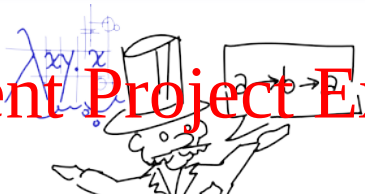


# Environments

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

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



- Function application is still executed via substitution:

$$\frac{}{(\texttt{Apply}\ \langle\!\langle f\ x.\ e \rangle\!\rangle\ \square) \triangleright s \prec v \mapsto_C s \succ e[x := v, f := (\texttt{Fun}\ (f\ x\ e))]}$$



- 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 does not change over time.

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

We write $\eta(x)$ to indicate the leftmost value corresponding to $x$ in $\eta$.

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:

$$\frac{}{s \mid \eta \succ x \quad \mapsto_E \quad s \mid \eta \prec \eta(x)}$$



Then, we just need to handle function application.

**One broken attempt:**



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

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.



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



Calling a function, we save the environment to the stack.

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



When the function returns, we restore the old environment, clearing out the new bindings:

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

# Simple Example



| | | | |
|---|---|---|---|
| | $\circ$ | $\bullet$ | $\succ$ (Ap (Fun ($f$ $x$ (Plus $x$ (N 1)))) (N 3)) |
| (Ap $\square$ (N 3)) $\triangleright$ $\circ$ | | $\bullet$ | $\succ$ (Fun ($f$ $x$ (Plus $x$ (N 1)))) |
| (Ap $\square$ (N 3)) $\triangleright$ $\circ$ | | $\bullet$ | $\prec$ $\langle\langle f.x.$ (Plus $x$ (N 1))$\rangle\rangle$ |
| (Ap $\langle\langle\cdots\rangle\rangle$ $\square$) $\triangleright$ $\circ$ | | $\bullet$ | $\succ$ (N 3) |
| (Ap $\langle\langle\cdots\rangle\rangle$ $\square$) $\triangleright$ $\circ$ | | $\bullet$ | $\prec$ 3 |
| $\bullet$ $\triangleright$ $\circ$ | $x = 3, f = \langle\langle\cdots\rangle\rangle$, | $\bullet$ | $\succ$ (Plus $x$ (N 1)) |
| (Plus $\square$ (N 1)) $\triangleright$ $\bullet$ $\triangleright$ $\circ$ | $x = 3, f = \langle\langle\cdots\rangle\rangle$, | $\bullet$ | $\succ$ $x$ |
| (Plus $\square$ (N 1)) $\triangleright$ $\bullet$ $\triangleright$ $\circ$ | $x = 3, f = \langle\langle\cdots\rangle\rangle$, | $\bullet$ | $\prec$ 3 |
| (Plus 3 $\square$) $\triangleright$ $\bullet$ $\triangleright$ $\circ$ | $x = 3, f = \langle\langle\cdots\rangle\rangle$, | $\bullet$ | $\succ$ (N 1) |
| (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 |

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

# Closure Capture

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

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

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

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

$$\mapsto_E\ (\texttt{Ap}\ \langle\!\langle f \cdot \cdot \cdot \rangle\!\rangle\ \square) \triangleright (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid \bullet \succ (\texttt{N}\ 3)$$

$$\mapsto_E\ (\texttt{Ap}\ \langle\!\langle f \cdot \cdot \cdot \rangle\!\rangle\ \square) \triangleright (\texttt{Ap}\ \square\ (\texttt{N}\ 4)) \triangleright \circ \mid \bullet \prec 3$$

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

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

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

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

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

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

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.



$$(\textbf{let } x = 3 \textbf{ in recfun } f\ y = x + y)\ 5$$

The function value $\langle\!\langle f.y.\ x + y \rangle\!\rangle$, when it is applied, does not "remember" that $x = 3$ when the function was defined.

**Solution:** Store the environment inside the function value!



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

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

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

# Our Example

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

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

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

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

$(\mathtt{Ap}\ \langle\!\langle \bullet, f \cdots \rangle\!\rangle\ \square) \triangleright (\mathtt{Ap}\ \square\ (\mathtt{N}\ 4)) \triangleright \circ \mid \bullet \succ (\mathtt{N}\ 3)$

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

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

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

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

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

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

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

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

$\circ \mid \bullet \prec 3$

## Refinement



- We already sketched the proof that shows that each C-machine execution has a corresponding M-machine execution (refinement).
- This means that any functional correctness (not security or cost) property we prove about all M-machine executions of a program apply just as well to any C-machine executions of the same program.
- Now we want to prove that each E-machine execution has a corresponding C-machine execution (and therefore a M-machine execution).

## Ingredients for Refinement



Once again, we want the same ingredients to prove a simulation proof that we did in the previous refinement. That is, we must define an abstraction function $\mathcal{A}$ that converts E-machine states to C-machine states, such that:



- Each initial state in the E-machine is mapped to an initial state in the C-Machine.
- Each final state in the E-machine is mapped to a final state in the C-Machine.
- For each transition from one state to another in the E-machine, either there exists a corresponding transition in the C-Machine, or the two states map to the same C-machine state.

# How to define $\mathcal{A}$?



- Our abstraction function $\mathcal{A}$ applies the environment $\gamma$ as a substitution to the current expression, and to the stack, starting at the left.
- If any environment is encountered in the stack, switch to substituting with that environment instead.
- E-Machine values are converted to C-Machine values merely by applying the 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-Machine transition 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?