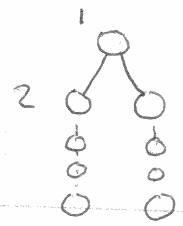
6-1




$\rightarrow$ (overall order)

$\hookrightarrow$ (all threads move a tou ce)

$(M\ N) \hookrightarrow_v (M'\ N')$ if $M \hookrightarrow_v M'$ and $N \hookrightarrow_v$

case application:

m' = code, rator, run()

N' = code, rand, run()

ret new application(m', N')

---

$M \simeq_v N \triangleq \forall C.\ C[M] \overset{\text{eval}(\ )}{=} C[N]$

$C = (\ [\ ]\ 5)$

---

$M = (+\ \underbrace{(+\ 8\ 9)}_{①}\ 10)$

②

redex
reducible
expression

$= E[(+\ 8\ 9)]$ $\quad E = (+\ [\ ]\ 10)$

$\rightarrow E[\ 17\ ]$

$= (+\ 17\ 10)$

$= E'[\ (+\ 17\ 10)\ ]$ $\quad E' = [\ ]$

$M = (-\ (+\ 2\ 2)\ \ (+\ 1\ 1))$

$= E[(+\ 2\ 2)]$ $\quad = F[(+\ 1\ 1)]$

$E = (-\ [\ ]\ (+\ 1\ 1))$ $\quad F = (-\ (+\ 2\ 2)\ [\ ])$

---

Theorem : $\forall m.\ m = V$ or $\exists E.$ (unique) s.t.

$M = E[(V_1\ V_2)]$ or $M = E[(\text{op}\ V_1 \ldots V_n)]$

(general contexts)

$$C = [\,] \mid (\lambda X.\ C) \mid (C\ M) \mid (M\ C) \mid (\text{op}\ M \ldots C\ M \ldots)$$

(EVALUATION contexts)

$$E = [\,] \mid \text{// not included, because } C \mid (E\ M) \mid (V\ E) \mid (\text{op}\ V \ldots E\ M \ldots)$$

① $M = (-\ 4\ 1) = E[(-\ 4\ 1)]$ $E = [\,]$ $\mapsto E[3] = 3$

② $M = (\lambda X.\ (+\ 1\ 1)) = E[(+\ 1\ 1)]$ $E = (\lambda X.\ [\,]) \mapsto E[2] = (\lambda X.\ 2)$

③ $M = ((\lambda x.x)\ (\lambda y.(+\ 1\ y))\ 5)$ $E = ([\,]\ 5)$

$= E[(\lambda x.x)\ (\lambda y.(+\ 1\ y))] \mapsto E[(\lambda y.(+\ 1\ y))]$

$= ((\lambda y.(+\ 1\ y))\ 5)$

④ $M = ((+\ 1\ 1)\ 2)$ so rule ④ doesn't apply

↳ Not a value

$M' = ((\lambda x.x)\ (\lambda z.2))$ so rule ④ does apply

↳ is a V

⑤ $M = (+\ 5\ 6\ 7\ (+\ 1\ 0)\ (+\ 2\ 0))$ ← redex

$E = (+\ 5\ 6\ 7\ [\,]\ (+\ 2\ 0))$ $(+\ 1\ 0)$ (redactum)

$\mapsto 1$

The Standard Reduction, $\mapsto_v$

$E[M] \mapsto_v E[M']$ if $M \vee M'$

↳ $\vee = \Delta \cup \beta_v$

runs on $\Rightarrow (\lambda X.M)V$

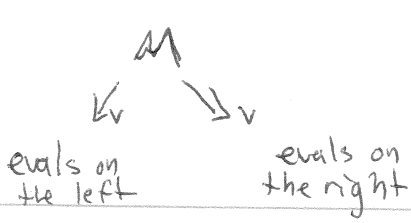
refl, trans closure: $\mapsto\!\!\!\!\rightarrow_v$

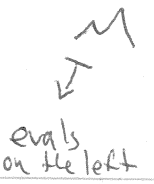
$$\text{eval}_v^s(M) = \begin{cases} b & \text{if } M \mapsto\!\!\!\!\rightarrow_v b \\ \text{fun} & \text{if } M \mapsto\!\!\!\!\rightarrow_v \lambda X.N \end{cases}$$

how use $=_v$

if $\text{eval}_v(M) = \text{eval}_v^s(M)$ ?

6-3




$X = Y$ ~~$\Leftarrow$~~ $X \subseteq Y$ and $Y \subseteq X$

$\Downarrow$

$nr\ (eval_v) = sr\ (eval^s_v)$

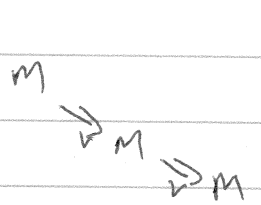
$nr \Rightarrow sr$

$sr \Rightarrow nr$

└ easy, every step of sr is clearly just a different choice of context

$M = ((\lambda x.\ 5)\ (\lambda y.\ \Omega))$

$\mapsto_v\ 5$ ↙r




---

```
Program SR (Program p) {
    Program p' = SR1 (p);
    if (p') { return SR(p'); };
    return p; }

Program⊥ SR1 (Program p) {
    Filled Context fc = p.split(); // p = E[m]
    Program⊥ m' = V(fc.m);
    return m' ∂∂ fc.E.fill(m')

V((λX.M) L) = subst(X, L, M)
V(o^n V...u) = δ
```

filled context

split : Program → Context × Program

N ⇒ E [ M ]

split ( (application (M V̄) M) )

= ~~let E' [M] = split (M)~~

let E' [R] = split (N)

ret (E' M) [R]

new FilledContext (new ContextOfAppOnLeft (E', M), R)

split ( (app (r : V) (m : NV)) ) =

let E' [R] = split (m)

ret (r E') [R]

new ContextOfAppOnRight (r, E')

split ( o^n V ... NV M ... )

let E' [NV'] = split (NV)

ret (o^n V ... E' M ...) [NV']

split (M) = [] [M]

---