Abstract machines & interpreters

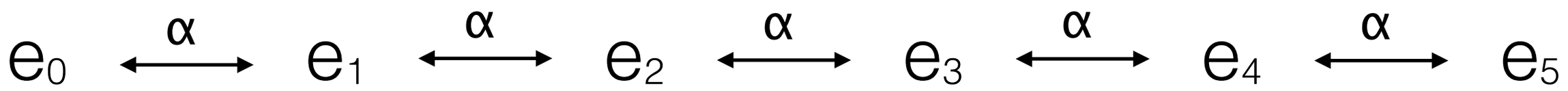
(Term rewriting semantics)

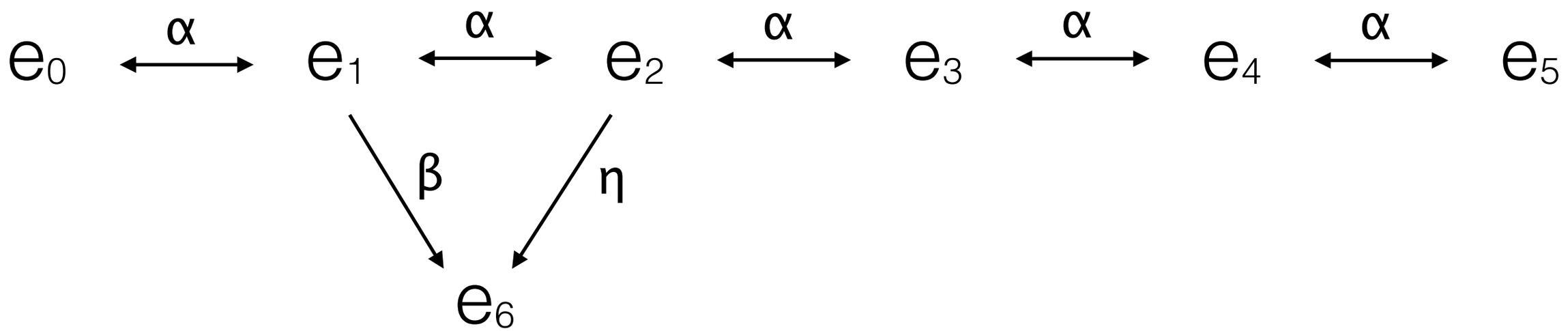
recap

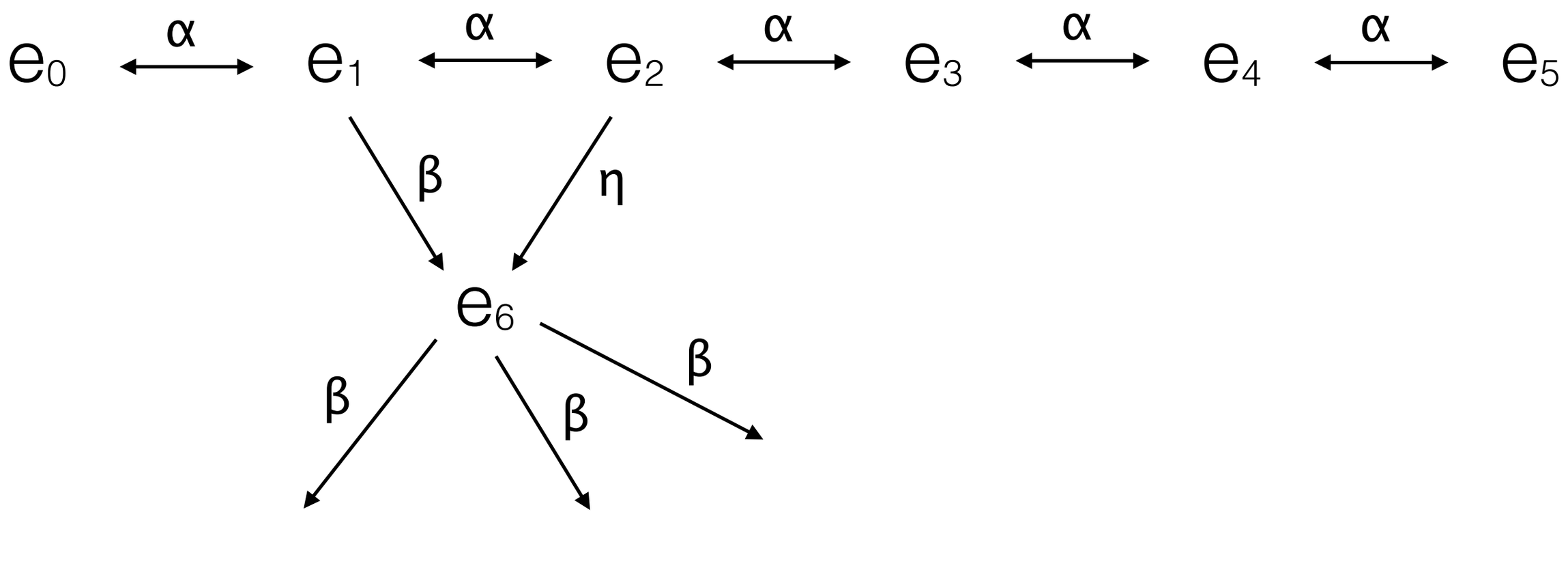
$$(\rightarrow) \;=\; (\rightarrow_\beta) \cup (\rightarrow_\alpha) \cup (\rightarrow_\eta)$$

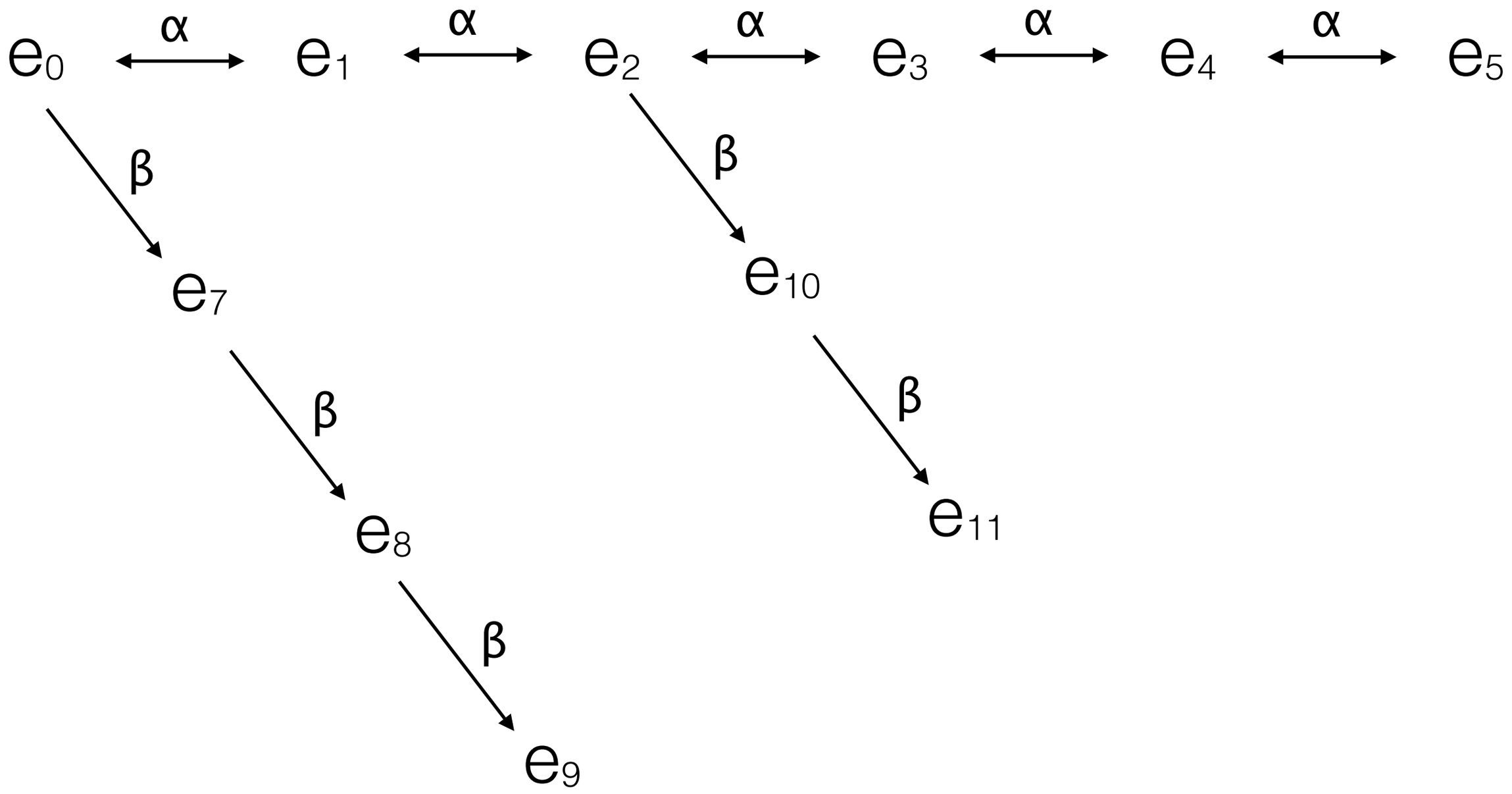
$$(\rightarrow^*)$$

A non-deterministic pre-order (i.e., reflexive, transitive)

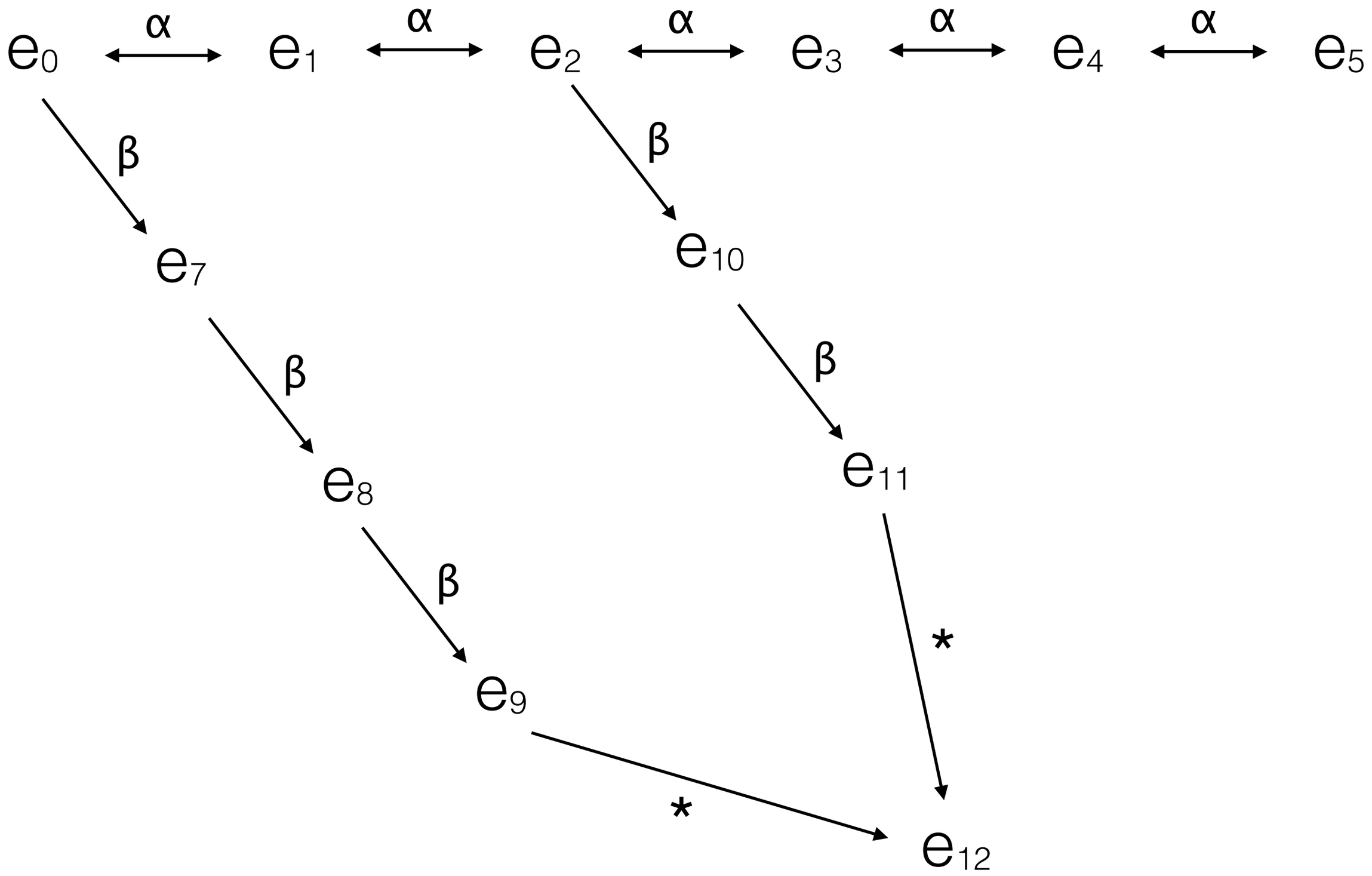

e0
α
e1
α
e2
α
e3
α
e4
α
e5


e0
α
e1
α
e2
α
e3
α
e4
α
e5
β
η
e6


e0
α
e1
α
e2
α
e3
α
e4
α
e5
β
η
e6
β
β
β


e0
α
e1
α
e2
α
e3
α
e4
α
e5
β
e7
β
e8
β
e9
β
e10
β
e11

Confluence (i.e., Church-Rosser Theorem)



$$((\lambda\ (x)\ E_0)\ E_1) \quad \rightarrow_\beta \quad E_0[x \leftarrow E_1]$$

redex

$$x[x \leftarrow E] = E$$

$$y[x \leftarrow E] = y \quad \text{where} \quad y \neq x$$

$$(E_0 \ E_1)[x \leftarrow E] = (E_0[x \leftarrow E] \ E_1[x \leftarrow E])$$

$$(\lambda \ (x) \ E_0)[x \leftarrow E] = (\lambda \ (x) \ E_0)$$

$$(\lambda \ (y) \ E_0)[x \leftarrow E] = (\lambda \ (y) \ E_0[x \leftarrow E])$$

where $y \neq x$ and $y \notin FV(E)$

```
(...

   (λ (x)
      ((λ (a) (λ (x) a))
        (λ (b) x)))

      ...)
```

```
...
  (λ (x)
     (λ (x) (λ (b) x)))
```

(λ (x)
(λ (a) (λ (x) a))
(λ (b) x)))

(λ (y) E₀)[x ← E] = (λ (y) E₀[x ← E])

where y ≠ x and y ∉ FV(E)

Evaluation contexts

$$
\begin{array}{rcl}
v & ::= & (\lambda\ (x)\ e) \\
\mathscr{E} & ::= & (\mathscr{E}\ e) \\
 & | & (v\ \mathscr{E}) \\
 & | & \square
\end{array}
$$

$$
\begin{array}{rcl}
e & ::= & (\lambda\ (x)\ e) \\
 & | & (e\ e) \\
 & | & x
\end{array}
$$

Context and redex

$$\mathscr{E}[\overbrace{(\mathsf{v}\ \ \mathsf{v})}^{\mathsf{r}}] =$$

```
((((λ (x) ((λ (y) y) x)) (λ (z) z)) (λ (w) w))
```

$\mathscr{E}$ = (□ (λ (w) w))

r = ((λ (x) ((λ (y) y) x)) (λ (z) z))

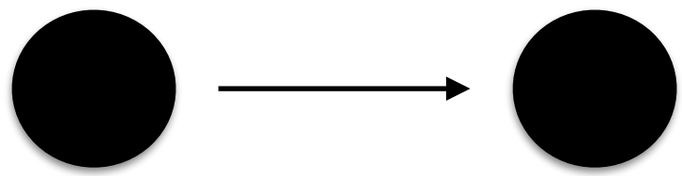
Abstract Machines

$$\frac{(\texttt{e}_0, \text{env}) \Downarrow (\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e}_2\texttt{)}, \text{env'}) \qquad (\texttt{e}_1, \text{env}) \Downarrow v_1 \qquad (\texttt{e}_2, \text{env'}[x \mapsto v_1]) \Downarrow v_2}{(\texttt{(e}_0\ \texttt{e}_1\texttt{)}, \text{env}) \Downarrow v_2}$$

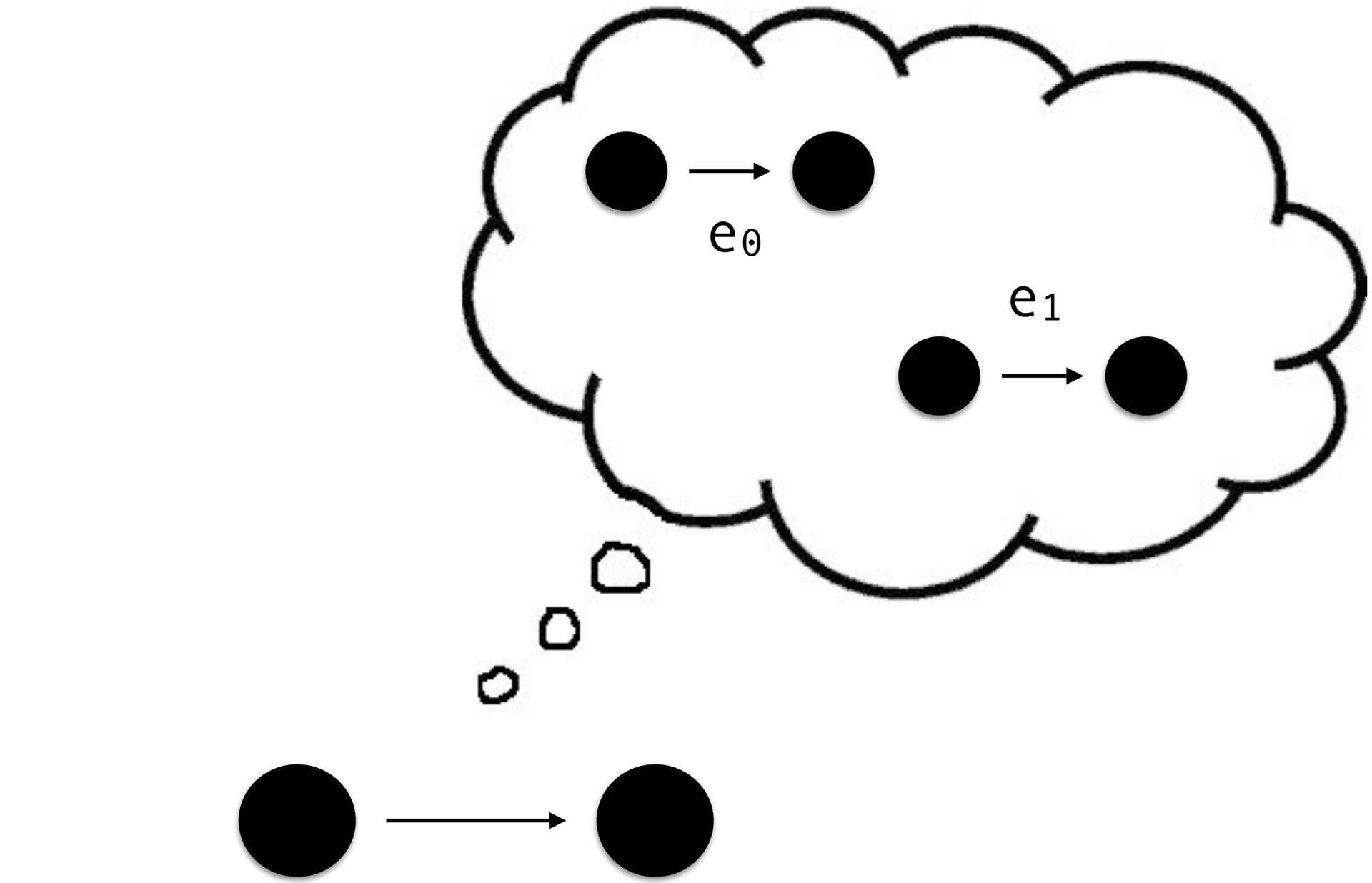
$$\frac{}{(\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e)}, \text{env}) \Downarrow (\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e)}, \text{env})}$$

$$\frac{}{(x, \text{env}) \Downarrow \text{env}(x)}$$

Big-step (Natural) Semantics



$(e_0\ e_1)$, env e', env'



(e0 e1), env e', env'

```
(define (interp e env)
  (match e
         [(? symbol? x)
          (hash-ref env x)]

         [`(λ (,x) ,e₀)
          `(clo (λ (,x) ,e₀) ,env)]

         [`(,e₀ ,e₁)
          (define v₀ (interp e₀ env))
          (define v₁ (interp e₁ env))
          (match v₀
            [`(clo (λ (,x) ,e₂) ,env)
             (interp e₂ (hash-set env x v₁))])]))
```

```
(define (interp e env)
  (match e
         [(? symbol? x)
          (hash-ref env x)]

         [`(λ (,x) ,e₀)
          `(clo (λ (,x) ,e₀) ,env)]

         [`(,e₀ ,e₁)
          (define v₀ (interp e₀ env))
          (define v₁ (interp e₁ env))
          (match v₀
            [`(clo (λ (,x) ,e₂) ,env)
             (interp e₂ (hash-set env x v₁))])]))
```

$$\frac{(\texttt{e}_0, \text{env}) \Downarrow (\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e}_2\texttt{)}, \text{env'}) \qquad (\texttt{e}_1, \text{env}) \Downarrow v_1 \qquad (\texttt{e}_2, \text{env'}[x \mapsto v_1]) \Downarrow v_2}{(\texttt{(e}_0\ \texttt{e}_1\texttt{)}, \text{env}) \Downarrow v_2}$$

$$\frac{}{(\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e)}, \text{env}) \Downarrow (\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e)}, \text{env})}$$

$$\frac{}{(x, \text{env}) \Downarrow \text{env}(x)}$$

$((\lambda\ (x)\ \ldots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow \ldots$

$(((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \ldots$

$$\frac{}{((\lambda\ (x)\ \ldots), \varnothing) \Downarrow ((\lambda\ (x)\ \ldots), \varnothing)}$$

$$\frac{((\lambda\ (x)\ \ldots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow \ldots}{}$$

$$\frac{(((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \ldots}{}$$

$$\frac{\dots \Downarrow ((\lambda\ (x)\ \dots), \varnothing) \qquad ((\lambda\ (z)\ z), \varnothing) \Downarrow ((\lambda\ (z)\ z), \varnothing)}{((\lambda\ (x)\ \dots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow \dots}$$

$$\frac{}{(((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \dots}$$

$$\frac{(\texttt{e}_0, \text{env}) \Downarrow (\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e}_2\texttt{)}, \text{env'}) \qquad (\texttt{e}_1, \text{env}) \Downarrow v_1 \qquad (\texttt{e}_2, \text{env'}[x \mapsto v_1]) \Downarrow v_2}{(\texttt{(e}_0\ \texttt{e}_1\texttt{)}, \text{env}) \Downarrow v_2}$$

$$\frac{}{(\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e)}, \text{env}) \Downarrow (\texttt{(}\lambda\ \texttt{(x)}\ \texttt{e)}, \text{env})}$$

$$\frac{}{(x, \text{env}) \Downarrow \text{env}(x)}$$

$$\frac{\dots \Downarrow ((\lambda\ (z)\ z), \varnothing)}{(((\lambda\ (x)\ \dots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow \dots}\qquad \frac{}{(((\lambda\ (y)\ y)\ x)), [x \mapsto ((\lambda\ (z)\ z), \varnothing)]) \Downarrow \dots}$$

$$\frac{(((\lambda\ (x)\ \dots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow \dots}{(((((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \dots}$$

$$\dots \Downarrow ((\lambda\ (y)\ y), [x \mapsto ((\lambda\ (z)\ z), \varnothing)])$$

$$\frac{\dots \Downarrow ((\lambda\ (z)\ z), \varnothing) \qquad (((\lambda\ (y)\ y)\ x)), [x \mapsto ((\lambda\ (z)\ z), \varnothing)]) \Downarrow \dots}{(((\lambda\ (x)\ \dots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow \dots}$$

$$(((((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \dots$$

$$\frac{}{(x,\ [x \mapsto ((\lambda\ (z)\ z), \varnothing)]) \Downarrow ((\lambda\ (z)\ z), \varnothing)}$$

$$\frac{}{\ldots \Downarrow ((\lambda\ (y)\ y), [x \mapsto ((\lambda\ (z)\ z), \varnothing)])}$$

$$\frac{}{\ldots \Downarrow ((\lambda\ (z)\ z), \varnothing)} \qquad \frac{}{(((\lambda\ (y)\ y)\ x)), [x \mapsto ((\lambda\ (z)\ z), \varnothing)]) \Downarrow \ldots}$$

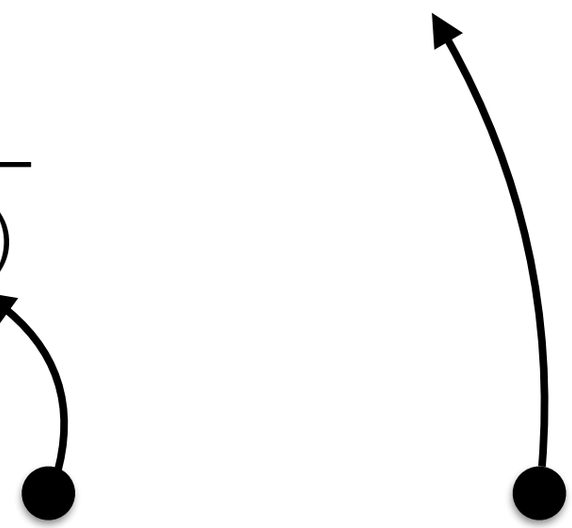
$$\frac{}{((\lambda\ (x)\ \ldots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow \ldots}$$

$$\frac{}{((((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \ldots}$$

$$\frac{}{(y, [y \mapsto ((\lambda\ (z)\ z), \varnothing), x \mapsto ((\lambda\ (z)\ z), \varnothing)]) \Downarrow ((\lambda\ (z)\ z), \varnothing)}$$

$$\frac{}{(x, [x \mapsto ((\lambda\ (z)\ z), \varnothing)]) \Downarrow ((\lambda\ (z)\ z), \varnothing)}$$

$$\frac{}{\ldots \Downarrow ((\lambda\ (y)\ y), [x \mapsto ((\lambda\ (z)\ z), \varnothing)])}$$



$$\frac{}{\ldots \Downarrow ((\lambda\ (z)\ z), \varnothing)} \qquad \frac{}{(((\lambda\ (y)\ y)\ x)), [x \mapsto ((\lambda\ (z)\ z), \varnothing)]) \Downarrow \ldots}$$

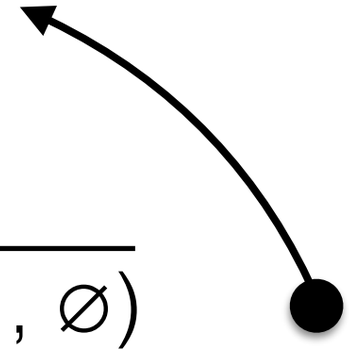
$$\frac{}{((\lambda\ (x)\ \ldots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow \ldots}$$

$$\frac{}{((((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \ldots}$$

$$((\lambda\ (w)\ w), \varnothing) \Downarrow ((\lambda\ (w)\ w), \varnothing)$$

$$\frac{}{((\lambda\ (x)\ \ldots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow ((\lambda\ (z)\ z), \varnothing)}$$

$$\frac{}{(((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \ldots}$$

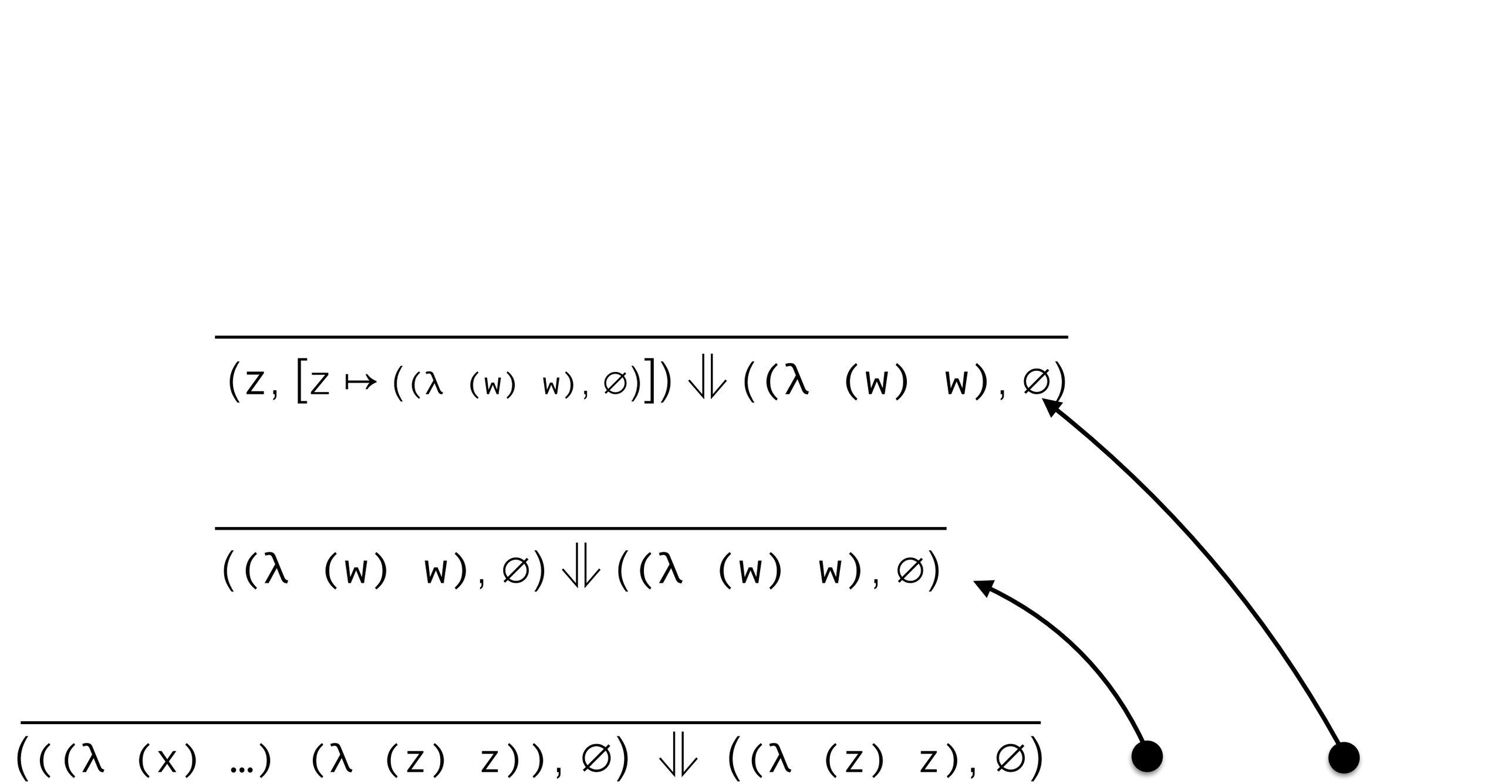


$$\frac{}{(z, [z \mapsto ((\lambda\ (w)\ w), \varnothing)]) \Downarrow ((\lambda\ (w)\ w), \varnothing)}$$

$$\frac{}{((\lambda\ (w)\ w), \varnothing) \Downarrow ((\lambda\ (w)\ w), \varnothing)}$$

$$\frac{}{(((\lambda\ (x)\ \ldots)\ (\lambda\ (z)\ z)), \varnothing) \Downarrow ((\lambda\ (z)\ z), \varnothing)}$$

$$\frac{}{(((((\lambda\ (x)\ ((\lambda\ (y)\ y)\ x))\ (\lambda\ (z)\ z))\ (\lambda\ (w)\ w)), \varnothing) \Downarrow \ldots}$$



Review

- Meta-circular interpreters (reuse all features from host)
- Term-rewriting (non-deterministic but consistent)
- outermost and innermost beta-reductions =>
    - Lazy (CBN) and Eager (CBV) when stop at **λ**
- Using a grammar for evaluation contexts, we force CBV
- Substitution environments and big-step reductions
    - Equivalent to a closure-creating interpreter

Exercises

Try evaluating using context&redex and then big-step

1.
```
(((λ (z) z) (λ (u) (u u)))
 (λ (x) x))
```

2.
```
(((λ (f) (λ (g) ((f g) f)))
  (λ (z) z))
 (λ (u) (u u)))
```