Proofnets and Paths in Constructive Classical Logic: Too Old, Too New

Harry Mairson

In America, the young are always ready to give to those who are older than themselves the full benefits of their inexperience.
Oscar Wilde

Constructive classical logic — extending the Curry-Howard correspondance to classical proofs

- Is there anything left to be said — especially in Marseille?
- Old Research: how do we construct proofnets for classical logic? (Lawall and Mairson, Sharing Continuations: Proofnets for Languages with Explicit Control, ESOP 2000)
- New Research: some observations on paths in classical proofs — extending the GoI, Q/A, interaction metaphor to synchronized, concurrent paths (multiple threads/continuations)

Point of departure: Polarized proofnets and $\lambda\mu$-calculus (O. Laurent, TCS 2003)

λ-calculus + call/cc: "Unfortunately, the reduction rules for this new constant depend on the reduction strategy (CBV or CBN), contradicting the Church-Rosser property ..."

$$\text{call/cc}\ \lambda k.(\lambda x.1)(k\,2) \;\to\; 1 \quad \text{CBN}$$

$$\text{call/cc}\ \lambda k.(\lambda x.1)(k\,2) \;\to\; 2 \quad \text{CBV}$$

Idea: Compile into MELL, using CBx CPS translation + direct-style translation

Result: Proofnets à la Laurent, translated from LLP to MELL, but with boxes around the "kingdoms"

Continuations FOR DUMMIES

```
(begin (factorial n k)
  (if (= n 0)
      (k 1)
      (factorial (- n 1) (lambda (v) (k (* n v))))))
```

```
(factorial 5 (lambda (v) v))
```

Bound variable in continuation acts like a crude register (if you are a compiler writer, maybe)...

```
(factorial 5 (lambda (v) v))
...
(factorial 3 (lambda (v) (* 5 (* 4 v))))
...
(factorial 0 (lambda (v) (* 5 (* 4 (* 3 (* 2 (* 1 v)))))))
```

Continuation passing style isn't much harder than this – it's just a different procedure– the evaluator...

Continuations FOR DUMMIES (2)

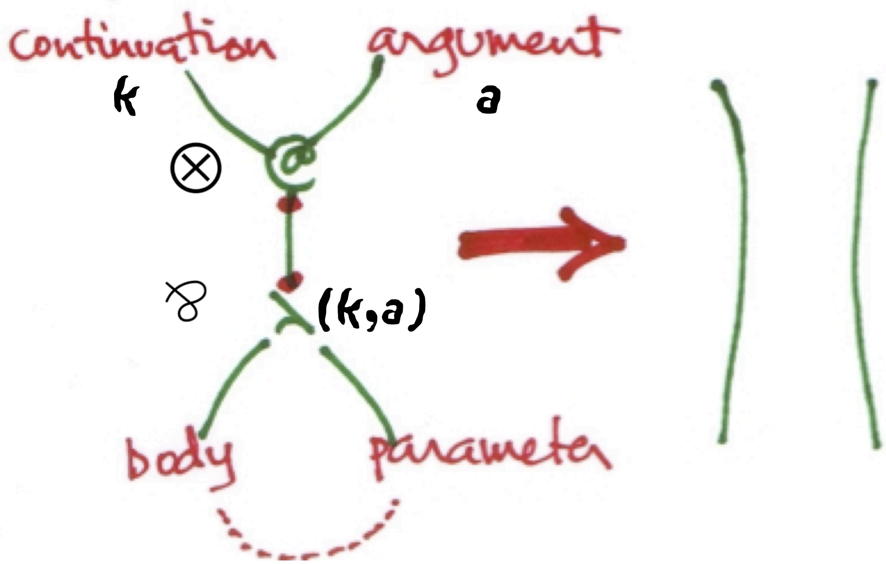
When calling a procedure, you need to tell it the **input**, and **what to do with the output** (here, call by value)

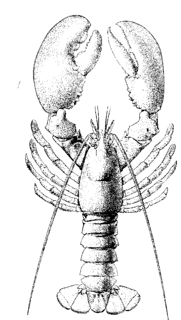
```
(define (eval exp env cont)
  (cond ((variable? exp) (cont (lookup env exp)))
        ((lambda? exp)
         (cont (lambda (x k)
                 (eval (lambda-body exp)
                       (extend-env (binder exp) x env)
                       k))))
        ((application? exp)
         (eval (function-part exp)
               env
               (lambda (f)
                 (eval (argument-part exp)
                       env
                       (lambda (a) (f a cont)))))))
```

Continuations FOR DUMMIES (3)

When calling a procedure, you need to tell it the input, and what to do with the output.

This should come as no surprise in the context of proofnets for lambda calculus...





(Par is like a lobster...)

Proofnets for λ-calculus + call/cc

For now: CBV CPS transformation
CBN $!A \multimap B$ translation to MELL proofnets
Direct style translation on proofnets

Recall $\alpha^* \equiv \alpha$ (variables and $\perp$)
$(\alpha \to \beta)^* \equiv \alpha^* \to \neg\neg\beta^*$ $\quad (\neg\tau \equiv \tau \to \perp)$

CBV CPS maps $\Gamma \vdash E : \sigma$ to $\Gamma^* \vdash E : \neg\neg\sigma^*$

Double negation embeddings: If $\Gamma \vdash E : \sigma$ is classically true, then $\Gamma^* \vdash E : \neg\neg\sigma^*$ is intuitionistically true.

call/cc gets the type of Peirce's axiom:
((A -> B) -> A) -> A.

Proofnets for λ-calculus + call/cc

For now: CBV CPS transformation
CBN $!A \multimap B$ translation to MELL proofnets
Direct style translation on proofnets

Recall $\alpha^* \equiv \alpha$ (variables and $\perp$)
$(\alpha \to \beta)^* \equiv \alpha^* \to \neg\neg\beta^*$ $\quad (\neg\tau \equiv \tau \to \perp)$

CBV CPS maps $\Gamma \vdash E : \sigma$ to $\Gamma^* \vdash E : \neg\neg\sigma^*$

$$[\![x]\!] \equiv \lambda k. k x$$
$$[\![\lambda x. M]\!] \equiv \lambda k. k(\lambda x. \lambda k. [\![M]\!] k) =_\eta \lambda k. k(\lambda x. [\![M]\!])$$
$$[\![M N]\!] \equiv \lambda k. [\![M]\!](\lambda m. [\![N]\!](\lambda n. m n k))$$
$$[\![\mathrm{call/cc}]\!] \equiv \lambda k. k(\lambda f. \lambda k. f(\lambda v. \lambda c. k v) k)$$

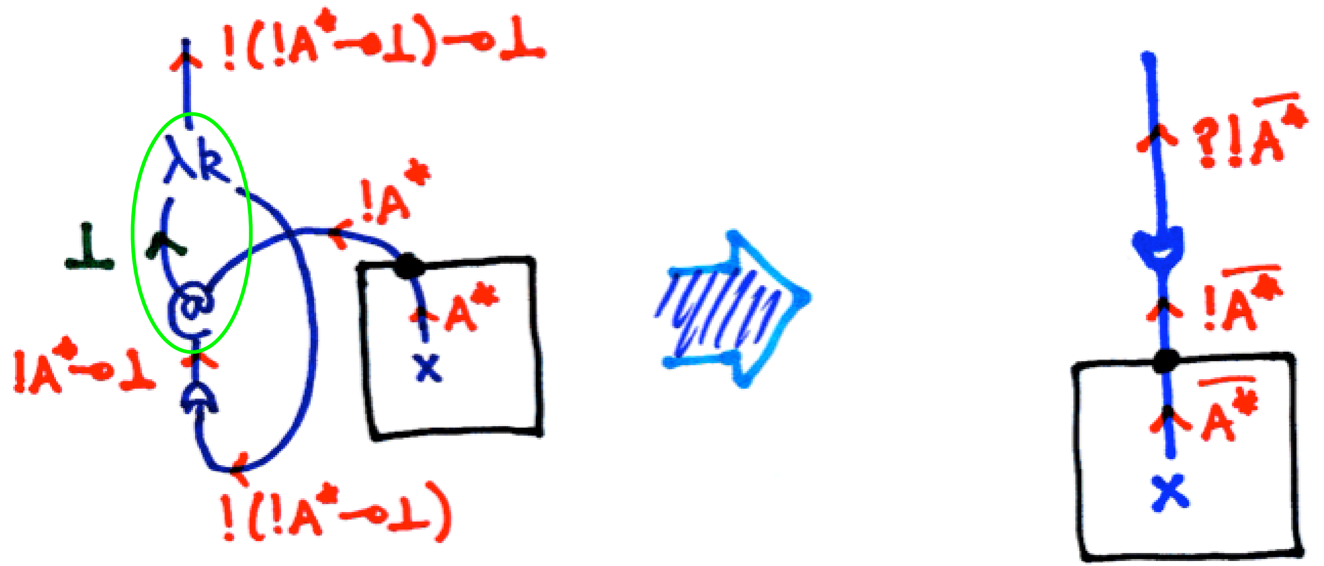
CBN translation to MELL: $\overline{A \to B} \equiv !\overline{A} \multimap B$

What about the direct-style transformation?

Direct-style transformation

$\alpha \otimes 1 = \alpha, \quad \alpha ⅋ \perp = \alpha$

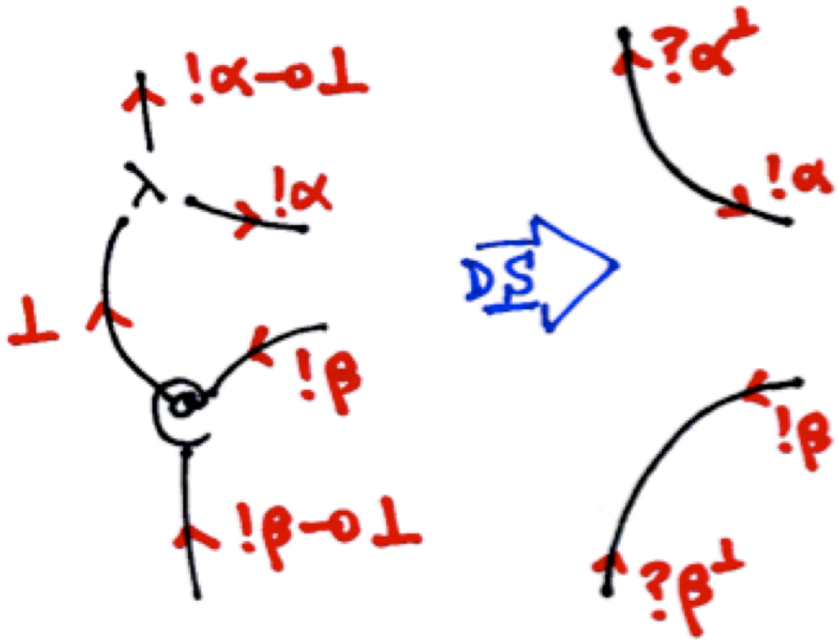
$[\![x^A]\!] = \lambda k : A^* \to \perp .\, k x \; : \; (A^* \to \perp) \to \perp$



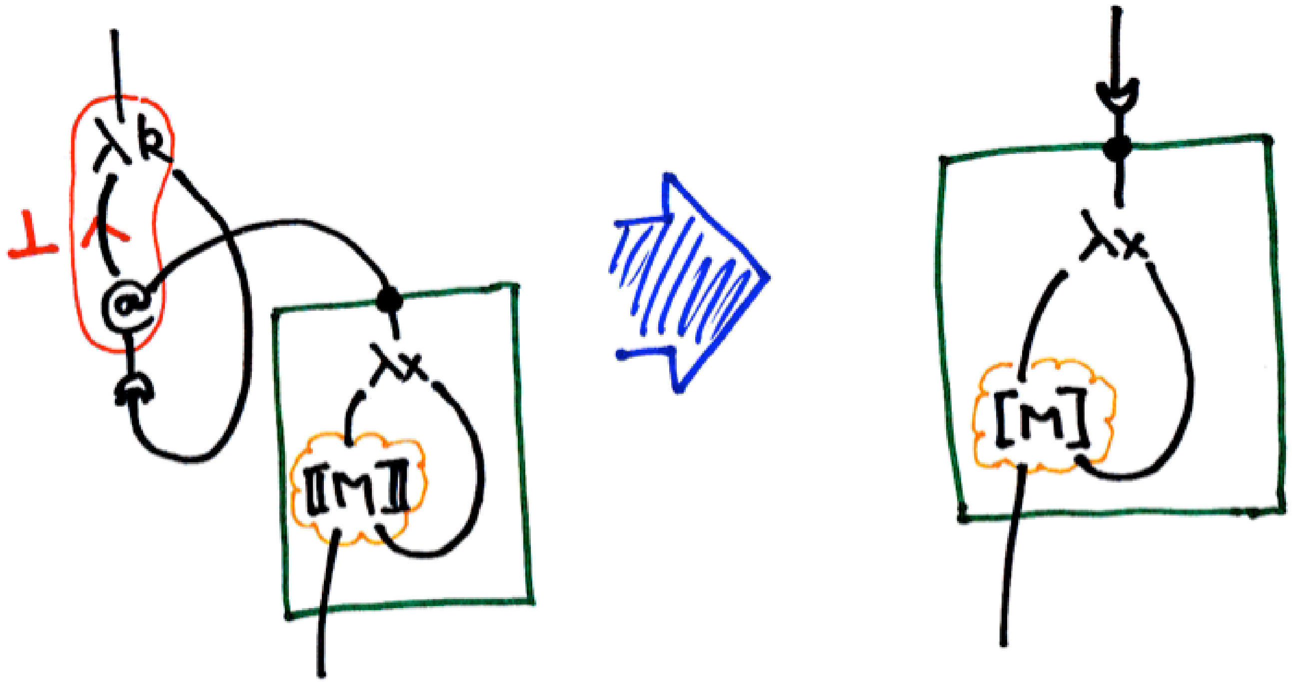
Recall $!\alpha \multimap \perp = (!\alpha)^\perp ⅋ \perp = (!\alpha)^\perp = ?\alpha^\perp$

$!(!A^* \multimap \perp) \multimap \perp = ?(!\overline{A^*} \multimap \perp)^\perp = ?(?\overline{A^*}^\perp)^\perp = ?!\overline{A^*}$

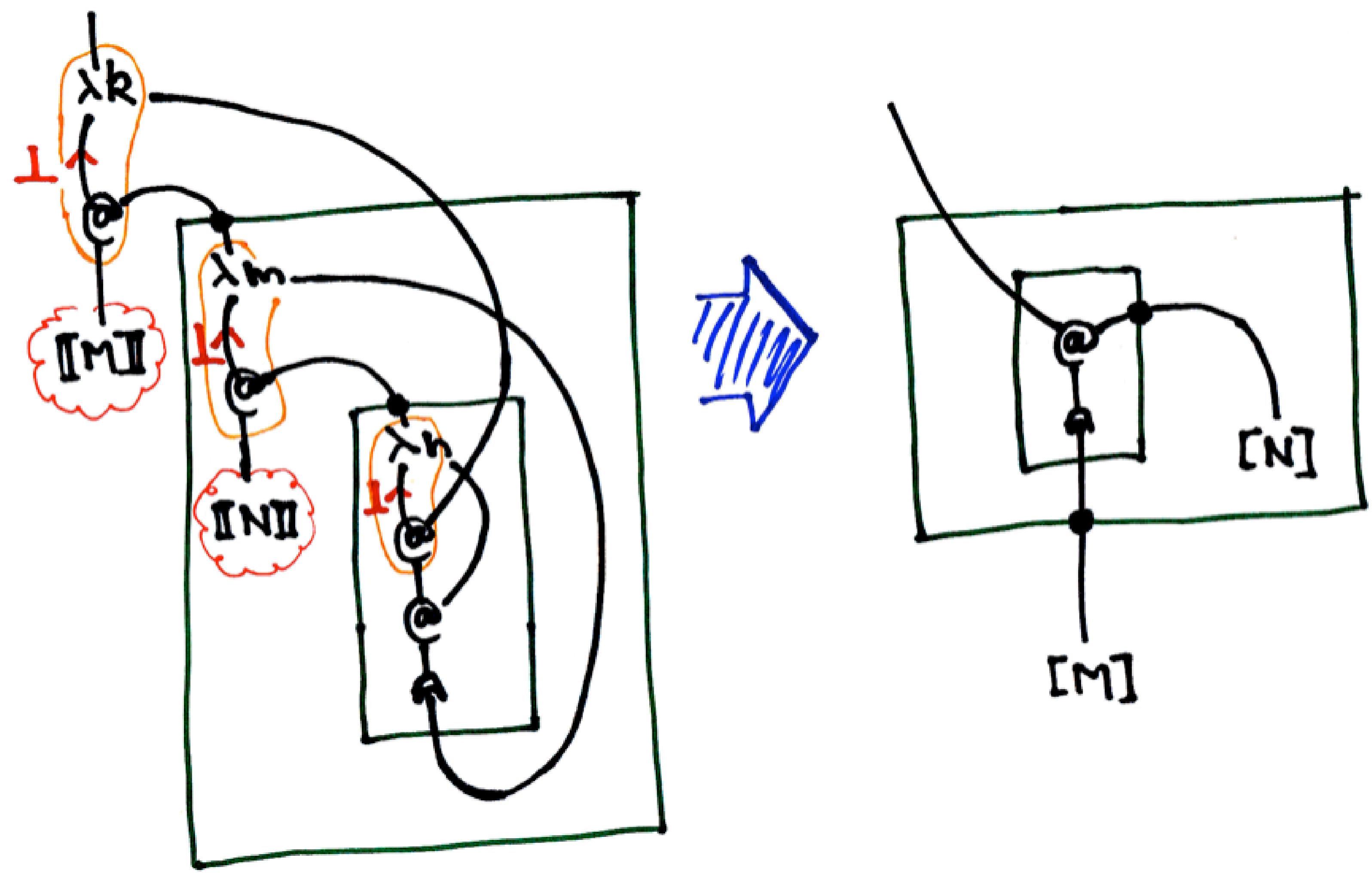
Direct-style transformation



$$[\![\lambda x. M]\!] =_\eta \lambda k. k(\lambda x. [\![M]\!])$$

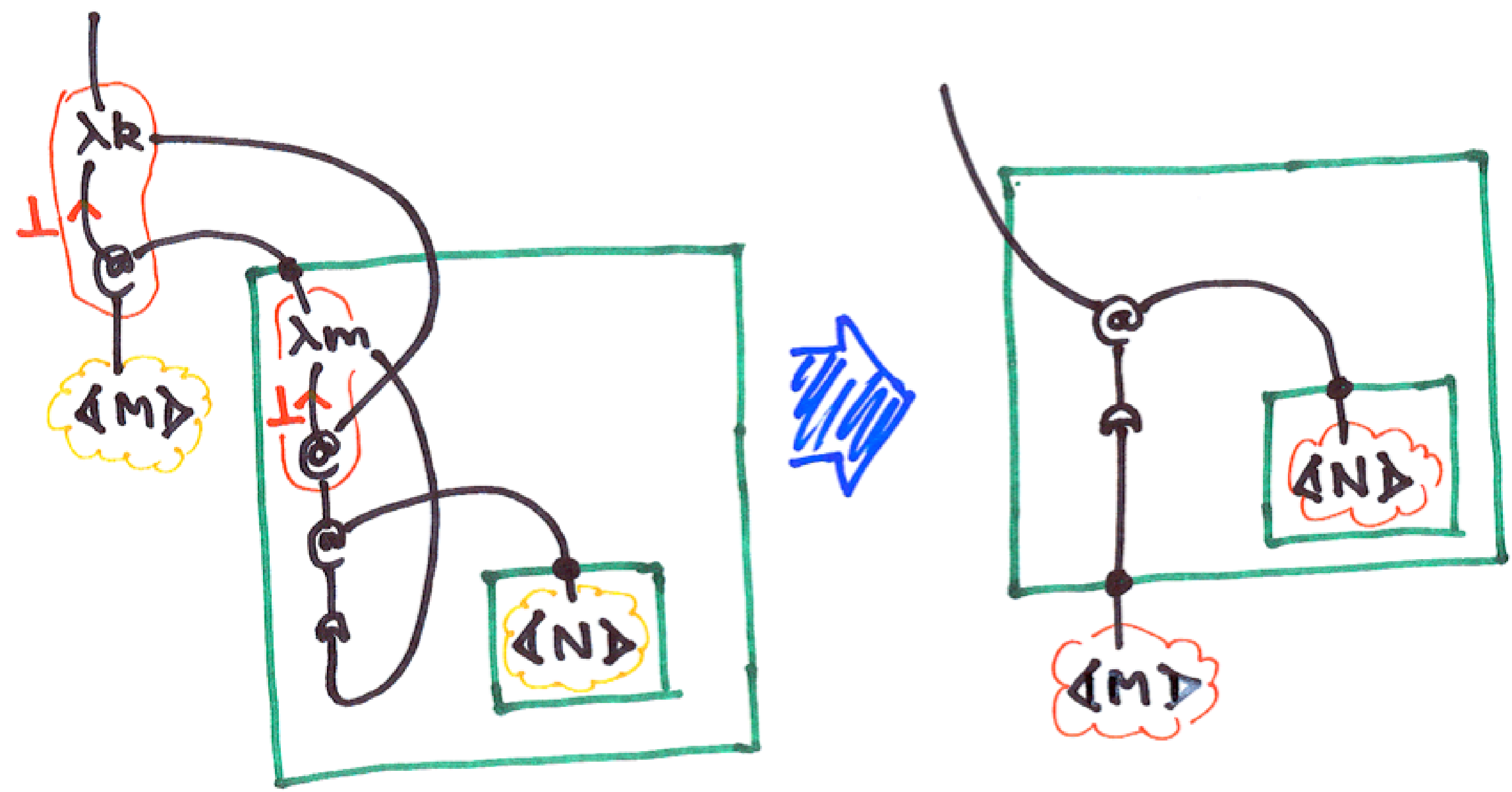


$$[\![MN]\!] = \lambda k.\,[\![M]\!]\,(\lambda m.\,[\![N]\!]\,(\lambda n.\,mnk))$$

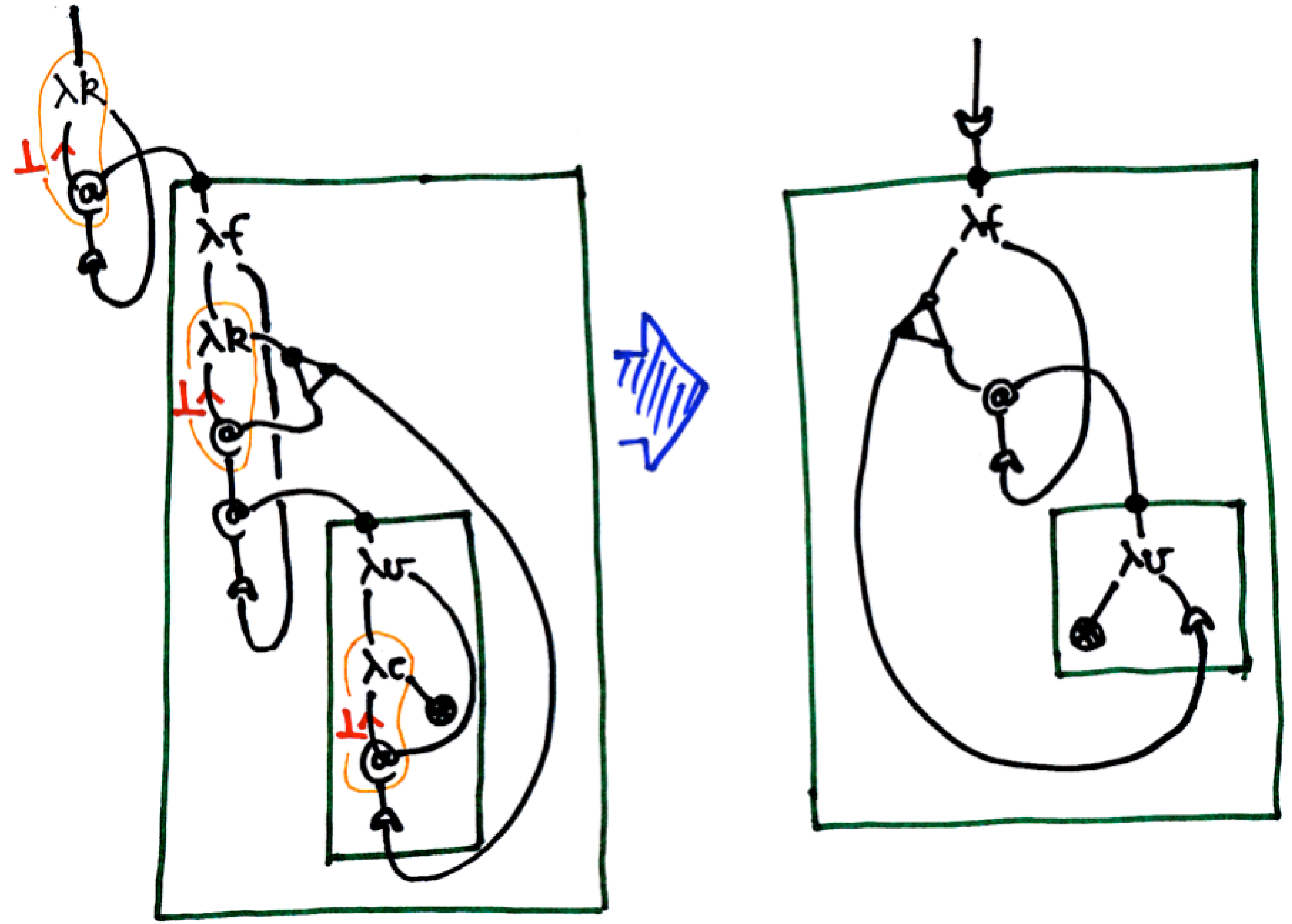


What about the CBN CPS transformation, &c.?

$$\langle\!\langle MN \rangle\!\rangle = \lambda k.\langle\!\langle M \rangle\!\rangle(\lambda m.m\langle\!\langle N \rangle\!\rangle k)$$

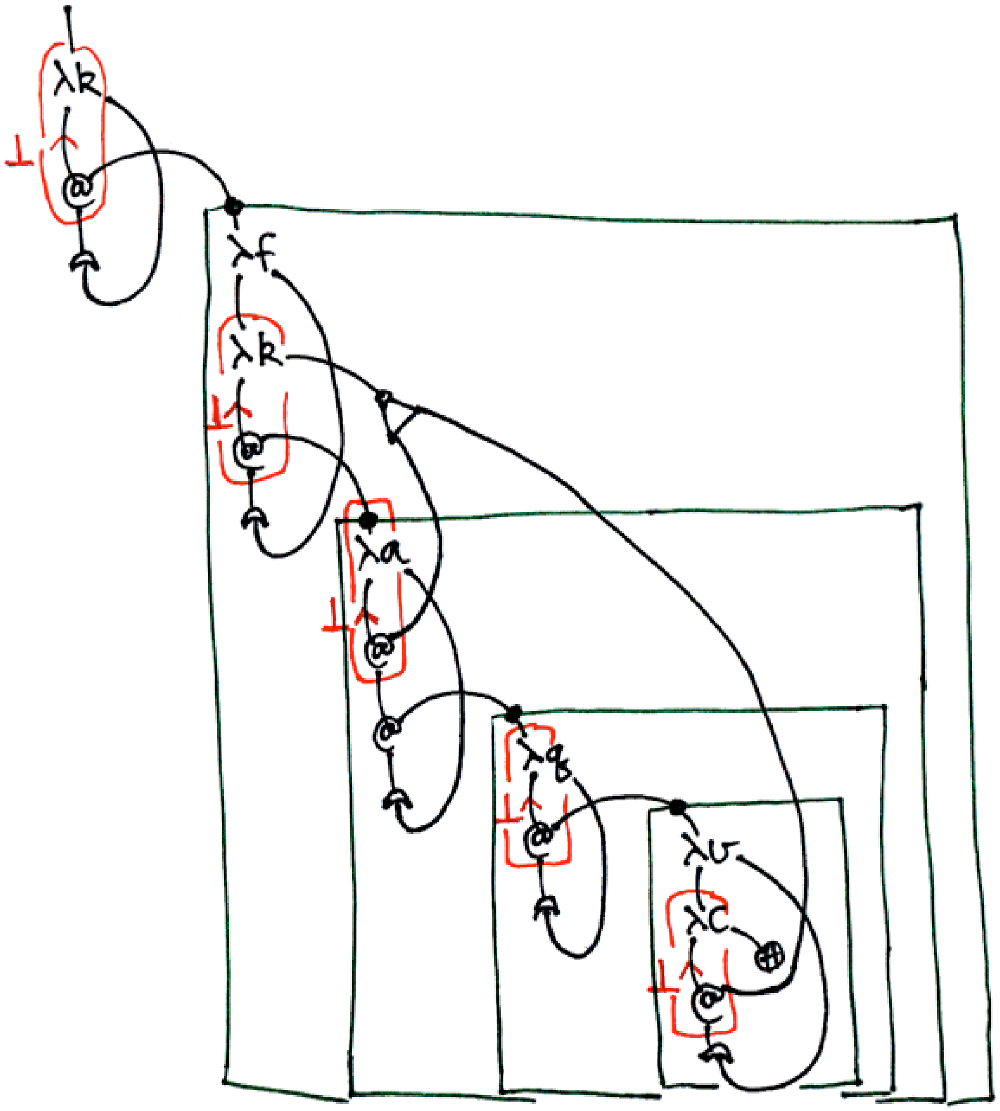


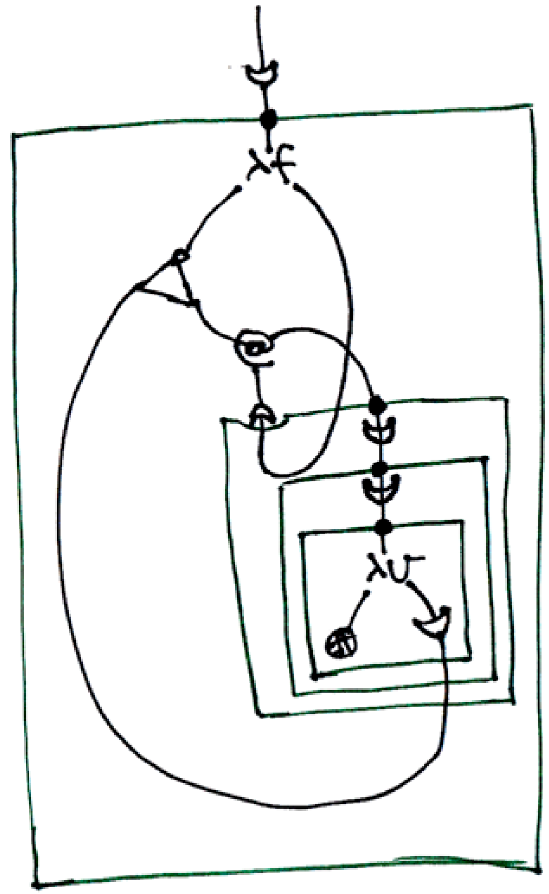
$$[\![\mathrm{call/cc}]\!] = \lambda k.\, k\,(\lambda f.\lambda k.\, f\,(\lambda v.\lambda c.\, k\, v)\; k)$$



more interesting:

$\langle\text{call/cc}\rangle = \lambda k.\, k(\lambda f.\lambda k.\, f(\lambda a.\, a(\lambda q.\, q(\lambda v.\lambda c.\, v\, k))\ k))$

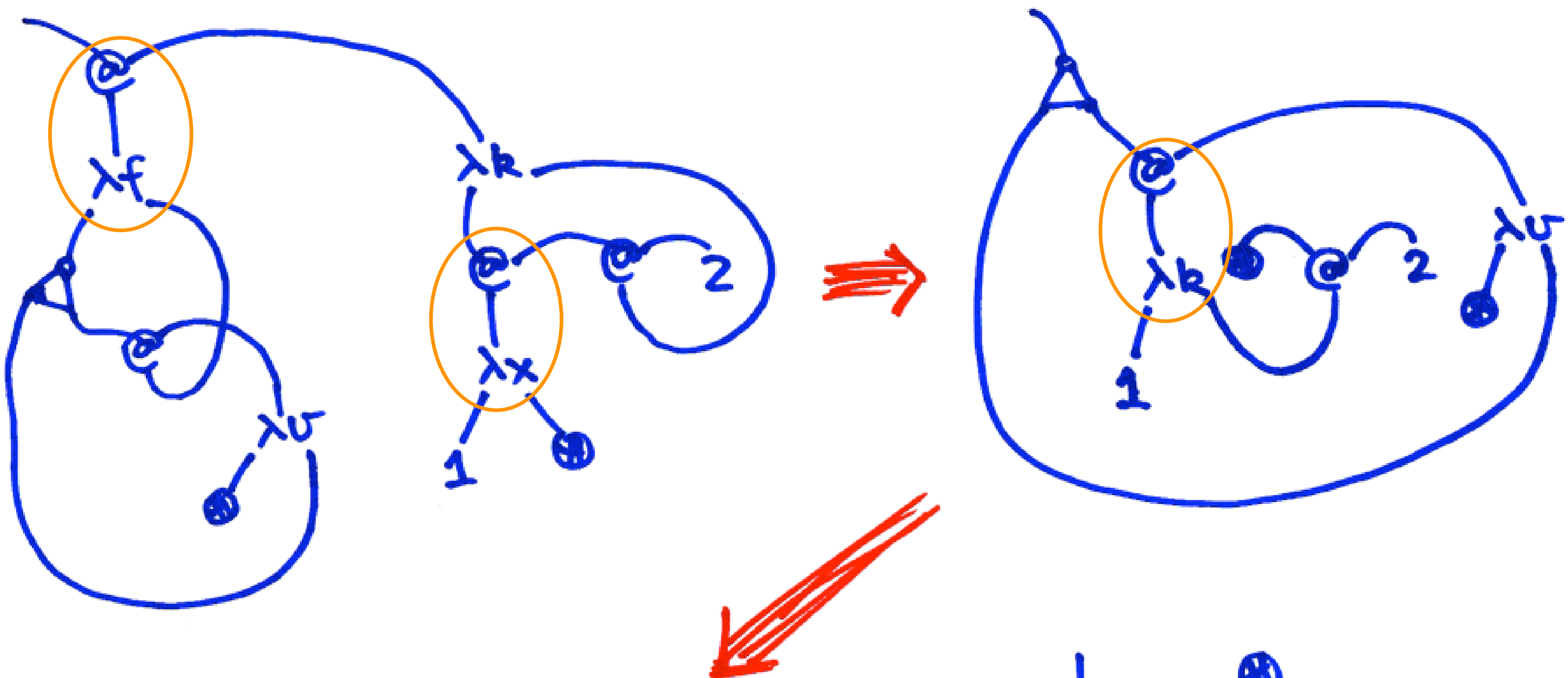


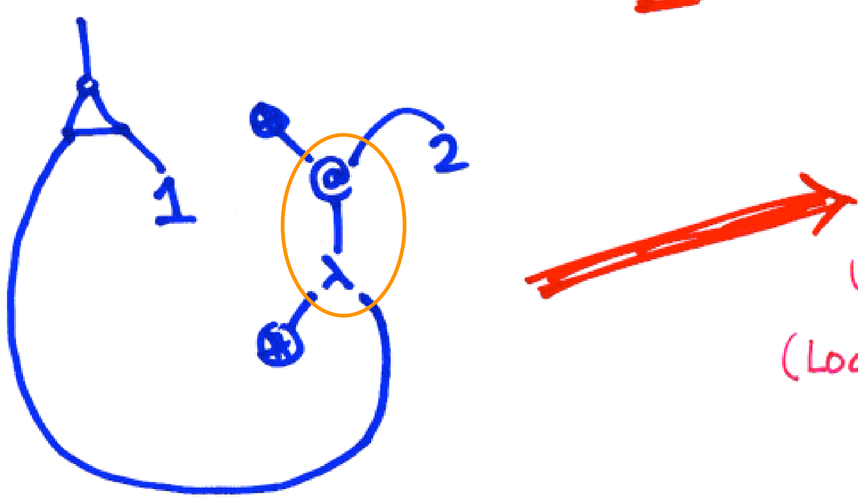


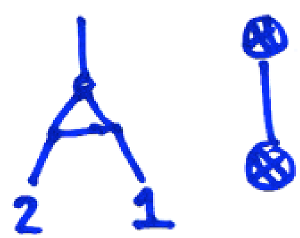
(same picture as in CBV, save boxes, ⊕)

call/cc $\lambda k. (\lambda x.1)(k\ 2)$

with invisible boxes — nothing interesting is shared



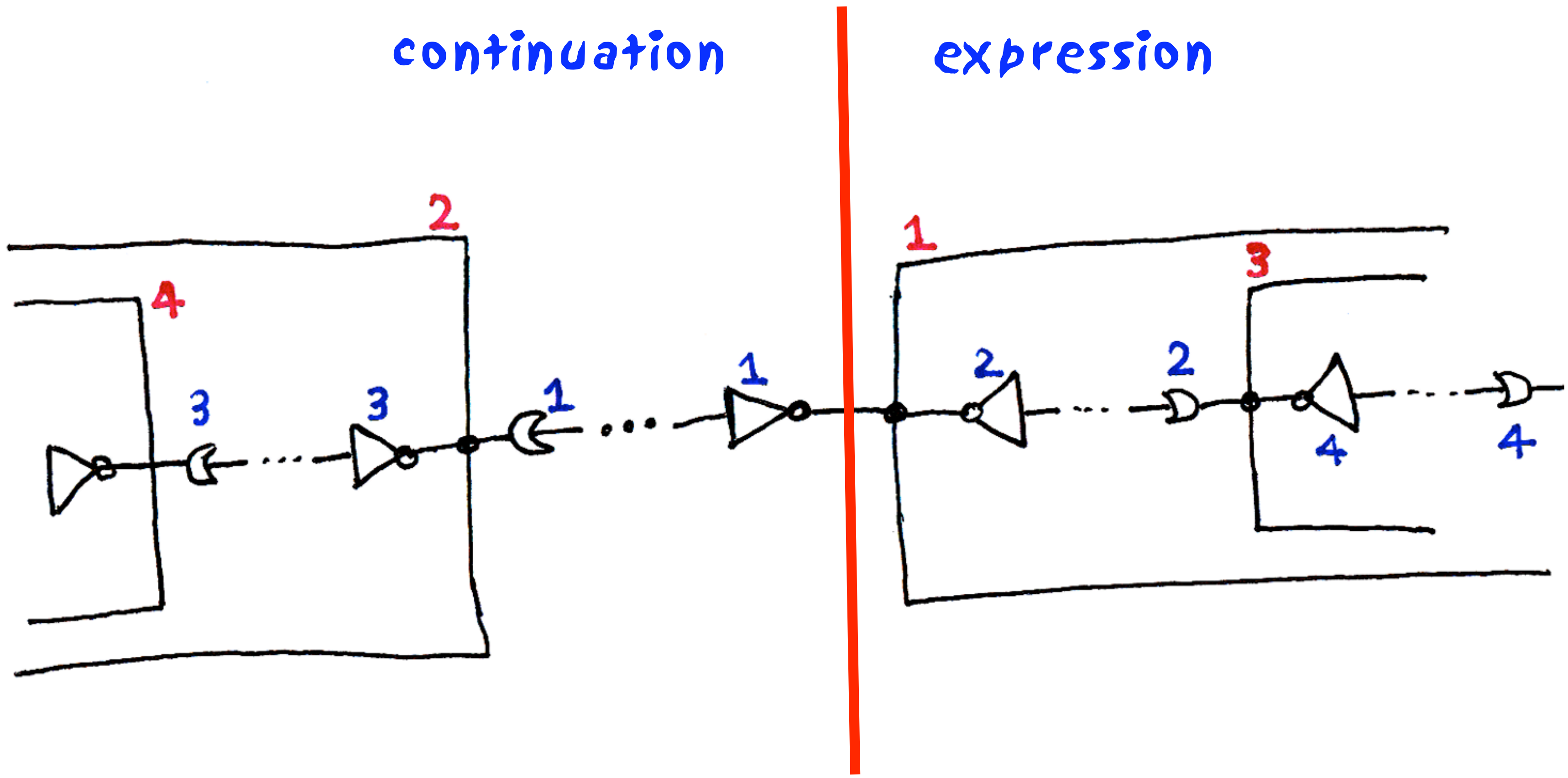




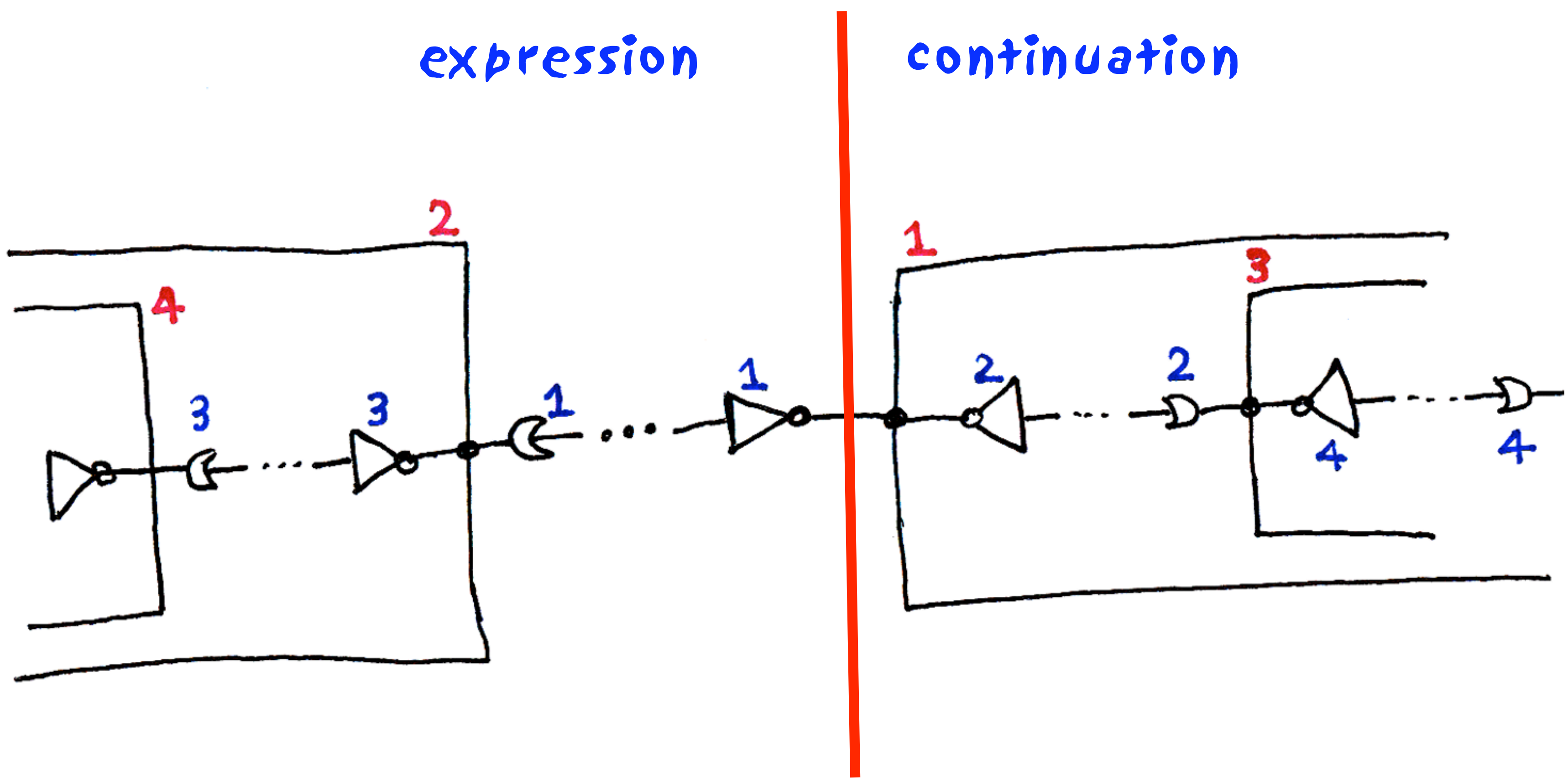
which is the CBV answer?

(Look at boxes, control operators)

Alternation of copying



Alternation of copying



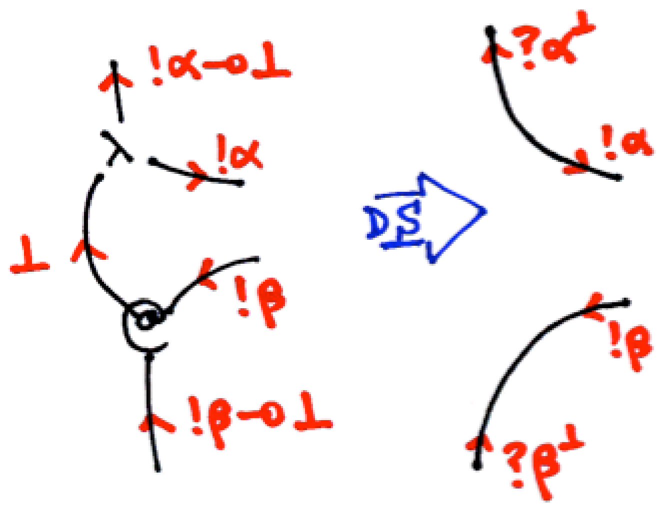
either way – it only matters who goes first... but for CPS-converted terms, evaluation order doesn't matter

Linearity of continuations vs. nonlinearity of continuation-passing

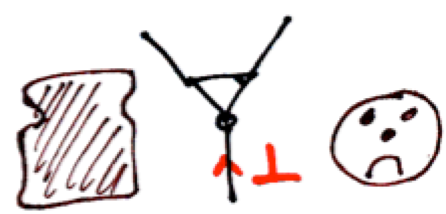
(Remark of Laurent & Regnier, About translations of CL into LLP)

call/cc has nonlinear CBx CPS translation (normal, escape returns)

... but continuations (answer type) are linear



If the continuation is shared, you are toast



La logique linéaire polarisée à l'américaine

(with apologies...)

?

$$P ::= \; !X \mid \; !N \mid P \otimes P \mid 1$$

$$N ::= \; ?X^{\perp} \mid \; ?P \mid N \parr N \mid \bot$$

Choose one for expressions, the other for continuations...

CBN $!A \multimap B$ boxing $\langle \neg\neg\tau \rangle = \; !?\langle \tau \rangle$ $+$

CBV $!(A \multimap B)$ boxing $\langle\!\langle \neg\neg\tau \rangle\!\rangle = \; ?!\langle\!\langle \tau \rangle\!\rangle$ $-$

$\lambda\mu$-calculus, Filinski symmetric λ-calculus, Curien-Herbelin dual calculus, &c.

- Translate into λ-calculus using CBx CPS transform
- Translate into MELL using CBy $(!A \multimap B, !(A \multimap B))$ embedding
- Compute direct-style variant of MELL proofnet by eliminating $\perp$-edges (and administrative redexes)

Filinski's symmetric lambda calculus (1989): CBV CPS

(now translate this into MELL, apply direct-style transformation...)

$$
\begin{array}{rcl}
Val & = & Basic + Unit() + Pair(Val \times Val) + In_1(Val) + In_2(Val) + \\
 & & Closr(Val \to Cnt \to Ans) + Contx(Val \times Cnt) \\
Cnt & = & Val \to Ans \\
Env & = & Ide \to (Val + Cnt)
\end{array}
$$

$$
\begin{array}{rcl}
\mathcal{E} : E \to Env \to Cnt & \to & Ans \\
\mathcal{E}[\![cst]\!]\rho\kappa & = & \kappa\ cst \\
\mathcal{E}[\![x]\!]\rho\kappa & = & \text{let } val(v) = \rho\ x \text{ in } \kappa\ v \\
\mathcal{E}[\![(E_1, E_2)]\!]\rho\kappa & = & \mathcal{E}[\![E_1]\!]\rho\ (\lambda v_1.\mathcal{E}[\![E_2]\!]\rho\ (\lambda v_2.\kappa\ pair(v_1, v_2))) \\
\mathcal{E}[\![()]\!]\rho\kappa & = & \kappa\ unit() \\
\mathcal{E}[\![F \uparrow E]\!]\rho\kappa & = & \mathcal{E}[\![E]\!]\rho\ (\lambda v.\mathcal{F}[\![F]\!]\rho v\kappa) \\
\mathcal{E}[\![\ulcorner F \urcorner]\!]\rho\kappa & = & \kappa\ closr(\lambda vc.\mathcal{F}[\![F]\!]\rho vc) \\
 & & \\
\mathcal{C} : C \to Env \to Val & \to & Ans \\
\mathcal{C}[\![y]\!]\rho v & = & \text{let } cnt(\kappa) = \rho\ y \text{ in } \kappa\ v \\
\mathcal{C}[\![\{C_1, C_2\}]\!]\rho v & = & \text{case } v \text{ of } in_1(t) : \mathcal{C}[\![C_1]\!]\rho t \ [\!]\ in_2(t) : \mathcal{C}[\![C_2]\!]\rho t \text{ esac} \\
\mathcal{C}[\![\{\}]\!]\rho v & = & \text{case } v \text{ of esac} \\
\mathcal{C}[\![C \downarrow F]\!]\rho v & = & \mathcal{F}[\![F]\!]\rho v\ (\lambda t.\mathcal{C}[\![C]\!]\rho t) \\
\mathcal{C}[\![\llcorner F \lrcorner]\!]\rho v & = & \text{let } contx(a, c) = v \text{ in } \mathcal{F}[\![F]\!]\rho ac \\
 & & \\
\mathcal{F} : F \to Env \to Val \to Cnt & \to & Ans \\
\mathcal{F}[\![p]\!]\rho v\kappa & = & \kappa(pv) \\
\mathcal{F}[\![X \Rightarrow E]\!]\rho v\kappa & = & \mathcal{E}[\![E]\!]\ ([\mathcal{X}[\![X]\!] \mapsto v]\rho)\kappa \\
\mathcal{F}[\![Y \Leftarrow C]\!]\rho v\kappa & = & \mathcal{C}[\![C]\!]\ ([\mathcal{Y}[\![Y]\!] \mapsto \kappa]\rho)v \\
\mathcal{F}[\![\overline{E}]\!]\rho v\kappa & = & \mathcal{E}[\![E]\!]\rho\ (\lambda t.\text{let } closr(f) = t \text{ in } f v\kappa) \\
\mathcal{F}[\![\underline{C}]\!]\rho v\kappa & = & \mathcal{C}[\![C]\!]\rho\ contx(v, \kappa) \\
 & & \\
\mathcal{X} : X \to Val & \to & Env \to Env \\
[\mathcal{X}[\![x]\!] \mapsto v]\rho & = & [x \mapsto val(v)]\rho \\
[\mathcal{X}[\![()]\!] \mapsto v]\rho & = & \text{let } unit() = v \text{ in } \rho \\
[\mathcal{X}[\![(X_1, X_2)]\!] \mapsto v]\rho & = & \text{let } pair(v_1, v_2) = v \text{ in } [\mathcal{X}[\![X_1]\!] \mapsto v_1, \mathcal{X}[\![X_2]\!] \mapsto v_2]\rho \\
 & & \\
\mathcal{Y} : Y \to Cnt & \to & Env \to Env \\
[\mathcal{Y}[\![y]\!] \mapsto \kappa]\rho & = & [y \mapsto cnt(\kappa)]\rho \\
[\mathcal{Y}[\![\{\}]\!] \mapsto \kappa]\rho & = & \rho \\
[\mathcal{Y}[\![\{Y_1, Y_2\}]\!] \mapsto \kappa]\rho & = & [\mathcal{Y}[\![Y_1]\!] \mapsto (\lambda v.\kappa\ in_1(v)),\ \mathcal{Y}[\![Y_2]\!] \mapsto (\lambda v.\kappa\ in_2(v))]\rho
\end{array}
$$

Filinski's symmetric lambda calculus

Identity group

$$\frac{}{\alpha \;\blacktriangleright\; \underline{A} \vdash \alpha : A}\,(\text{Ax-L}) \qquad \frac{}{x \;\blacktriangleright\; x : A \vdash \underline{A}}\,(\text{Ax-R}) \qquad \frac{M \;\blacktriangleright\; \Gamma \vdash \Delta, \underline{A} \quad K \;\blacktriangleright\; \underline{A}, \Sigma \vdash \Lambda}{M \bullet K \;\blacktriangleright\; \Gamma, \Sigma \vdash \Delta, \Lambda}\,(\textsc{Cut})$$

Logical rules

$$\frac{M \;\blacktriangleright\; \Gamma \vdash \underline{A}, \Delta \quad N \;\blacktriangleright\; \Sigma \vdash \underline{B}, \Lambda}{\langle M, N\rangle \;\blacktriangleright\; \Gamma, \Sigma \vdash \underline{A \wedge B}, \Delta, \Lambda}\,(\wedge\text{R}) \qquad \frac{K \;\blacktriangleright\; \underline{A}, \Gamma \vdash \Delta}{\mathsf{fst}[K] \;\blacktriangleright\; \underline{A \wedge B}, \Gamma \vdash \Delta}\,(\wedge\text{L}_1) \qquad \frac{L \;\blacktriangleright\; \underline{B}, \Gamma \vdash \Delta}{\mathsf{snd}[L] \;\blacktriangleright\; \underline{A \wedge B}, \Gamma \vdash \Delta}\,(\wedge\text{L}_2)$$

$$\frac{K \;\blacktriangleright\; \Lambda, \underline{B} \vdash \Sigma \quad L \;\blacktriangleright\; \Delta, \underline{A} \vdash \Gamma}{[K, L] \;\blacktriangleright\; \Lambda, \Delta, \underline{B \vee A} \vdash \Sigma, \Gamma}\,(\vee\text{L}) \qquad \frac{M \;\blacktriangleright\; \Delta \vdash \Gamma, \underline{A}}{\langle M\rangle\mathsf{inr} \;\blacktriangleright\; \Delta \vdash \Gamma, \underline{B \vee A}}\,(\vee\text{R}_2) \qquad \frac{N \;\blacktriangleright\; \Delta \vdash \Gamma, \underline{B}}{\langle N\rangle\mathsf{inl} \;\blacktriangleright\; \Delta \vdash \Gamma, \underline{B \vee A}}\,(\vee\text{R}_1)$$

$$\frac{K \;\blacktriangleright\; \Gamma, \underline{A} \vdash \Delta}{[K]\mathsf{not} \;\blacktriangleright\; \Gamma \vdash \underline{\neg A}, \Delta}\,(\neg\text{R}) \qquad \frac{M \;\blacktriangleright\; \Delta \vdash \underline{A}, \Gamma}{\mathsf{not}\langle M\rangle \;\blacktriangleright\; \Delta, \underline{\neg A} \vdash \Gamma}\,(\neg\text{L})$$

$$\frac{M \;\blacktriangleright\; x : A, \Gamma \vdash \underline{B}, \Delta}{\lambda x.M \;\blacktriangleright\; \Gamma \vdash \underline{A \to B}, \Delta}\,(\to\text{R}) \qquad \frac{M \;\blacktriangleright\; \Gamma \vdash \underline{A}, \Delta \quad K \;\blacktriangleright\; \Sigma, \underline{B} \vdash \Lambda}{M @ K \;\blacktriangleright\; \Gamma, \Sigma, \underline{A \to B} \vdash \Delta, \Lambda}\,(\to\text{L})$$

$$\frac{M \;\blacktriangleright\; \Lambda \vdash \underline{B}, \Sigma \quad K \;\blacktriangleright\; \Delta, \underline{A} \vdash \Gamma}{K @^{\circ} M \;\blacktriangleright\; \Lambda, \Delta \vdash \underline{B \leftarrow A}, \Sigma, \Gamma}\,(\leftarrow\text{R}) \qquad \frac{K \;\blacktriangleright\; \Delta, \underline{B} \vdash \Gamma, \alpha : A}{\lambda^{\circ}\alpha.K \;\blacktriangleright\; \Delta, \underline{B \leftarrow A} \vdash \Gamma}\,(\leftarrow\text{L})$$

$$\frac{B \leftarrow A \vdash \quad A \vdash}{B \vdash}$$

Duality, via CPS transformations...

$$
\begin{array}{rclcrcl}
(\langle M, N\rangle)^\circ & = & [M^\circ, N^\circ] & \qquad & ([K, L])^\circ & = & \langle K^\circ, L^\circ\rangle \\
(\langle M\rangle\mathsf{inl})^\circ & = & \mathsf{fst}[M^\circ] & & (\mathsf{fst}[K])^\circ & = & \langle K^\circ\rangle\mathsf{inl} \\
(\langle N\rangle\mathsf{inr})^\circ & = & \mathsf{snd}[N^\circ] & & (\mathsf{snd}[L])^\circ & = & \langle L^\circ\rangle\mathsf{inr} \\
([K]\mathsf{not})^\circ & = & \mathsf{not}\langle K^\circ\rangle & & (\mathsf{not}\langle M\rangle)^\circ & = & [M^\circ]\mathsf{not} \\
((S).\alpha)^\circ & = & \alpha^\circ.(S^\circ) & & (x.(S))^\circ & = & ((S^\circ).x^\circ) \\
(\lambda x.M)^\circ & = & \lambda^\circ x^\circ.M^\circ & & (\lambda^\circ \alpha.K)^\circ & = & \lambda\alpha^\circ.K^\circ \\
(K @^\circ M)^\circ & = & K^\circ @ M^\circ & & (M @ K)^\circ & = & M^\circ @^\circ K^\circ
\end{array}
$$

$$(M \bullet K)^\circ = K^\circ \bullet M^\circ$$

Duality, via CPS transformations... (2)

Call-by-value CPS translation

$$
\begin{aligned}
x^v &= \lambda\gamma.\gamma x \\
(\lambda x.M)^v &= \lambda\gamma.\gamma(\lambda x.\lambda\kappa.M^v\kappa) \\
(K @^\circ M)^v &= \lambda\gamma.\gamma(\lambda z.M^v(zK^v)) \\
(\langle M, N\rangle)^v &= \lambda\gamma.M^v(\lambda m.N^v(\lambda n.\gamma\langle m, n\rangle)) \\
(\langle M\rangle\mathsf{inr})^v &= \lambda\gamma.M^v(\lambda m.\gamma(\mathsf{inr}\ m)) \\
(\langle M\rangle\mathsf{inl})^v &= \lambda\gamma.M^v(\lambda m.\gamma(\mathsf{inl}\ m)) \\
([K]\mathsf{not})^v &= \lambda\gamma.\gamma(\lambda z.K^v z) =_\eta \lambda\gamma.\gamma K^v \\
((S).\alpha)^v &= \lambda\alpha.S^v
\end{aligned}
$$

$$
\begin{aligned}
\alpha^v &= \lambda z.\alpha z \\
(\lambda^\circ\alpha.K)^v &= \lambda z.z(\lambda\alpha.\lambda x.K^v x) \\
(M @ K)^v &= \lambda z.M^v(\lambda m.zmK^v) \\
([K, L])^v &= \lambda z.\mathsf{case}\ z\ \mathsf{of}\ \mathsf{inl}\ x \Rightarrow K^v x, \mathsf{inr}\ x \Rightarrow L^v y \\
(\mathsf{fst}[K])^v &= \lambda z.\mathsf{case}\ z\ \mathsf{of}\ \langle x, -\rangle \Rightarrow K^v x \\
(\mathsf{snd}[L])^v &= \lambda z.\mathsf{case}\ z\ \mathsf{of}\ \langle -, y\rangle \Rightarrow L^v y \\
(\mathsf{not}\langle M\rangle)^v &= \lambda z.(\lambda\gamma.M^v\gamma)z =_\eta M^v \\
(x.(S))^v &= \lambda x.S^v
\end{aligned}
$$

$$(M \bullet K)^v = M^v K^v$$

Call-by-name CPS translation

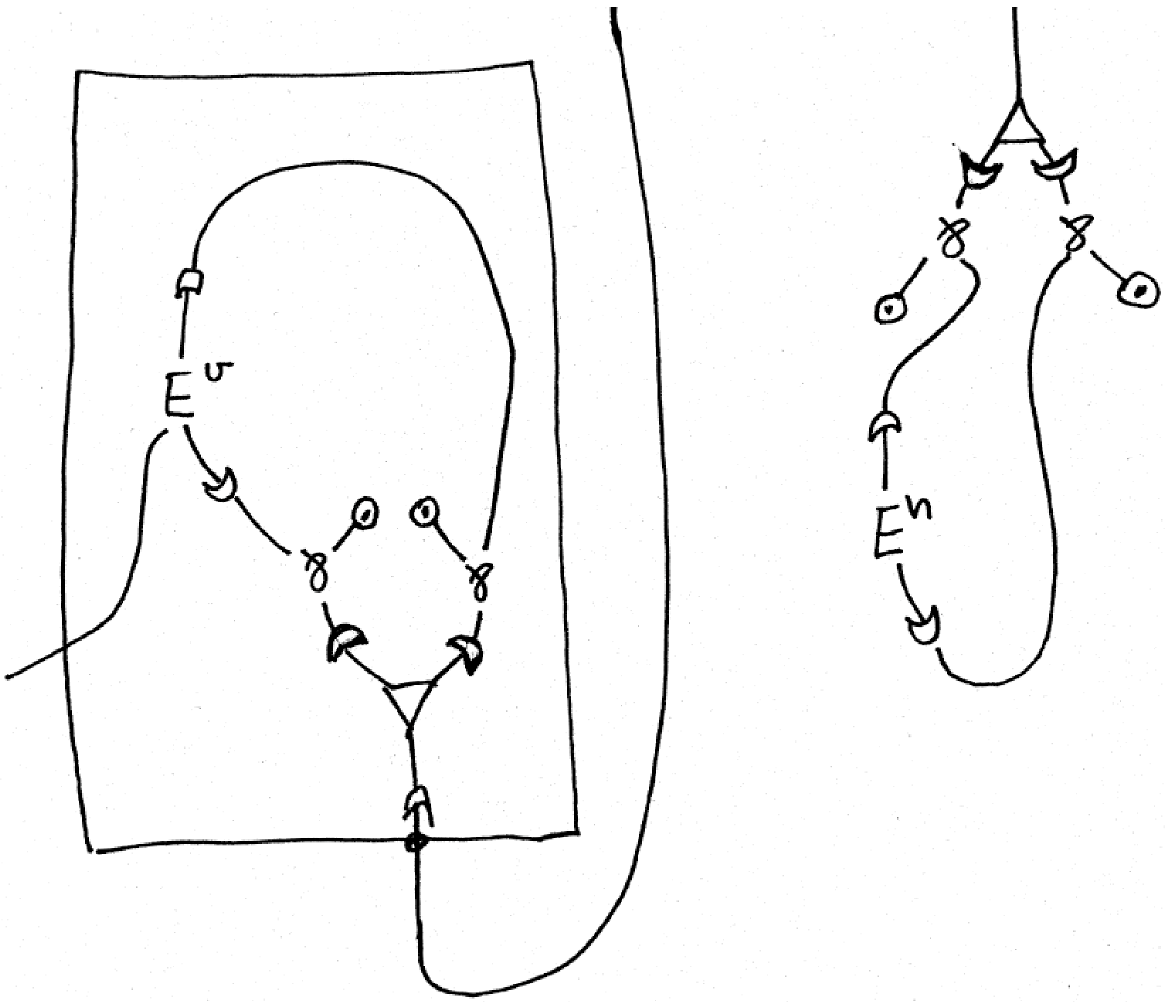
$$
\begin{aligned}
\alpha^n &= \lambda z.z\alpha \\
(\lambda^\circ\alpha.K)^n &= \lambda z.z(\lambda\alpha.\lambda x.K^n x) \\
(M @ K)^n &= \lambda z.z(\lambda\alpha.K^n(\alpha M^n)) \\
([K, L])^n &= \lambda z.K^n(\lambda\alpha.L^n(\lambda\beta.z\langle\alpha, \beta\rangle)) \\
(\mathsf{fst}[K])^n &= \lambda z.K^n(\lambda\alpha.z(\mathsf{inl}\ \alpha)) \\
(\mathsf{snd}[L])^n &= \lambda z.L^n(\lambda\beta.z(\mathsf{inr}\ \beta)) \\
(\mathsf{not}\langle M\rangle)^n &= \lambda z.z(\lambda\gamma.M^n\gamma) =_\eta \lambda z.zM^n \\
(x.(S))^n &= \lambda x.S^n
\end{aligned}
$$

$$
\begin{aligned}
x^n &= \lambda\gamma.x\gamma \\
(\lambda x.M)^n &= \lambda\gamma.\gamma(\lambda x.\lambda\kappa.M^n\kappa) \\
(K @^\circ M)^n &= \lambda\gamma.K^n(\lambda\alpha.\gamma\alpha M^n) \\
(\langle M, N\rangle)^n &= \lambda\gamma.\mathsf{case}\ \gamma\ \mathsf{of}\ \mathsf{inl}\ \alpha \Rightarrow M^n\alpha, \mathsf{inr}\ \beta \Rightarrow N^n\beta \\
(\langle M\rangle\mathsf{inr})^n &= \lambda\gamma.\mathsf{case}\ \gamma\ \mathsf{of}\ \langle\alpha, -\rangle \Rightarrow M^n\alpha \\
(\langle M\rangle\mathsf{inl})^n &= \lambda\gamma.\mathsf{case}\ \gamma\ \mathsf{of}\ \langle -, \beta\rangle \Rightarrow N^n\beta \\
([K]\mathsf{not})^n &= \lambda\gamma.(\lambda z.K^n z)\gamma =_\eta K^n \\
((S).\alpha)^n &= \lambda\alpha.S^n
\end{aligned}
$$

$$(M \bullet K)^n = K^n M^n$$

“Call by value is dual to call by name” (Wadler, ICFP2003)

When λ and @ are not primitives



call-by-value

call-by-name

Duality of proofnets (Filinski symmetric λ-calculus)

	CBV CPS		CBN CPS	
	CBV boxing	CBN boxing	CBV boxing	CBN boxing
Expressions	+	−	+	−
Continuations	−	+	−	+

Observation:

CBx CPS, CBy boxing on expressions

dual to

CB$\bar{x}$ CPS, CB$\bar{y}$ boxing on continuations

Paths...

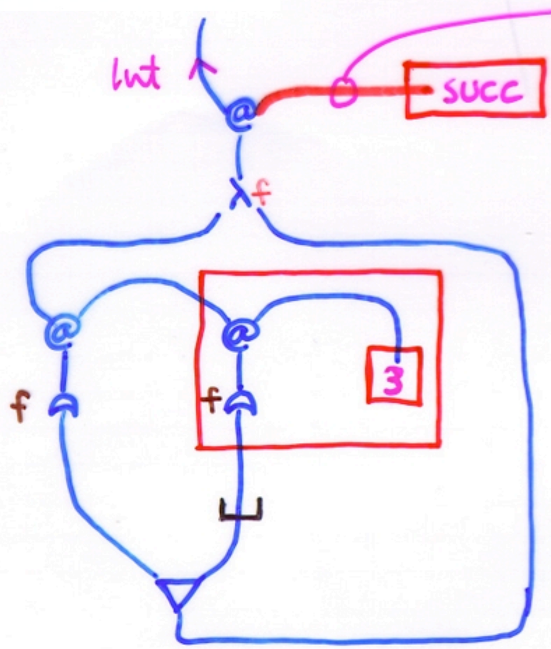
If confluence means we get the
CBV and the CBN values of

call/cc λk.(λx.1)(k 2),

then how do we extract the appropriate
answer? Which one is it?

Geometry of Interaction: $\lambda f. f(f\,3)$ vs. succ

Information flow across this wire



C_1: Lf, ∘? Left f asks succ, "What's the output?"

S_1: Lf, •α, ? succ responds, "what's the input?"

C_2: R⟨α, f⟩, ∘? Right f asks succ, "What's the output?"

S_2: R⟨α, f⟩, •α, ? "What's the input?"

C_3: R⟨α, f⟩, •α, 3 "The input is 3." (for the second call)

S_3: R⟨α, f⟩, ∘4 "The output is 4."

C_4: Lf, •α, 4 "The input is 4." (for the first call)

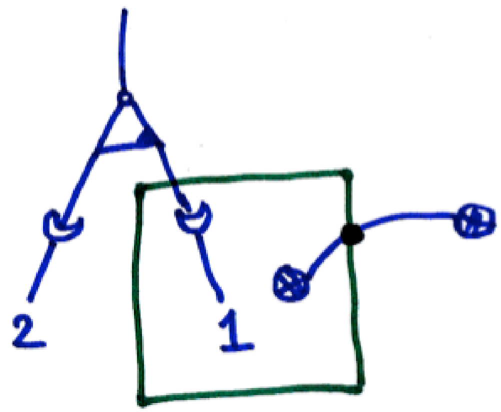
S_4: Lf, ∘5 "The output is 5."

Paths — extracting the CBV or CBN answer

CBV CPS
CBN boxing

call/cc $\lambda k.(\lambda x.1)(k\,2)$

normal form:



"When you come to a fork
in the road, take it."

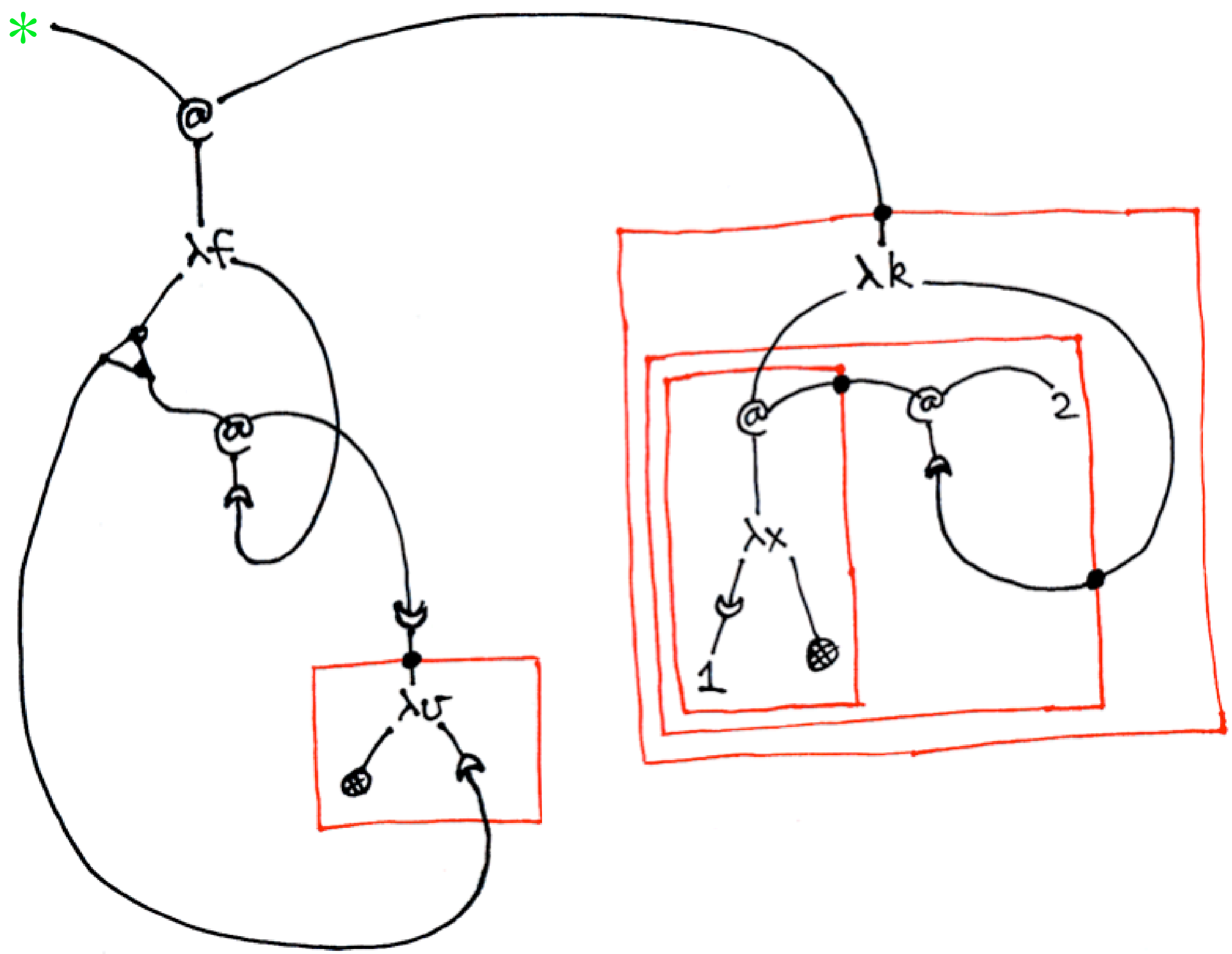
— Yogi Berra

"Two roads diverged in a yellow wood
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent..."

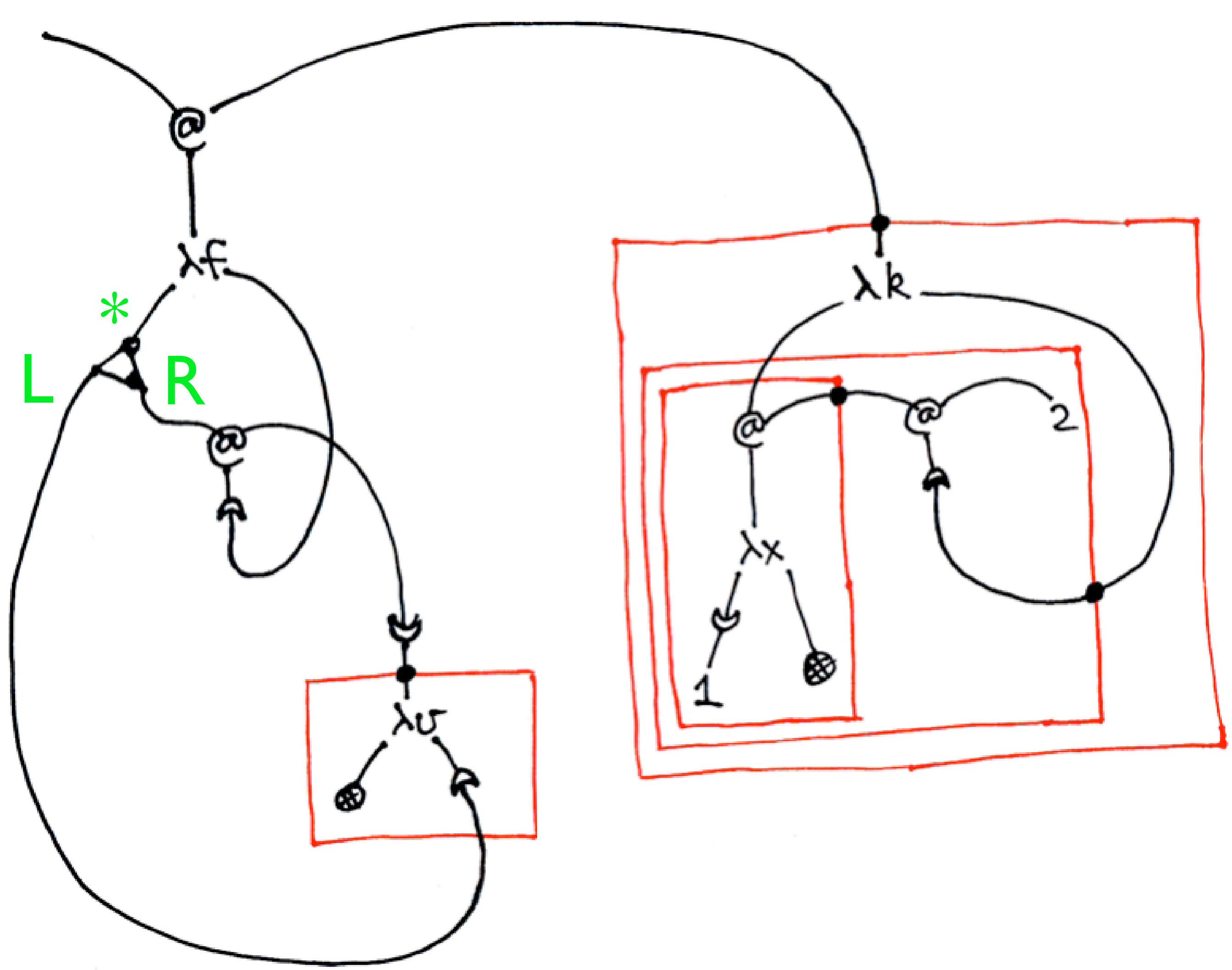
— Robert Frost
The Road Not Taken
(1915)

CPS (boxing) tells us where to go...

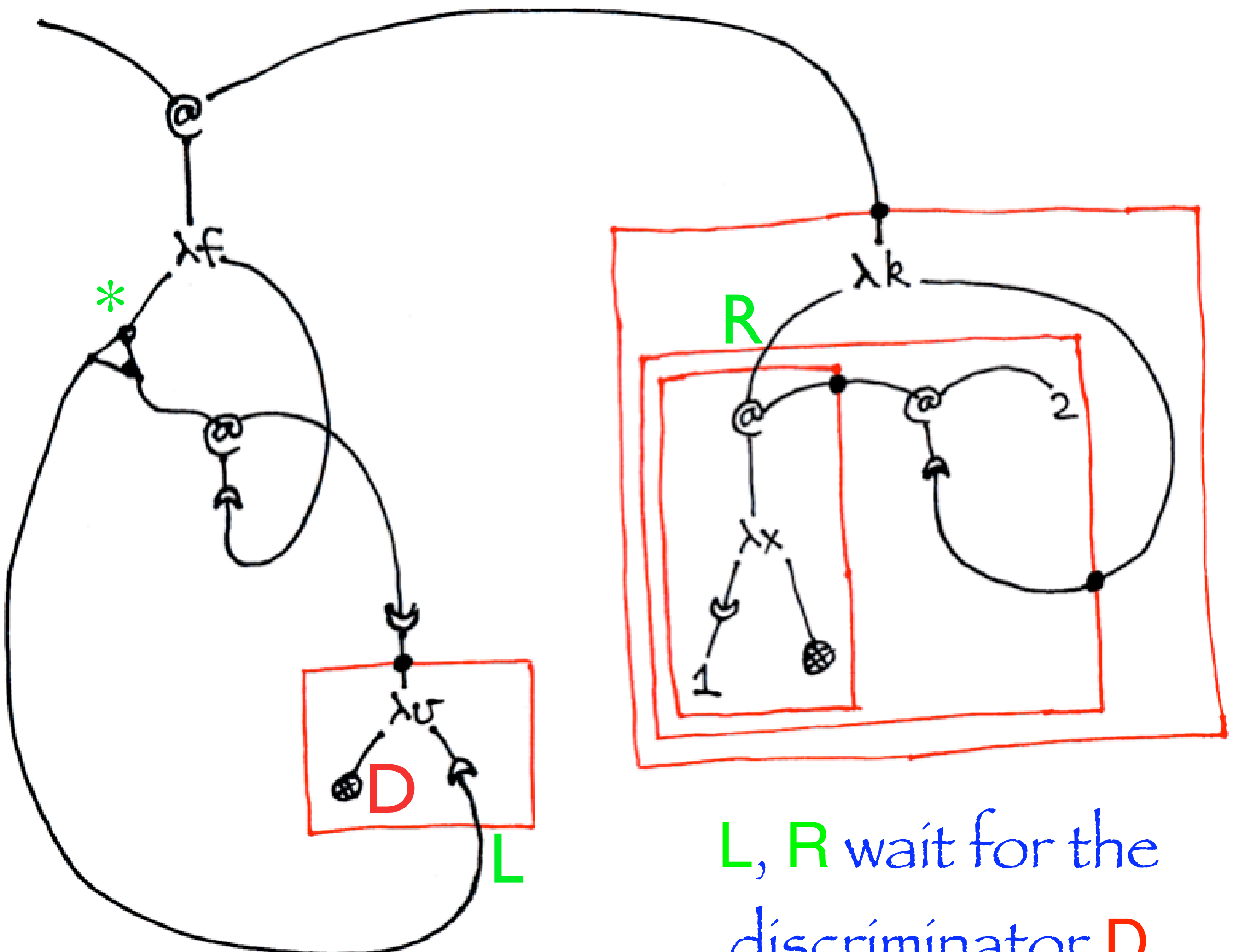
CBV CPS / CBN boxing / call/cc $\lambda k.\,(\lambda x.1)(k\,2)$



CBV CPS / CBN boxing / call/cc λk. (λx.1) (k 2)

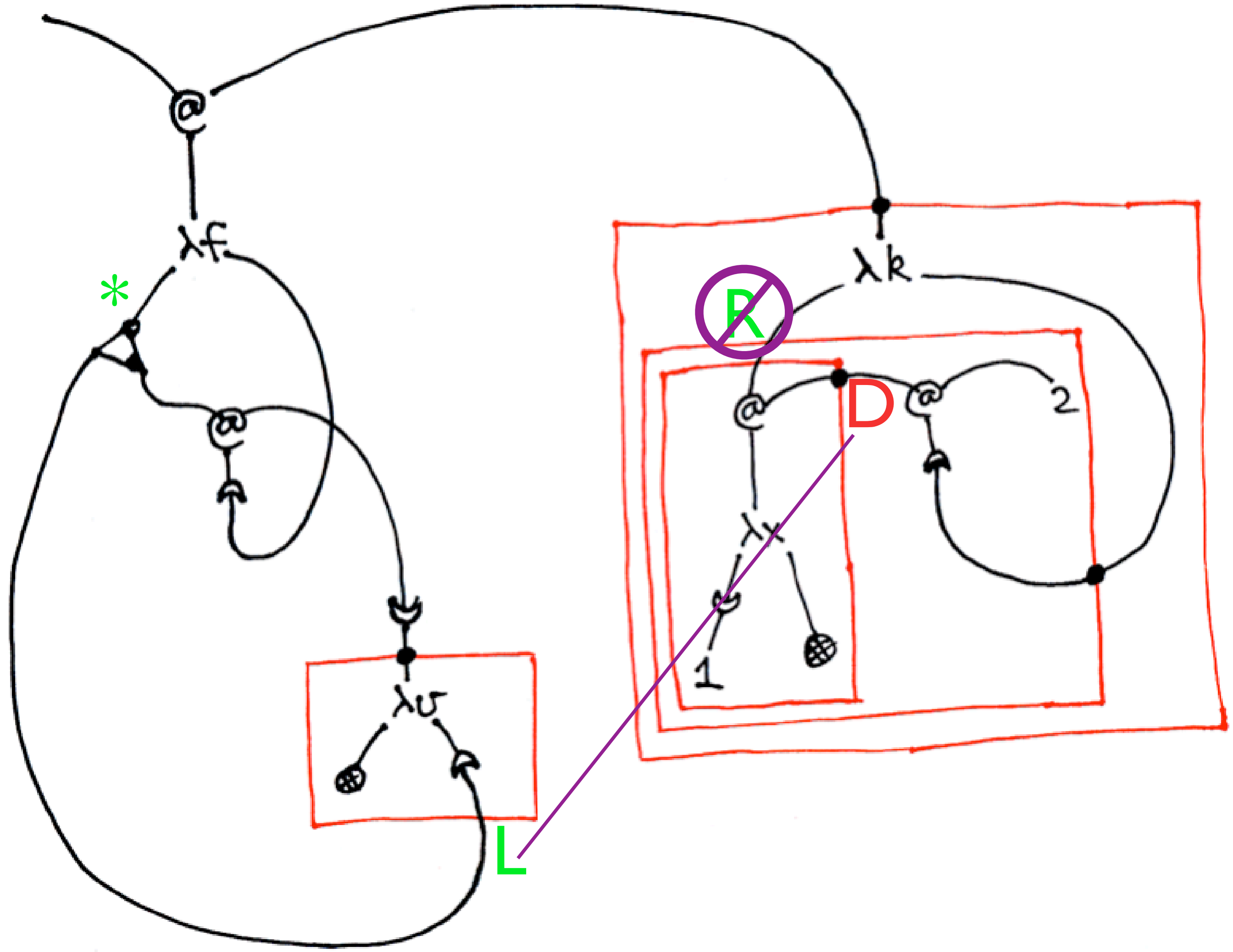


CBV CPS / CBN boxing / call/cc λk. (λx.1)(k 2)

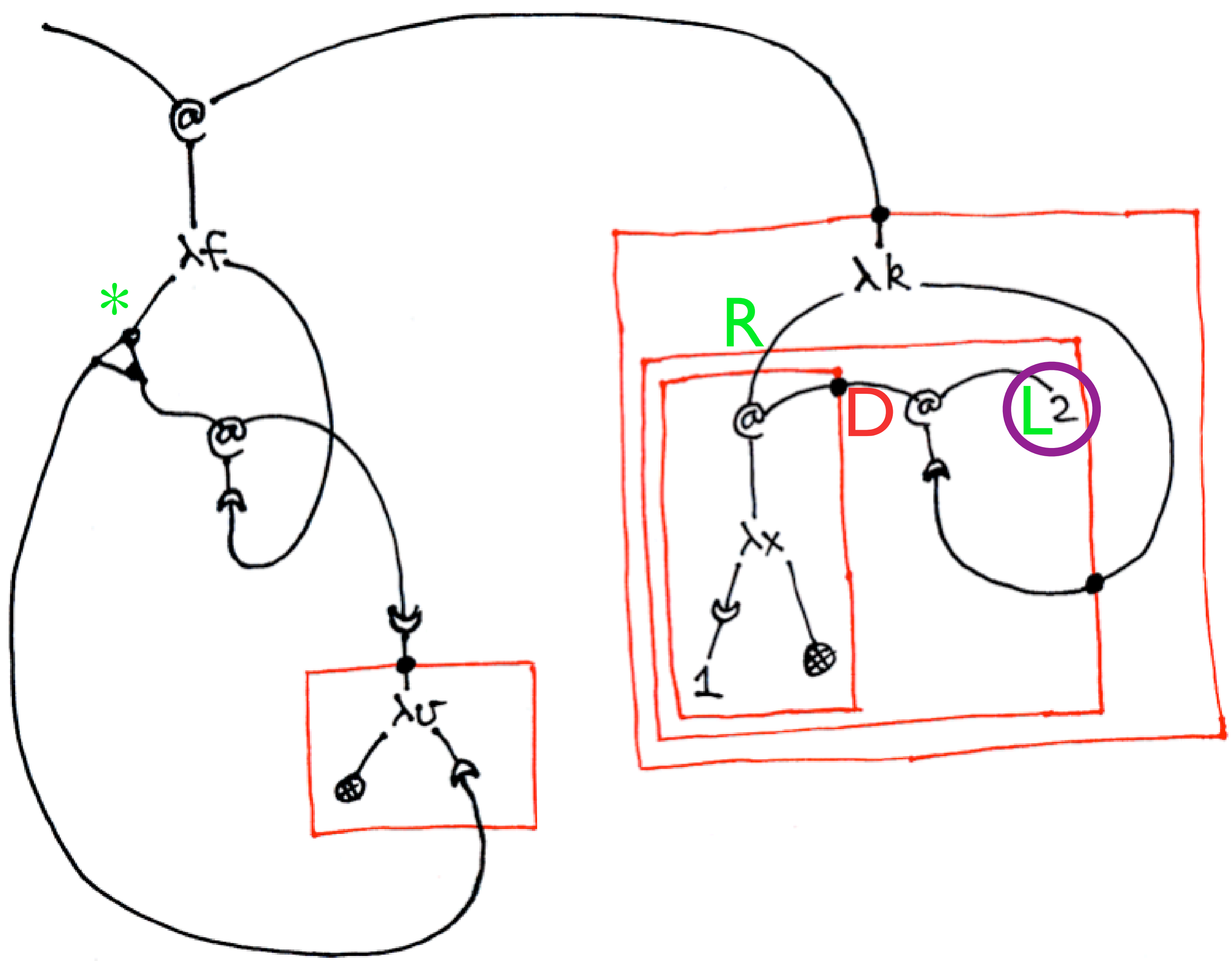


L, R wait for the discriminator D

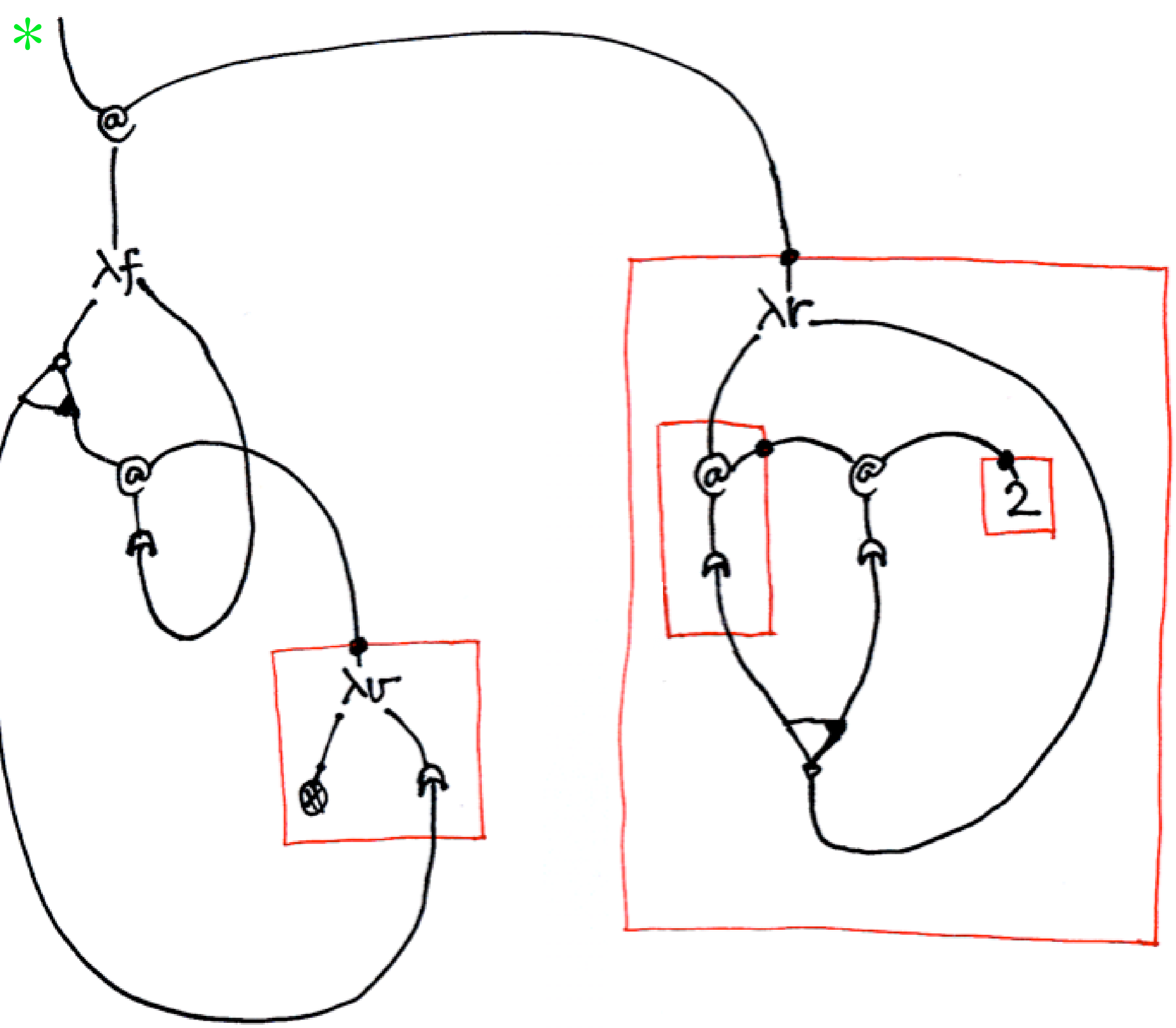
CBV CPS / CBN boxing / call/cc λk. (λx.1)(k 2)



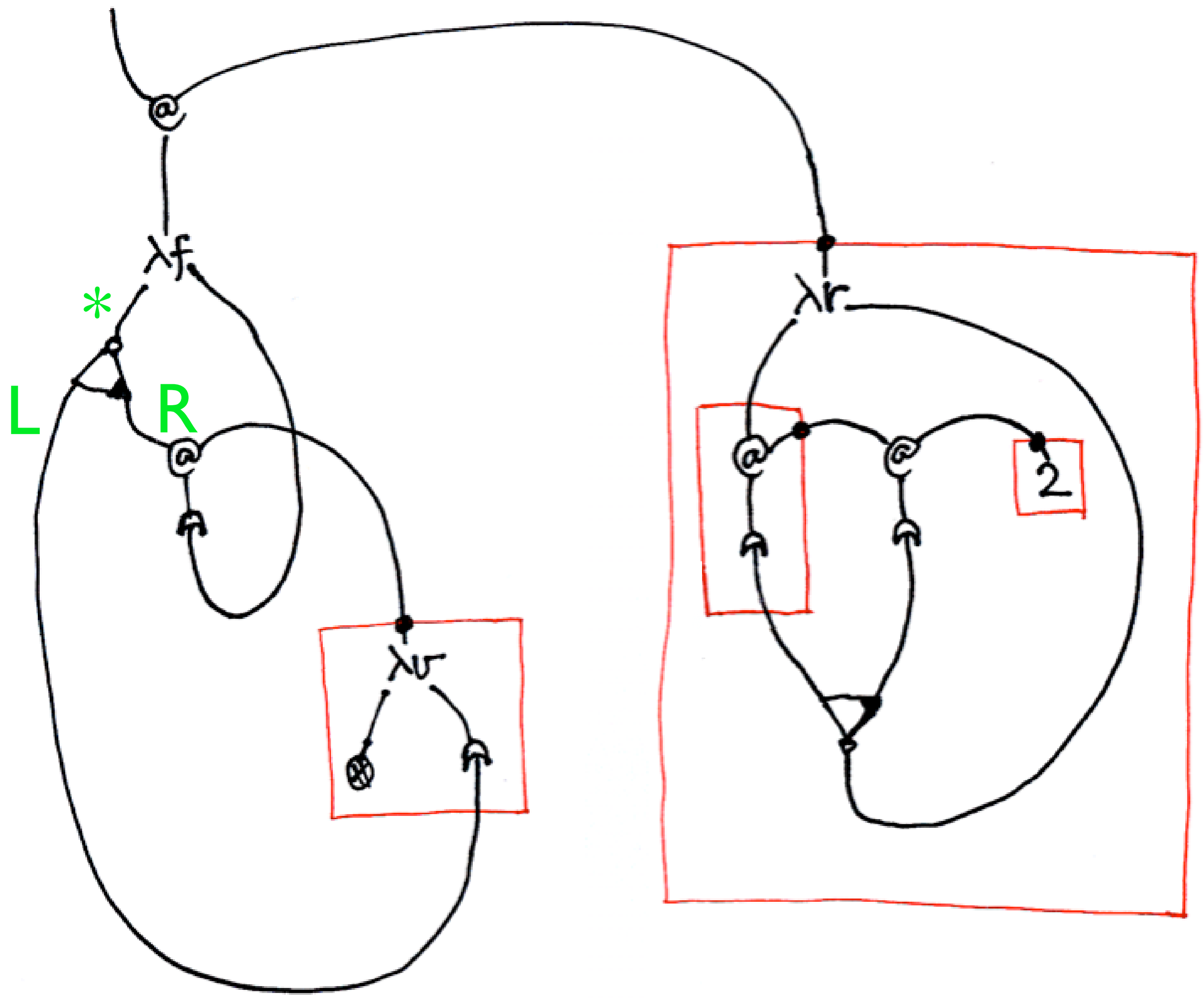
CBV CPS / CBN boxing / call/cc λk. (λx.1) (k 2)



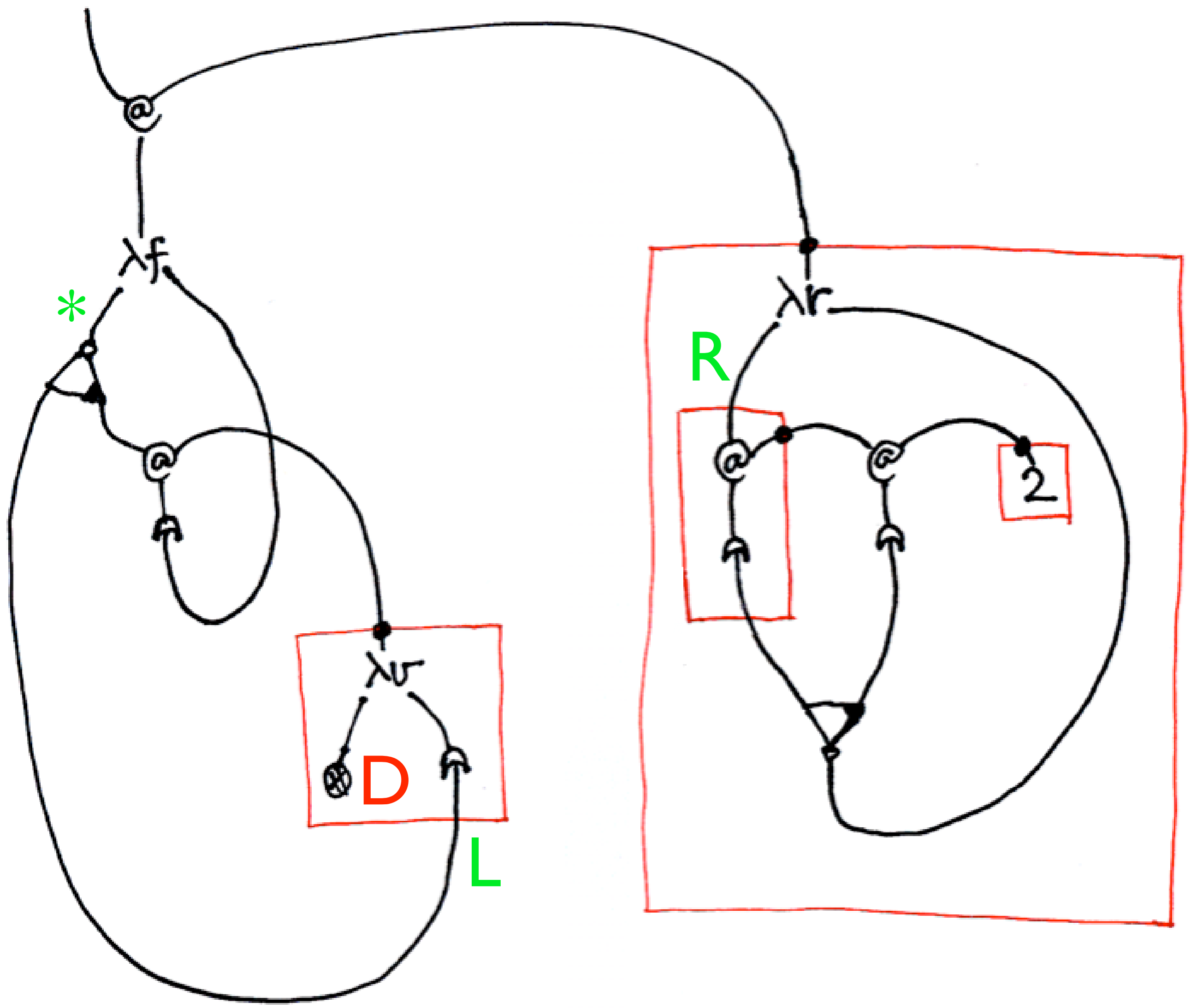
call/cc λr. r(r2)



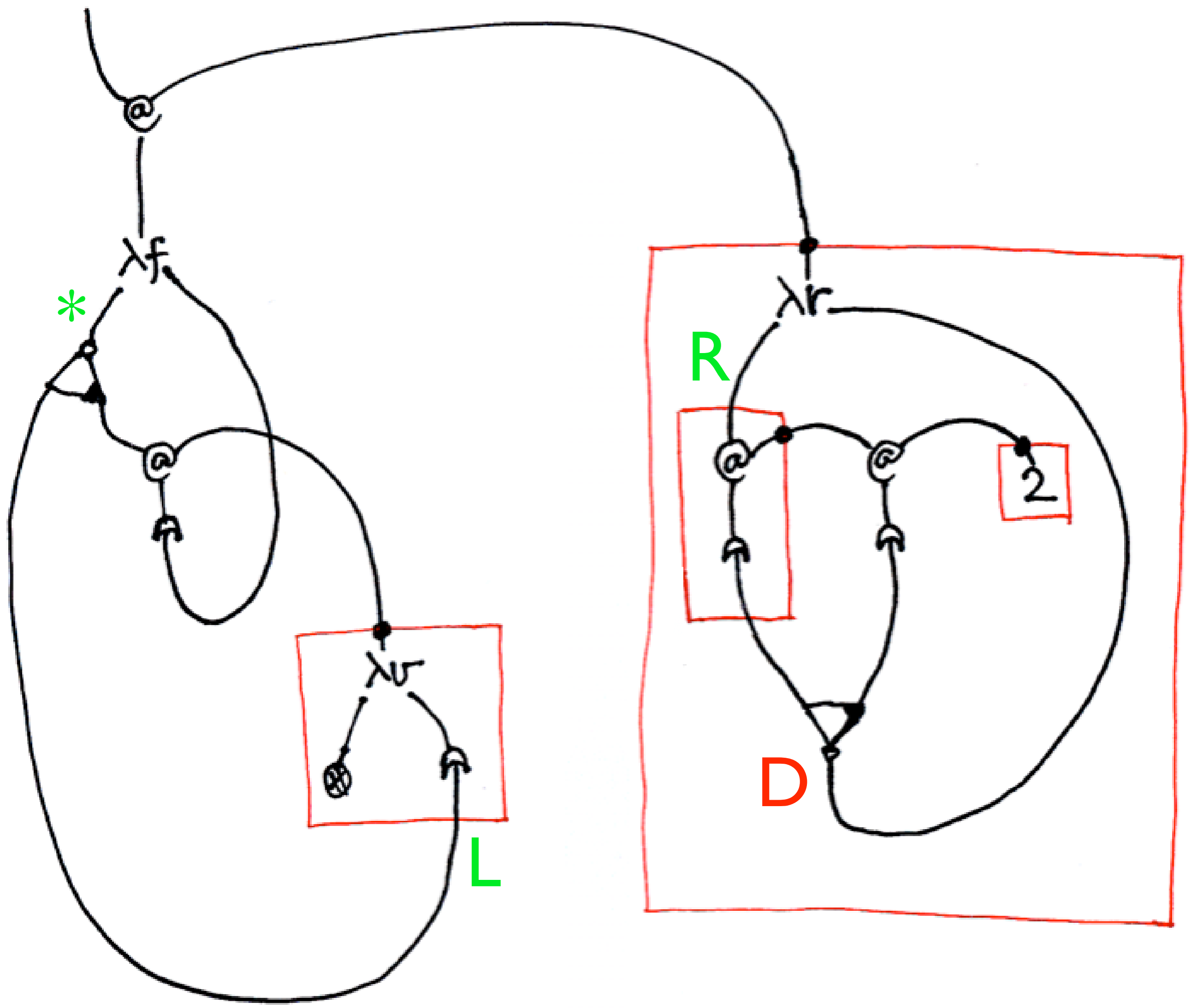
call/cc λr. r(r 2)



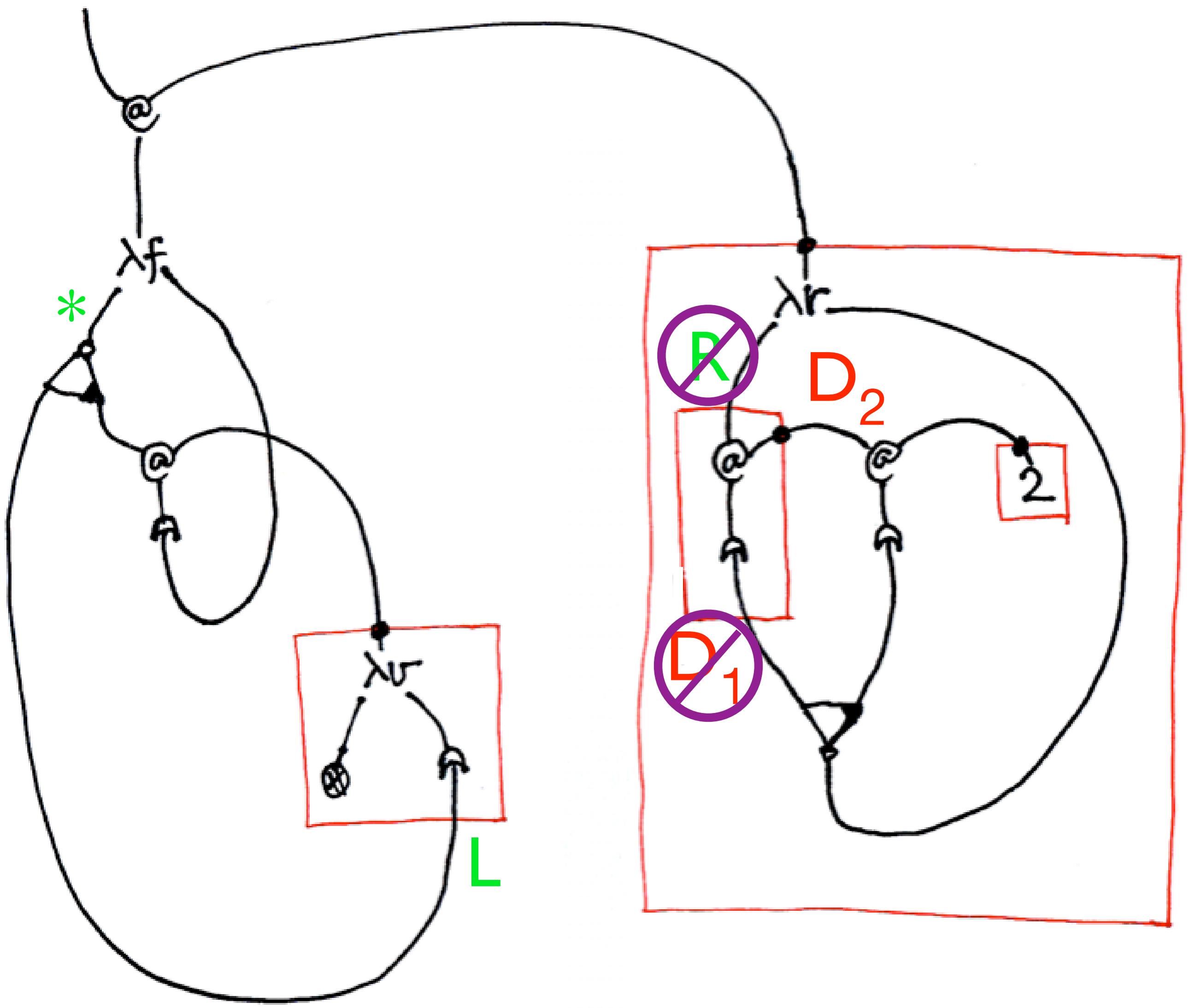
call/cc λr. r(r 2)



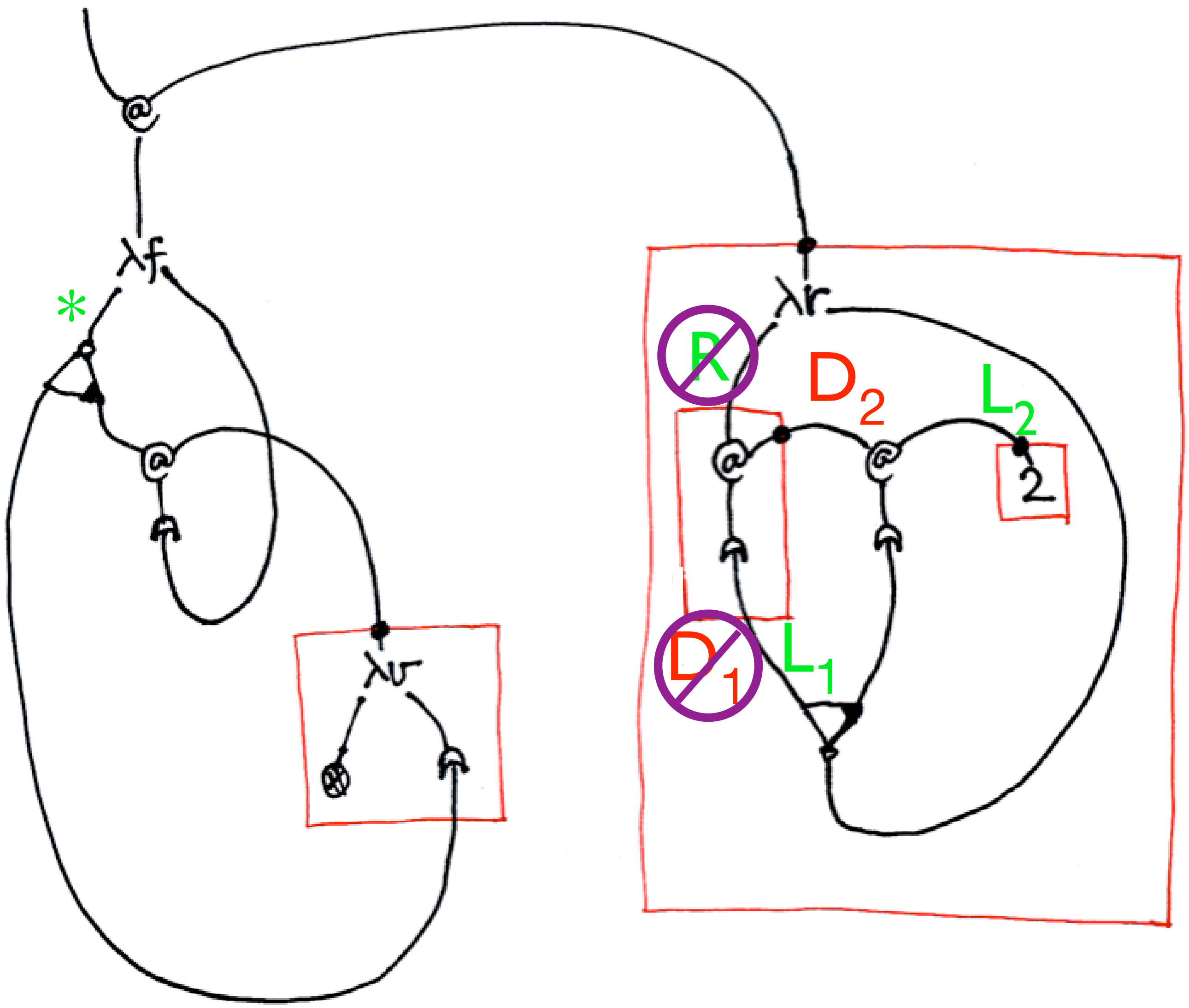
call/cc λr. r(r2)



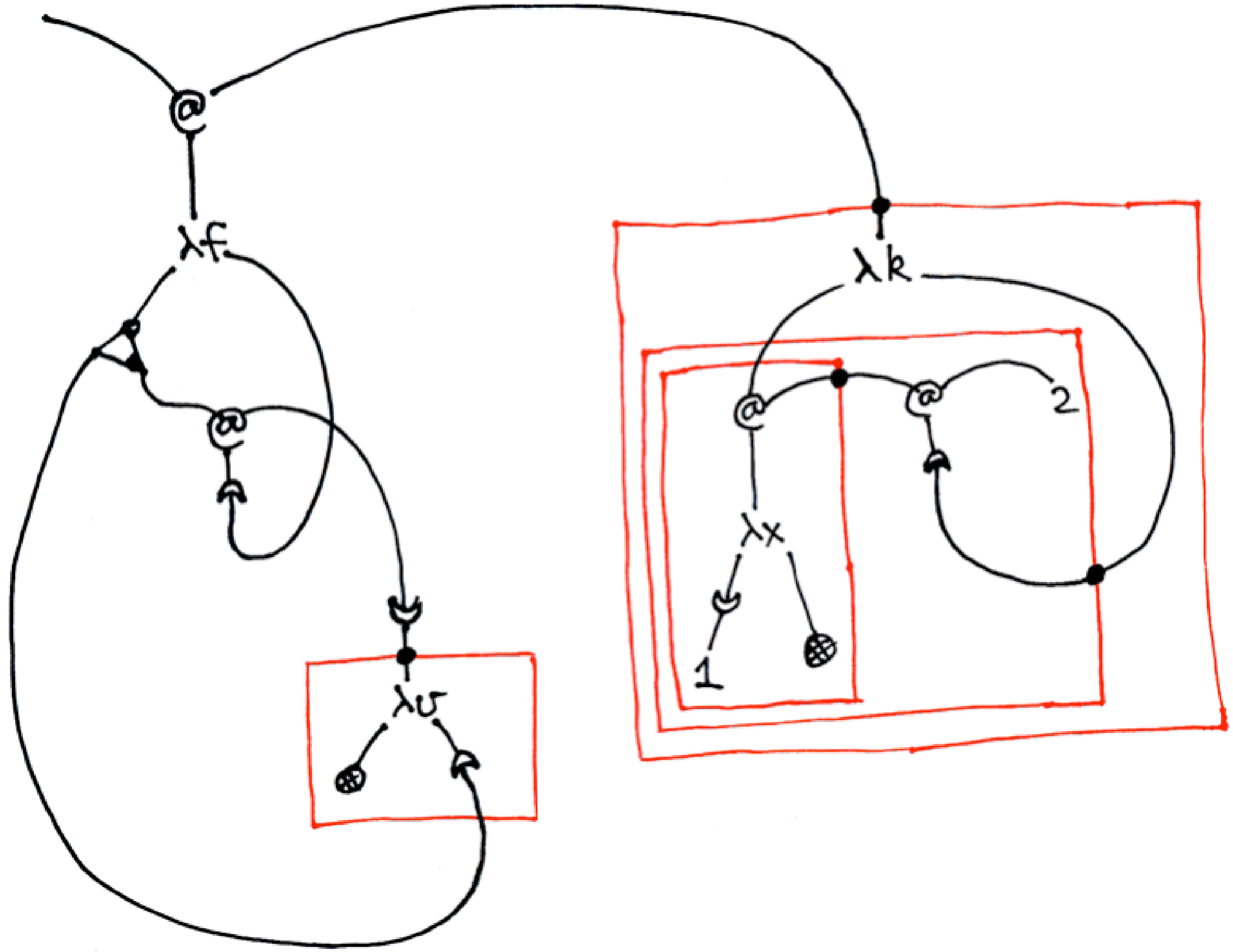
call/cc $\lambda r. r(r\,2)$



call/cc $\lambda r. r(r\,2)$

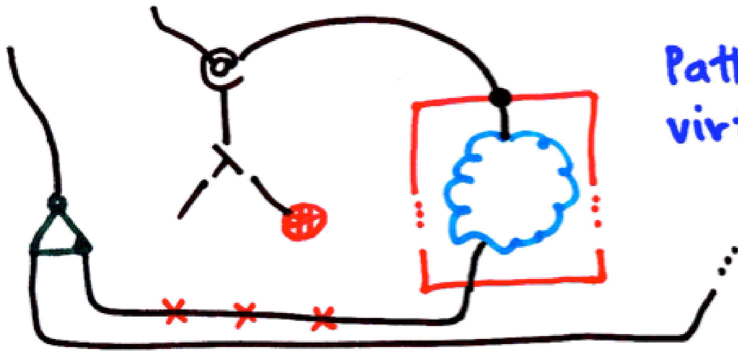


CBV CPS / CBN boxing / call/cc $\lambda k.(\lambda x.1)(k\,2)$



Why does this work? Some naive observations...

- "It's just CPS"... at some point in evaluation of a CPS λ-term, a continuation is thrown away...



Paths are a kind of virtual reduction...

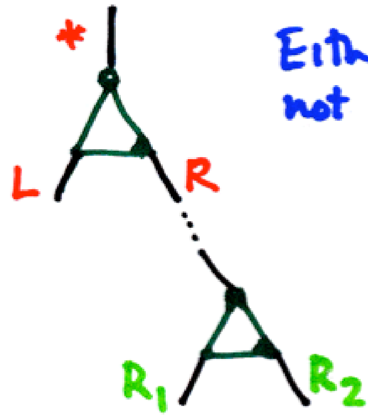
GoI for λ-terms (in CBV, CBN codings): paths enter principal port/root of box first...

- Why do plugs attach to principal ports of boxes, and not

Because under CPS, arguments are invariably boxed...

... λk. ...

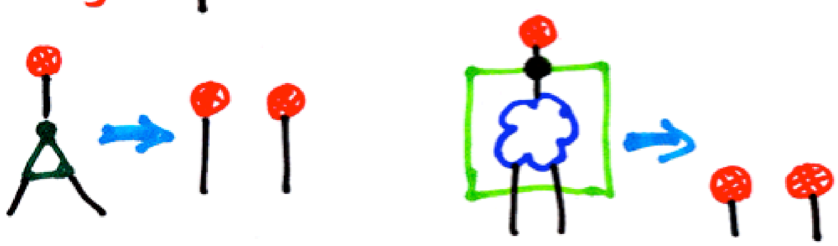
Under CBx, there's only one answer...



Either L or R will succeed, but not both...

... still, only one path to success...

- When arriving at an auxiliary part of a box, synchronize with sibling paths appropriately
- Plugs  act as discriminators...



similarly, to propagate paths

Fin

Filinski's symmetric lambda calculus

Structural rules

$$\frac{S \;\blacktriangleright\; \Gamma, x : A \vdash \Delta}{x.S \;\blacktriangleright\; \Gamma, \underline{A} \vdash \Delta}\,(\textsc{Sel-L}) \qquad \frac{S \;\blacktriangleright\; \Delta \vdash \alpha : A, \Gamma}{S.\alpha \;\blacktriangleright\; \Delta \vdash \underline{A}, \Gamma}\,(\textsc{Sel-R})$$

$$\frac{P \;\blacktriangleright\; \Gamma \vdash \Delta}{P \;\blacktriangleright\; \Gamma, x : A \vdash \Delta}\,(\textsc{W-L}) \qquad \frac{P \;\blacktriangleright\; \Delta \vdash \Gamma}{P \;\blacktriangleright\; \Delta \vdash \alpha : A, \Gamma}\,(\textsc{W-R})$$

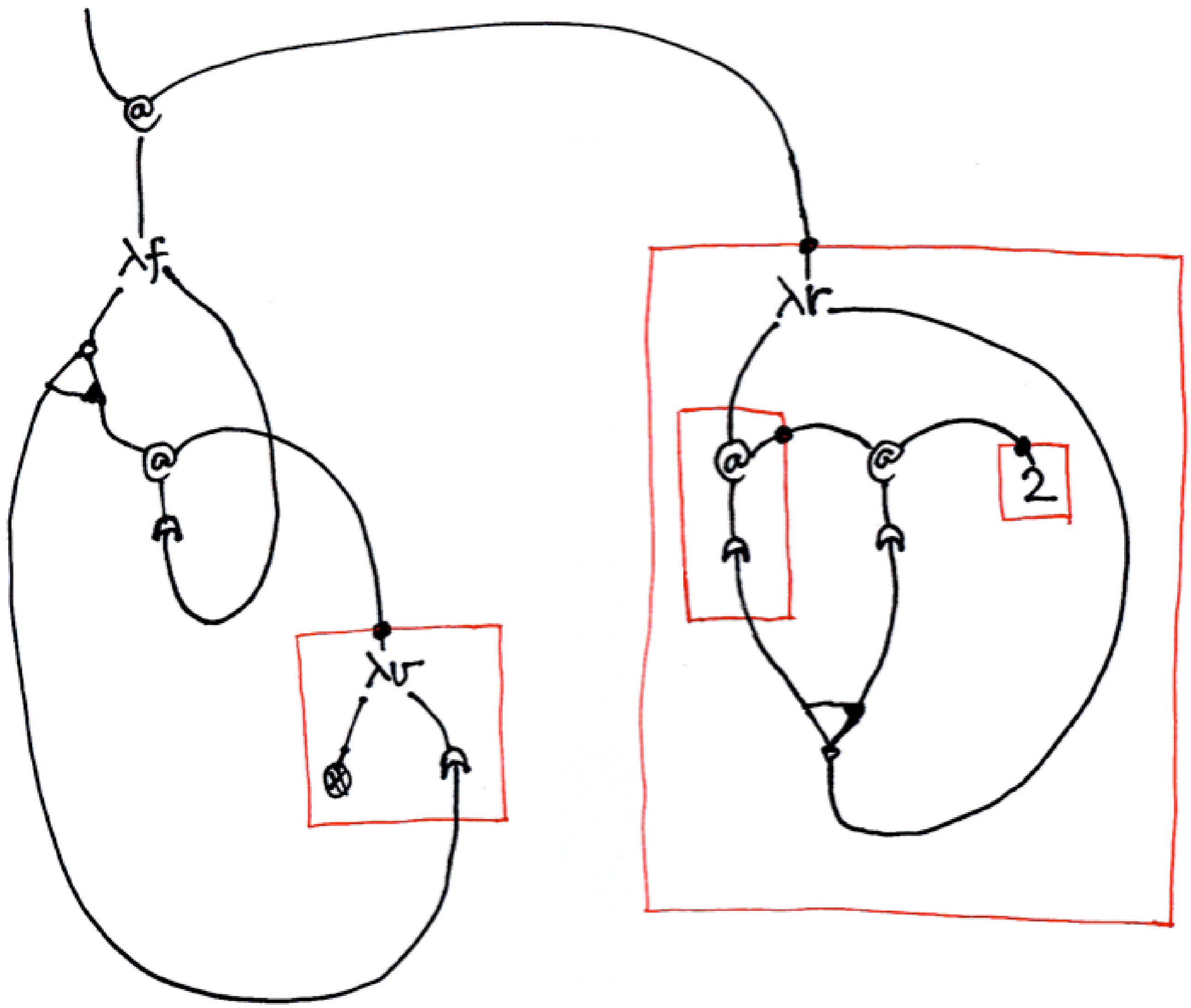
$$\frac{P \;\blacktriangleright\; \Gamma, x : A, y : A \vdash \Delta}{[x/y]P \;\blacktriangleright\; \Gamma, x : A \vdash \Delta}\,(\textsc{C-L}) \qquad \frac{P \;\blacktriangleright\; \Delta \vdash \beta : A, \alpha : A, \Gamma}{[\alpha/\beta]P \;\blacktriangleright\; \Delta \vdash \alpha : A, \Gamma}\,(\textsc{C-R})$$

$$\frac{P \;\blacktriangleright\; \Gamma, x : A, y : B, \Sigma \vdash \Delta}{P \;\blacktriangleright\; \Gamma, y : B, x : A, \Sigma \vdash \Delta}\,(\textsc{Int-L}) \qquad \frac{P \;\blacktriangleright\; \Delta \vdash \Sigma, \beta : B, \alpha : A, \Gamma}{P \;\blacktriangleright\; \Delta \vdash \Sigma, \alpha : A, \beta : B, \Gamma}\,(\textsc{Int-R})$$

Proofnets and Paths in Constructive Classical Logic: Too Old, Tout Nu

Harry Mairson
David van Horn

call/cc λr. r(r2)



Never, never do this (in large quantities)

$$
\begin{array}{rcl}
V, W & ::= & x \mid \langle V, W \rangle \mid \langle V \rangle \mathsf{inl} \mid \langle V \rangle \mathsf{inr} \mid [K]\mathsf{not} \mid \lambda x.N \mid K @^{\circ} V \\
P, Q & ::= & \alpha \mid [P, Q] \mid \mathsf{fst}[P] \mid \mathsf{snd}[P] \mid \mathsf{not}\langle M \rangle \mid M @ Q \mid \lambda^{\circ}\alpha.K
\end{array}
$$

```
(define (eval exp env cont)
  (cond ((variable? exp) ...)
        ((lambda? exp) ...)
        ((application? exp) ...)
```