7-1

Does ISWIM have errors?

$$
\begin{array}{lll}
M = X & E = ▨ & V = \lambda X.M \\
\quad \mid \lambda X.M & \quad \mid E\ M & \quad \mid b \\
\quad \mid M\ M & \quad \mid V\ E & \\
\quad \mid b & \quad \mid (o^n\ V \ldots\ E\ M \ldots) & \\
\quad \mid (o^n\ M \ldots n) & &
\end{array}
$$

$$E[(\lambda X.M)\ V] \mapsto E[M[X \mapsto V]]$$
$$E[(o^n\ V \ldots)] \mapsto E[\delta(o^n, V \ldots)]$$

① Are mistakes possible?
② What does language do?

① $(\lambda x.\ 0)\ \Omega \mapsto (\lambda x.0)\ \Omega$ (diverges)

"Error" doesn't mean "Doesn't do what I want"
= "wrong program"

"stuck term"? (No RHS of $\mapsto$)

$(+\ 5)$ where $+ \in O^2$ ie $\delta(+, 5) = \bot$

$(5\ 6)$ — does not match any LHS of $\mapsto$
AND not $\in V$

$(+\ 5\ (\lambda X.X))$ $\quad \delta(+, 5, (\lambda X.X)) = \bot$

$\forall E.\ E[(+\ 5)]$ has an error ie is stuck

$(+\ (+\ 5)\ 6)$ $\quad E = (+\ ▨\ 6)$

② An error means the language ignored the program

$$
eval(M) = \begin{cases} b & \text{if } M \mapsto^* b \\ \text{'fun} & \text{if } M \mapsto^* (\lambda X.N) \end{cases}
$$

$eval((5\ 6)) = \bot$

$eval(\Omega) = \bot$

$\neg \exists a.\ (\Omega, a) \in eval$

eval is partial and identifies errors w/ non-termination

17-2/

$$
\begin{array}{llll}
M = & X & V = & b \\
 & \mid (M\ N)^{\ell} & & \mid \lambda X.M \\
 & \mid (o^n\ M \ldots)^{\ell} & & \\
 & \mid V & & \\
 & \mid \text{error}_{\ell} & \ell = & \text{some set of labels}
\end{array}
\qquad
\begin{array}{ll}
E = & [\,] \\
 & \mid (E\ M) \\
 & \mid (V\ E) \\
 & \mid (o^n\ V \ldots E\ M \ldots)
\end{array}
$$

β $\quad E[(\lambda X.M)\ V] \mapsto E[M[X \leftarrow V]]$

error@ $\quad E[(b\ \ V)]^{loc} \mapsto$ $\text{error}_{\text{not-a-fun}}$ ← preferred

$E[\text{error}]$ ← okay

~~$E[\text{error}\ \ V]$~~ $\quad \mapsto \text{error}_{loc}$

error $\quad E[\text{error}] \mapsto \text{error}$ $\qquad (+\ 7\ (+\ ((\lambda X.X)\ \text{err})\ 6))$

$\mapsto$ error

$E[o^n\ V \ldots] \mapsto E[\delta(o^n, V \ldots)]$ partial

BEFORE : $\delta : op \times \vec{V} \rightharpoonup V$

AFTER : $\delta : op \times \vec{V} \longrightarrow V \cup \text{error}_{\ell}$

total

$$
\text{eval}(M) = \begin{cases} b & \text{if } M \mapsto^* b \\ \text{'fun} & \text{if } M \mapsto^* \lambda X.N \\ \text{error}_{\ell} & \text{if } M \mapsto^* \text{error}_{\ell} \end{cases}
$$

eval is still partial

$\text{eval}(\Omega) = \bot$ $\qquad \text{eval}(5\ 6) = \text{error}$

$\text{eval}(+\ 5) = \text{err}_{\text{+ needs more args}}$ $\qquad \text{eval}(+\ 5\ \ \text{true}) = \text{error}_{\text{true ain't a number}}$

Error – ISWIM

$o^1 = \{\text{not}, \text{negate}, \text{fac}\}$

λ, 5 & M $\qquad M = \ldots$ $\qquad o^2 = \{+, <, *, \ldots\}$

$\mid (o^0)$

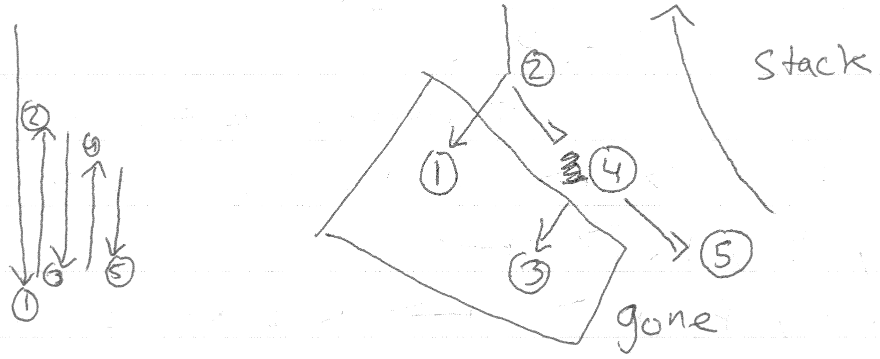
$(\lambda x,y.\ (+\ x\ \ y))$ $\qquad \mid (o^1\ M)$

$\mid (o^2\ M\ M)$

17-3/

Does the stack trace tell you where the fun was called? Or how it got these values?

Stack = Continuation = Future



Error-CEK

$\langle \text{error}_\ell,\ E,\ k \rangle$

$\longmapsto \langle \text{err}(\ell, E, tr(k)),\ \emptyset,\ mt \rangle$

$tr(k) \rightsquigarrow$ list of infor

$tr(mt) = \square$

$tr(fun(\ell, M, E, k)) = \ell$ on left $: tr(k)$

$tr(arg(\ell, V, k)) = \ell$ on right $: tr(k)$

$\langle (M\ N)^\ell,\ E,\ k \rangle$

$\longmapsto \langle M, E, fun(N, E, k) \rangle \rangle$

$\longmapsto \langle M, E, fun(\ell, N, E, k) \rangle \rangle$

Error - CEK + wcm

$\langle \text{error}_\ell,\ E,\ k \rangle \longmapsto \text{err}(\ell, E, tr'(k))$

$tr'(wcm(key, val, k)) = (key, val) : tr'(k)$

$$\begin{tikzcd}
 & (+\ 3\ 4) \\
\text{"}(+\ 3\ \ 4)\text{"} \arrow[ur] \arrow[r] & (\text{wcm debug-info line17 } (+\ 3\ 4))
\end{tikzcd}$$

17-4/

$< V, E, fun(M, E', k) >$

ORIG: $\mapsto < M, E', arg(V, k) >$

NEW $\mapsto < M, E', arg(D(V, E), k) >$

$C\cancel{E}K$ $\quad$ CEPK $\to$ continuation = future

↳ provenance = pats = past

NORM $< X, E, K > \mapsto < E(X), E, k >$

~~<X, E, K~~

NEW $< X, E, P, k > \mapsto < E(X), E, var(X, E, P), k >$

$< V, E, P, arg(\cancel{C}clo(\lambda X. M, E', P'), P'', k) >$

$\mapsto < M, E'[X \mapsto \text{camefrom}(V, P)],$

$funcall(P'', P), k >$