15-2 $\Omega = (w\ w) \qquad w = (\lambda x.\, x\,x)$

$c = \langle w, \emptyset \rangle$

bytecode compiler $\downarrow$ $\quad C$ = bytecode

$\lfloor \Omega \rfloor = \lfloor w \rfloor \lfloor w \rfloor$ ap

$\lfloor w \rfloor = (\lambda x.\ x\ x\ ap)$

$\langle \varepsilon, \emptyset, \Omega, mt \rangle$

$\mapsto \langle \varepsilon, \emptyset, w\ w\ ap, mt \rangle$

$\mapsto \langle \langle w, \emptyset \rangle, \emptyset, w\ ap, mt \rangle$

$= \langle c, \emptyset, w\ ap, mt \rangle$ useless

$\mapsto \langle \langle c, c \rangle, \emptyset, ap, mt \rangle$

$\mapsto \langle \varepsilon, [x \mapsto c], (x\ x), \langle \varepsilon, \emptyset, \varepsilon, mt \rangle \rangle$

$\to \langle \varepsilon, [x \mapsto c], x\ x\ ap, \Phi \rangle$

$\mapsto \langle c, [x \mapsto c], x\ ap, \Phi \rangle$

$\mapsto \langle \langle c, c \rangle, [x \mapsto c], ap, \Phi \rangle$

$\mapsto \langle \varepsilon, [x \mapsto c], (x\ x), \langle \varepsilon, [x \mapsto c], \varepsilon, \Phi \rangle \rangle$

useless

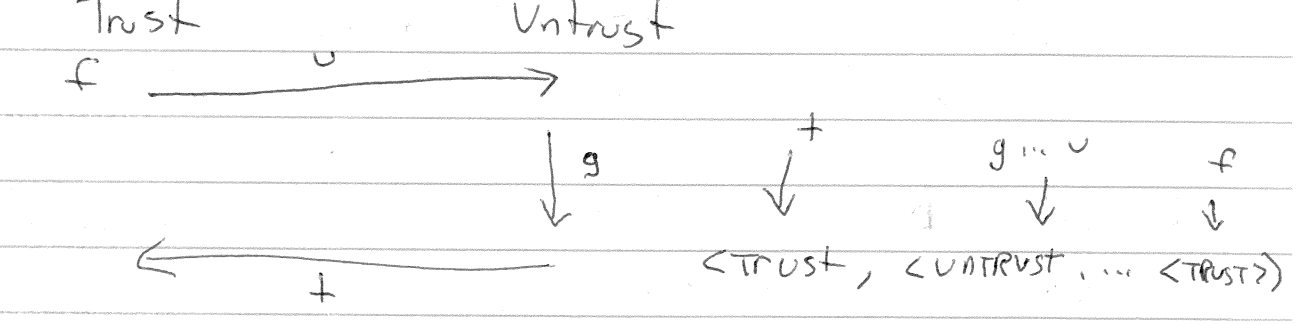
$(\lambda x.\ x\,x\,ap)\ (\lambda x.\ x\,x\,ap)\ ap$

CEK allocates on arg evaluation, frees on fun calls

SECD allocs on calls and arg eval $\quad = O(a+c)$ $\quad$ " $O(a)$

(F F) 0)

$F = (\lambda f.\ (\lambda x.\ ((f f)\ (\lambda x.\ x))))$




① $\langle (M\ N), E, K \rangle \overset{cek}{\mapsto} \langle M, E, \langle ar, N, E, K \rangle \rangle$

② $\langle (\lambda x.\ M), E, K \rangle \mapsto \langle \langle (\lambda x.\ M), E \rangle, E, K \rangle$

"safe for space"

$E|_S = E$ ~~except~~ only $S$ (restricting $E$ to $S$)

$\{x \mapsto 1, y \mapsto 2\}|_{\{x\}} = \{x \mapsto 1\}$

① $\mapsto \langle M, E|_{FV(M)}, \langle ar, N, E|_{FV(N)}, K \rangle \rangle$

② $\mapsto \langle \langle (\lambda x.\ M), E|_{FV(M)} \rangle, \emptyset, K \rangle$ ↑ prevents sharing (c.f. "flat closures")

6-1/ Does ISWIM have errors?

- stuck  $M \mapsto \_$ ← nothing
- $(x\ 7)$  (unbound variables are stuck)

$(5\ 7)$

$(/\ 1\ 0)$  $\delta$ is partial

$(+\ (\lambda(x)x)\ 5)$  $\delta$ takes only b

evalv $((5\ 7))$ = ? nothing = evalv $(\Omega)$

↓ partial

evalv $(M)$ = err  if  $M \to\!\!> N$ and $N$ is stuck

---

Error-ISWIM

$M = \ldots \mid err_l$

$l \in$ Labels (unbound, not-a-fun, invalid-arg, ...)

Old: $\delta : 0^n \times b^n \rightharpoonup V$ (partial) ← partial

New: $\delta : 0^n \times b^n \to V \cup err_l$ (total)

$\delta(/, \langle 1, 0\rangle) = err_{\text{div by zero}}$

$\delta(/, \langle 2, 1\rangle) = 2$

$(0^n\ b \ldots (\lambda(x)\ M)\ b \ldots) \to err_{\text{not constant}}$

$(b\ V) \to err_{\text{not a fun}}$

$(+\ 1\ 2\ 3)$  ① $\to err_{\text{too many args}}$  ② not a program

$m = (0^n\ m \ldots)$

$m = (0^n\ m \ldots n)$

$(x\ 5)$  ① $\to err_{\text{unbound}}$  ② not a program

$(+\ (/\ 1\ 0)\ 5) \to (+\ err_{\text{div0}}\ 5) \to err_{\text{not constant}}$

$((\lambda(x)\ (+\ x\ 5))\ (1\ 0)) \to (\ldots\ err_{\text{div0}}) \to (+\ \text{div0}\ 5) \to err_{\text{constant}}$

6-2 $((\lambda(x)\ 5)\ (/\ 1\ 0))$

$\to ((\lambda(x)\ 5)\ \text{err}_{div0})$ similar to laziness by Haskell

$\to 5$ // NOT C

$(o^n\ V \ldots \text{err}_1\ M \ldots) \to \text{err}_1$

$(\text{err}_1\ M) \to \text{err}_1$

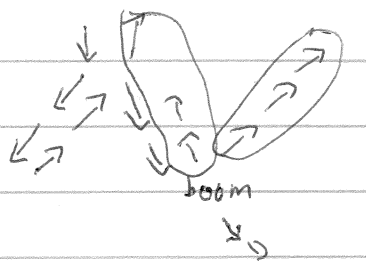
$(V\ \text{err}_1) \to \text{err}_1$

$(V_1\ (o^n\ V_2 \ldots\ (V_3\ \text{err}_1)\ M \ldots)) \to \text{err}_1$

$E = [\,] \mid (E\ M) \mid (V\ E) \mid (o^n\ V \ldots E\ M \ldots)$

$E[\text{err}_1] \to \text{err}_1$

printout E




$E[M] \to E[N]$

$E[((\lambda(x)M)\ V)] \to E[M[x \leftarrow V]]$

---

What's the connection between ISWIM and Err-ISWIM?

Extension Theorem:

For any M in ISWIM,

1. $\text{eval}_v(M) = A \Rightarrow \text{eval}_e(M) = A$
2. $\text{eval}_e(M) = A$ and $A \neq \text{err}_1 \Rightarrow \text{eval}_v(M) = A$
3. $\text{eval}_e(M) = \text{err}_1 \Rightarrow \text{eval}_v(M)$ is undefined
4. $\text{eval}_e(M)$ is undefined $\Rightarrow \text{eval}_v(M)$ is undefined

$M \simeq N$ iff $\forall C.\ C[M] = C[N]$

? $\simeq_v = \simeq_e$ ?

$\simeq_v \Rightarrow \simeq_e$

$\simeq_e \Rightarrow \simeq_v$

16->

For all M and N in ISWIM,

$M \simeq_e N \Rightarrow M \simeq_v N$ ? TRUE

Twe and easy

$M \simeq_v N \Rightarrow M \simeq_e N$ ? FALSE

↓ identifies infinite loop with error

↓ distinguishes errors and loops

$\exists M, N$ s.t.

~~$(\forall C.\ C[M] \simeq_v C[N])$~~ $(\forall C.\ C[M] =_v C[N])$

$(\exists C.\ C[M] \neq_e C[N])$

$M = (\lambda f.\ (\lambda x.\ ((f\ x)\ \Omega)))$

$N = (\lambda f.\ (\lambda x.\ \Omega))$

$C = ((\ [\ ]\ (\lambda x.\ err_1))\ (\lambda x.x))$

$C[M] = err_1$ $C[N] =$ diverge

Module A exports f

f takes a function argument

call f with a function that ~~halts~~ errors (g)

$\simeq_e \subsetneq \simeq_v$

↑ more observations

expressiveness

8.10 is interesting theorem