succ e}$$

When the function returns, we restore the old environment, cl

bindings:

$$\frac{}{\eta \triangleright s \mid \eta' \prec v \ \mapsto_E\ s \mid \eta \prec v}$$

Simple Example

$$
\begin{array}{rcl|lcl}
 & & \circ & \bullet & \succ & (\texttt{Ap}\ (\texttt{Fun}\ (f.x.\ (\texttt{Plus}\ x\ (\texttt{N}\ 1))))\ (\texttt{N}\ 3)) \\
(\texttt{Ap}\ \square\ (\texttt{N}\ 3)) & \triangleright & \circ & \bullet & \succ & (\texttt{Fun}\ (f.x.\ (\texttt{Plus}\ x\ (\texttt{N}\ 1)))) \\
(\texttt{Ap}\ \square\ (\texttt{N}\ 3)) & \triangleright & \circ & \bullet & \prec & \langle\!\langle f.x.\ (\texttt{Plus}\ x\ (\texttt{N}\ 1))\rangle\!\rangle \\
(\texttt{Ap}\ \langle\!\langle\cdots\rangle\!\rangle\ \square) & \triangleright & & \bullet & \succ & \\
(\texttt{Ap}\ \langle\!\langle\cdots\rangle\!\rangle\ \square) & \triangleright & & \bullet & \prec & \\
\bullet & \triangleright & & & & \\
(\texttt{Plus}\ \square\ (\texttt{N}\ 1)) \triangleright \bullet & \triangleright & & & & \\
(\texttt{Plus}\ \square\ (\texttt{N}\ 1)) \triangleright \bullet & \triangleright & \circ & x = 3, f = \langle\!\langle\cdots\rangle\!\rangle, \bullet & \prec & 3 \\
(\texttt{Plus}\ 3\ \square) \triangleright \bullet & \triangleright & \circ & x = 3, f = \langle\!\langle\cdots\rangle\!\rangle, \bullet & \succ & (\texttt{N}\ 1) \\
(\texttt{Plus}\ 3\ \square) \triangleright \bullet & \triangleright & \circ & x = 3, f = \langle\!\langle\cdots\rangle\!\rangle, \bullet & \prec & 1 \\
\bullet & \triangleright & \circ & x = 3, f = \langle\!\langle\cdots\rangle\!\rangle, \bullet & \prec & 4 \\
 & & \circ & \bullet & \prec & 4
\end{array}
$$

Seems to work for basic examples, but is there some way to break it?

Closure Capture

$\circ \mid \bullet \succ (\texttt{Ap}\ (\texttt{Ap}\ (\texttt{Fun}\ (f.x.\ (\texttt{Fun}\ (g.y.\ x))))\ (\texttt{N}\ 3))\ (\texttt{N}\ 4))$

$\mapsto_E\ (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid \bullet \succ (\texttt{Ap}\ (\texttt{Fun}\ (f.x.\ (\texttt{Fun}\ (g.y.\ x))))\ (\texttt{N}\ 3))$

$\mapsto_E\ (\texttt{Ap}\ \square\ (\texttt{N}\ 3)) \triangleright (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid \bullet \succ (\texttt{Fun}\ (f.x.\ (\texttt{Fun}\ (g.y.\ x))))$

$\mapsto_E\ (\texttt{Ap}\ \square\ ($

$\mapsto_E\ (\texttt{Ap}\ \langle\langle f \cdot$

$\mapsto_E\ (\texttt{Ap}\ \langle\langle f \cdot$

$\mapsto_E\ \bullet \triangleright (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid x = 3, f = \langle\langle f \cdots \rangle\rangle, \bullet \succ$

$\mapsto_E\ \bullet \triangleright (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid x = 3, f = \langle\langle f \cdots \rangle\rangle, \bullet \prec \langle$

$\mapsto_E\ (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid \bullet \prec \langle\langle g.y.\ x \rangle\rangle$

$\mapsto_E\ (\texttt{Ap}\ \langle\langle g.y.\ x \rangle\rangle\ \square) \triangleright \circ \mid \bullet \succ (\texttt{N}\ 4)$

$\mapsto_E\ (\texttt{Ap}\ \langle\langle g.y.\ x \rangle\rangle\ \square) \triangleright \circ \mid \bullet \prec 4$

$\mapsto_E\ \bullet \triangleright \circ \mid y = 4, g = \langle\langle g.y.\ x \rangle\rangle, \bullet \succ x$

Oh no! We're stuck!

Something went wrong!

Returning functions as a result means that the function's body expression escapes the scope of bound variables that existed where it is defined.

The function valu $x = 3$ when the function was defined.

Solution: Store the environment inside the function valu

$$\frac{}{s \mid \eta \succ (\texttt{Recfun}\ (f.x.\ e)) \mapsto_E s \mid \eta \prec \langle\langle \eta,\ f.x.\ e \rangle\rangle}$$

This type of function value is called a *closure*.

$$\overline{\qquad\qquad\qquad}$$

$$(\texttt{Apply}\ \langle\!\langle \eta', f.x.\ e\rangle\!\rangle\ \square) \triangleright s \mid \eta \prec v \ \mapsto_E\ \eta \triangleright s \mid (x = v, f = \langle\!\langle f.x.\ e\rangle\!\rangle, \eta') \succ e$$

Retrieve the new env. from the closure

Our Example

$\circ \mid \bullet \succ (\texttt{Ap}\ (\texttt{Ap}\ (\texttt{Fun}\ (f.x.\ (\texttt{Fun}\ (g.y.\ x))))\ (\texttt{N}\ 3))\ (\texttt{N}\ 4))$

$(\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid \bullet \succ (\texttt{Ap}\ (\texttt{Fun}\ (f.x.\ (\texttt{Fun}\ (g.y.\ x))))\ (\texttt{N}\ 3))$

$(\texttt{Ap}\ \square\ (\texttt{N}\ 3)) \triangleright (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid \bullet \succ (\texttt{Fun}\ (f.x.\ (\texttt{Fun}\ (g.y.\ x))))$

$(\texttt{Ap}\ \square\ (\texttt{N}\ 3)) \triangleright (\texttt{Ap}\quad (\texttt{N}\ 4)) \triangleright \qquad , f.x.\ (\texttt{Fun}\ (g.y.\ x))$

$(\texttt{Ap}\ \langle\!\langle \bullet, f \cdots \rangle$

$(\texttt{Ap}\ \langle\!\langle \bullet, f \cdots \rangle$

$\bullet \triangleright (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid x = 3, f = \langle\!\langle f \cdots \rangle\!\rangle, \bullet \succ (\texttt{Fun}\ (g.y.\ x))$

$\bullet \triangleright (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid x = 3, f = \langle\!\langle f \cdots \rangle\!\rangle, \bullet \prec \langle\!\langle (x$

$(\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid \bullet \prec \langle\!\langle (x = 3, f = \cdots, \bullet), g.y.$

$(\texttt{Ap}\ \langle\!\langle (x = 3, f = \cdots, \bullet), g.y.\ x \rangle\!\rangle\ \square) \triangleright \circ \mid \bullet \succ (\texttt{N}$

$(\texttt{Ap}\ \langle\!\langle (x = 3, f = \cdots, \bullet), g.y.\ x \rangle\!\rangle\ \square) \triangleright \circ \mid \bullet \prec 4$

$\bullet \triangleright \circ \mid y = 4, g = \langle\!\langle g.y.\ x \rangle\!\rangle, x = 3, f = \cdots, \bullet \succ x$

$\bullet \triangleright \circ \mid y = 4, g = \langle\!\langle g.y.\ x \rangle\!\rangle, x = 3, f = \cdots, \bullet \prec 3$

$\circ \mid \bullet \prec 3$

Refinement

- We already sketched the proof that shows that each C-machine execution has a correspon
- This means t prove abou C-machine executions of the same program.
- Now we want to prove that each E-machine execution ha C-machine execution (and therefore a M-machine ex

Ingredients for Refinement

Once again, we want the same ingredients to prove a simulation proof that we did in the previous refin $\mathcal{A}$ that converts E-mac

- Each initial s
- Each final state in the E-machine is mapped to a final state in t
- For each transition from one state to another in the E-mac a corresponding transition in the C-Machine, or the two s C-machine state.

How to define $\mathcal{A}$?

- Our abstraction function $\mathcal{A}$ applies the environment η as a substitution to the current expression, and to the stack, starting at the left.
- If any enviro
  environm
- E-Machin
  environment inside closures as a substitution to the expression inside the closure.

With such a function definition, it is trivial to prove that each E-M
has a corresponding transition in the C-Machine, as it is 1:1.

Except!

There is one rule which is not 1:1. Which one?