Dynamic GoI machine

Call-by-need and call-by-value graph rewriter

Koko Muroya & Dan Ghica
(Univ. Birmingham)

S-REPLS 4 (London), 27 September 2016

GoI machine [Danos & Regnier ’99] [Mackie ’95]

- abstract machine to interpret λ-calculus
- a λ-term → a graph → moves of a token

compositionally

- Geometry of Interaction [Girard, ’89]
- “an execution path on a proof net”

GoI machine [Danos & Regnier '99] [Mackie '95]

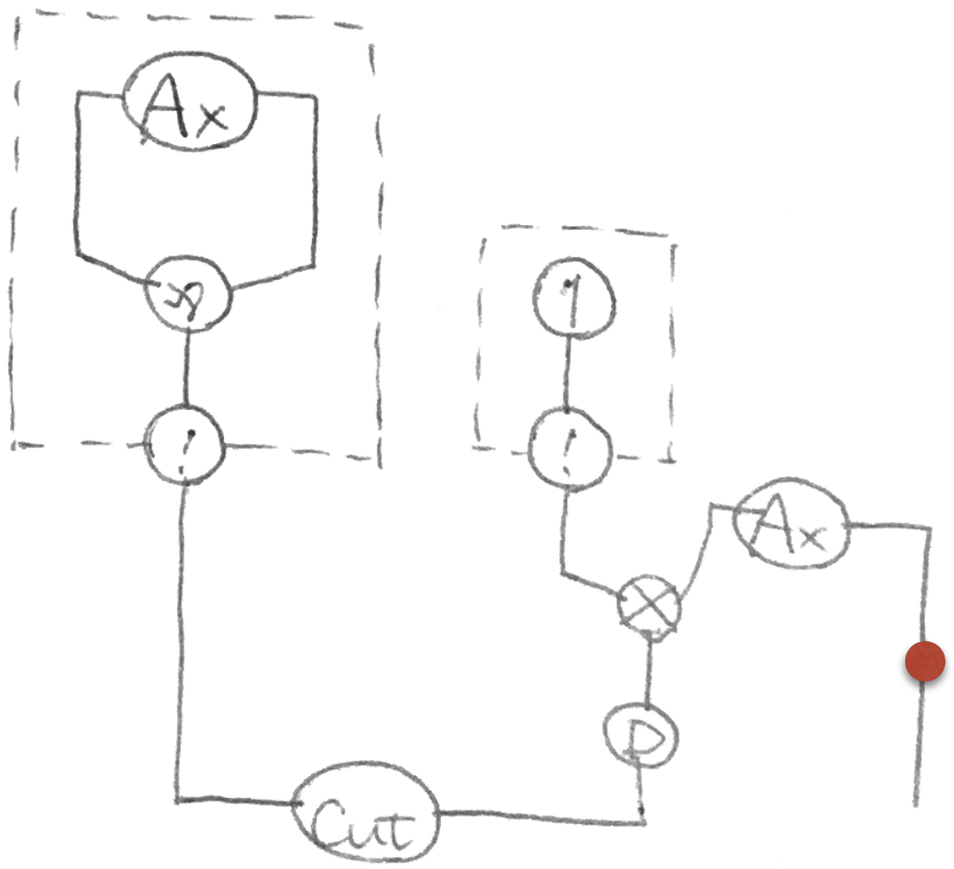
$$(\lambda x\,.\,x)\,1$$

q

q

1

1



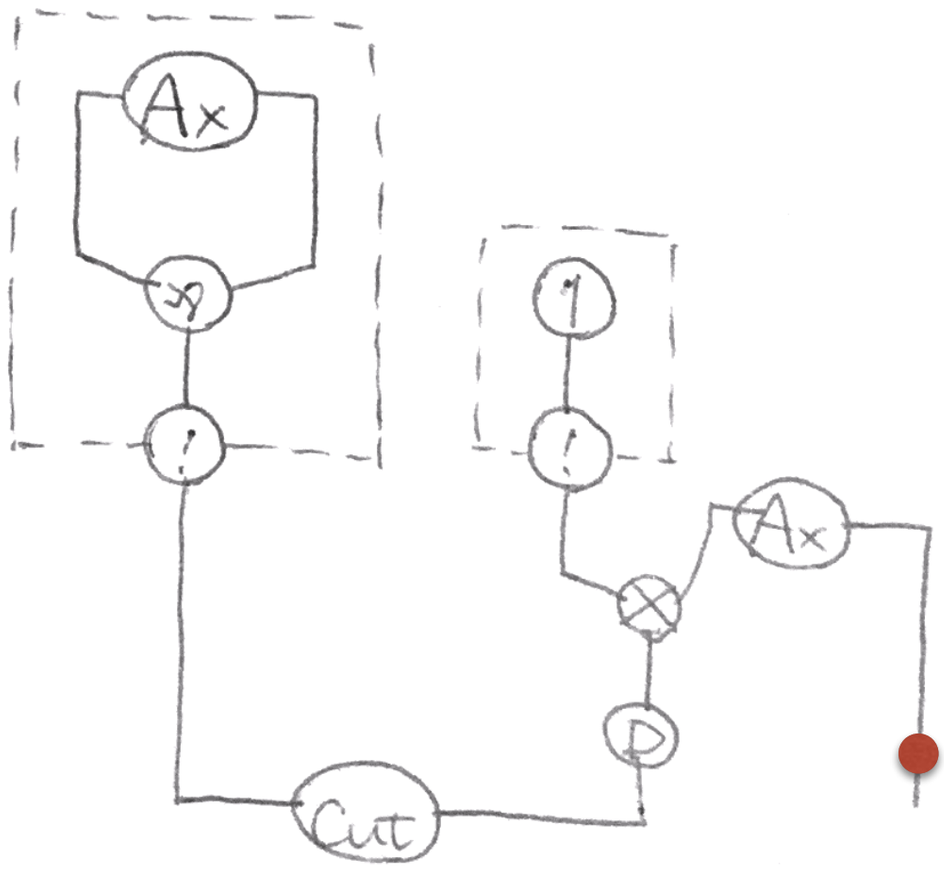
GoI machine [Danos & Regnier '99] [Mackie '95]

$$(\lambda x\,.\,x)\,1$$

q

q
1

1



GoI machine [Danos & Regnier '99] [Mackie '95]

call-by-name

- a token on a *static* graph
- space efficiency
- compositionality / interaction between components
  - categorical GoI [Abramsky+, '02]
  - compilation to digital circuits [Ghica+, '07-]

GoI machine [Danos & Regnier '99] [Mackie '95]

~~call-by-name~~ **call-by-value**

CPS transformation
(Schöpp, Hoshino+)

- a token on a *static* graph

memory assigned to nodes
(Hoshino+, Dal Lago+)

- space efficiency

dynamic jumps
(Fernández+)

parallelism: multiple tokens
(Dal Lago+)

Dynamic GoI machine

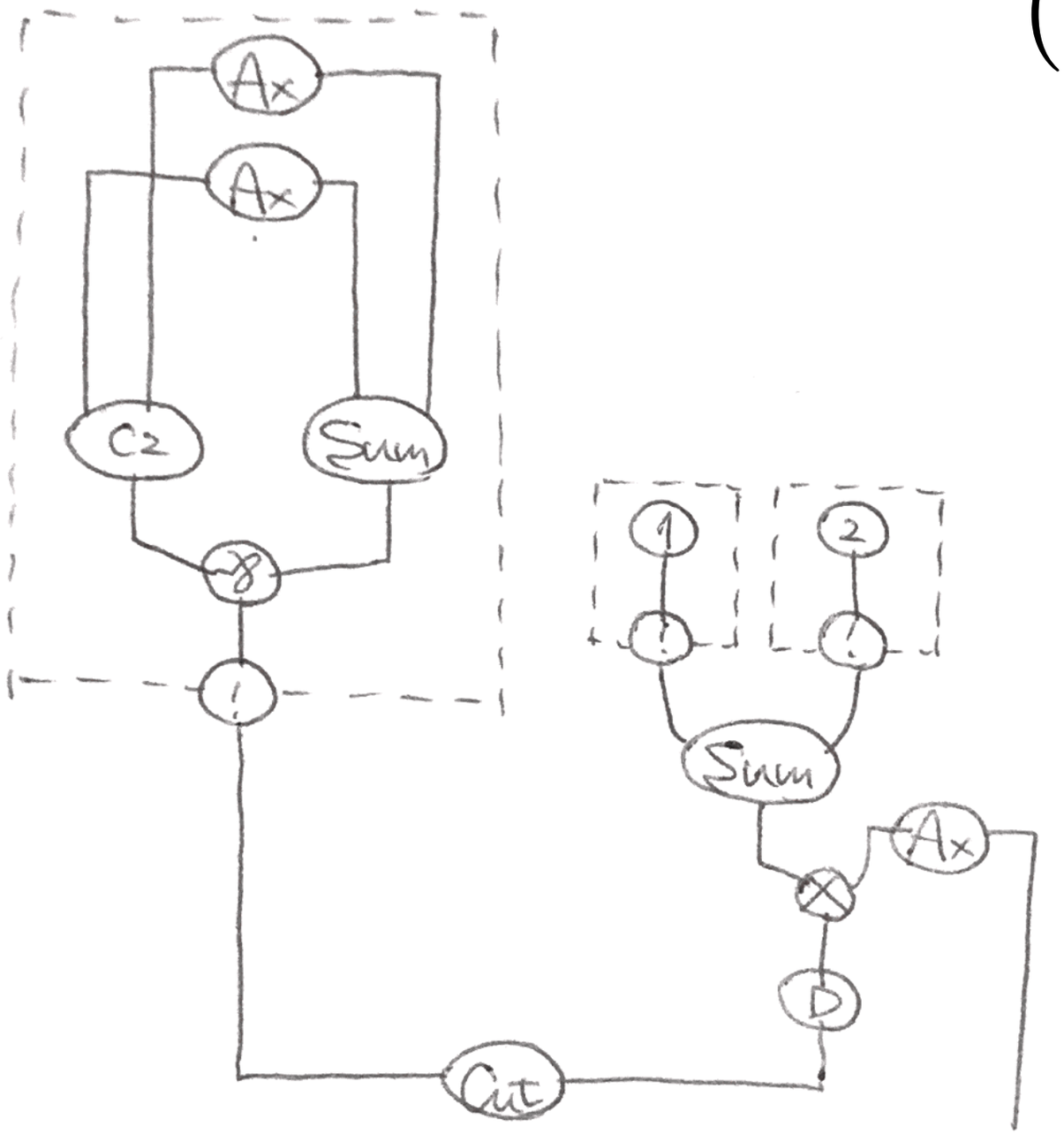
~~call-by-name~~ **call-by-value**

avoid repeated evaluation

force evaluation of function arguments

- a token on a ~~*static*~~ graph
  - dynamic rewrites & “checkpoint” mechanism
- ~~space~~ **time** efficiency

Avoid re-evaluation



$$(\lambda x\,.\,x + x)\,(1 + 2)$$

q

q

3

q

3

6

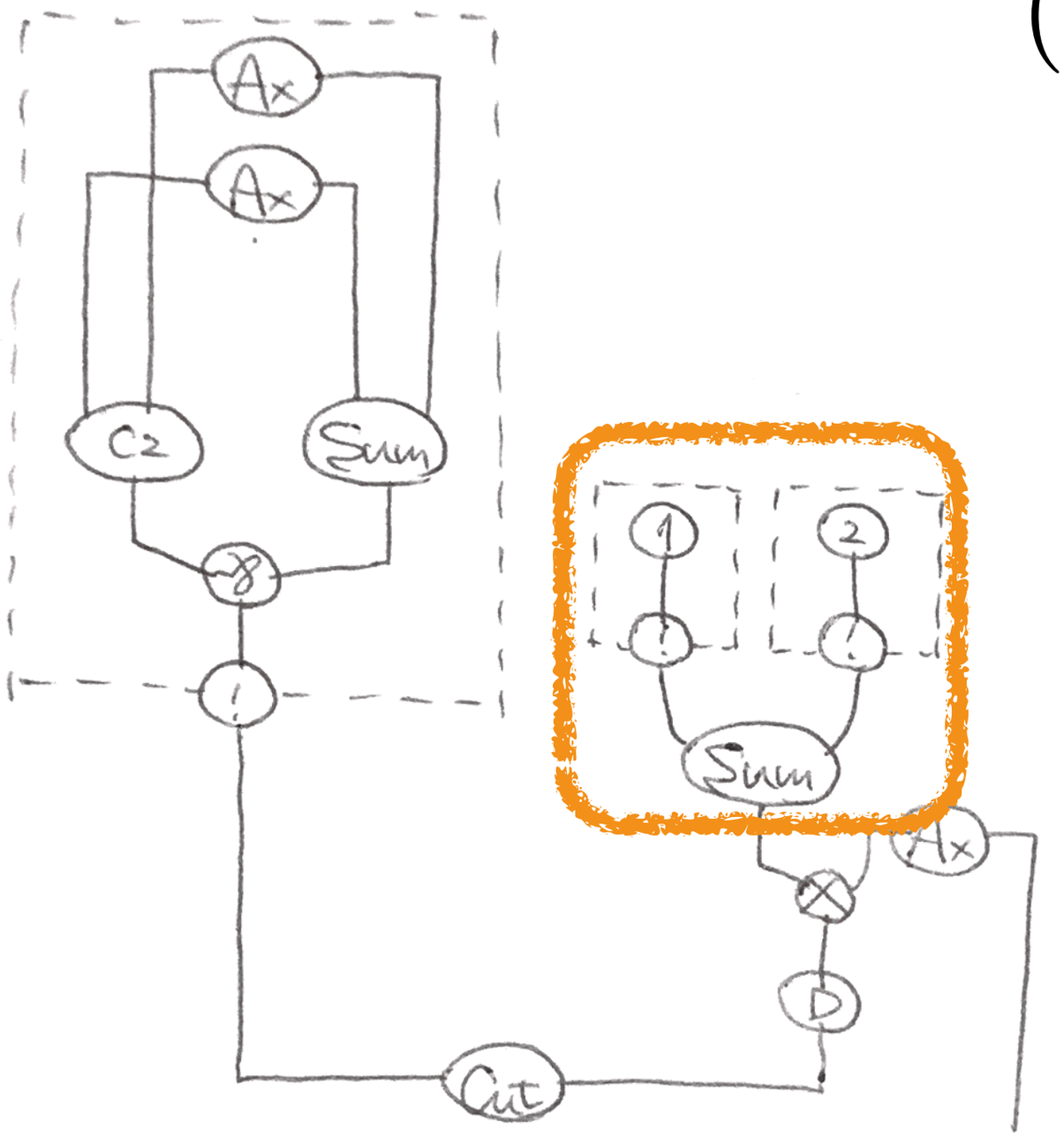
Avoid re-evaluation

$$(\lambda x\,.\,x + x)\,(1 + 2)$$

q

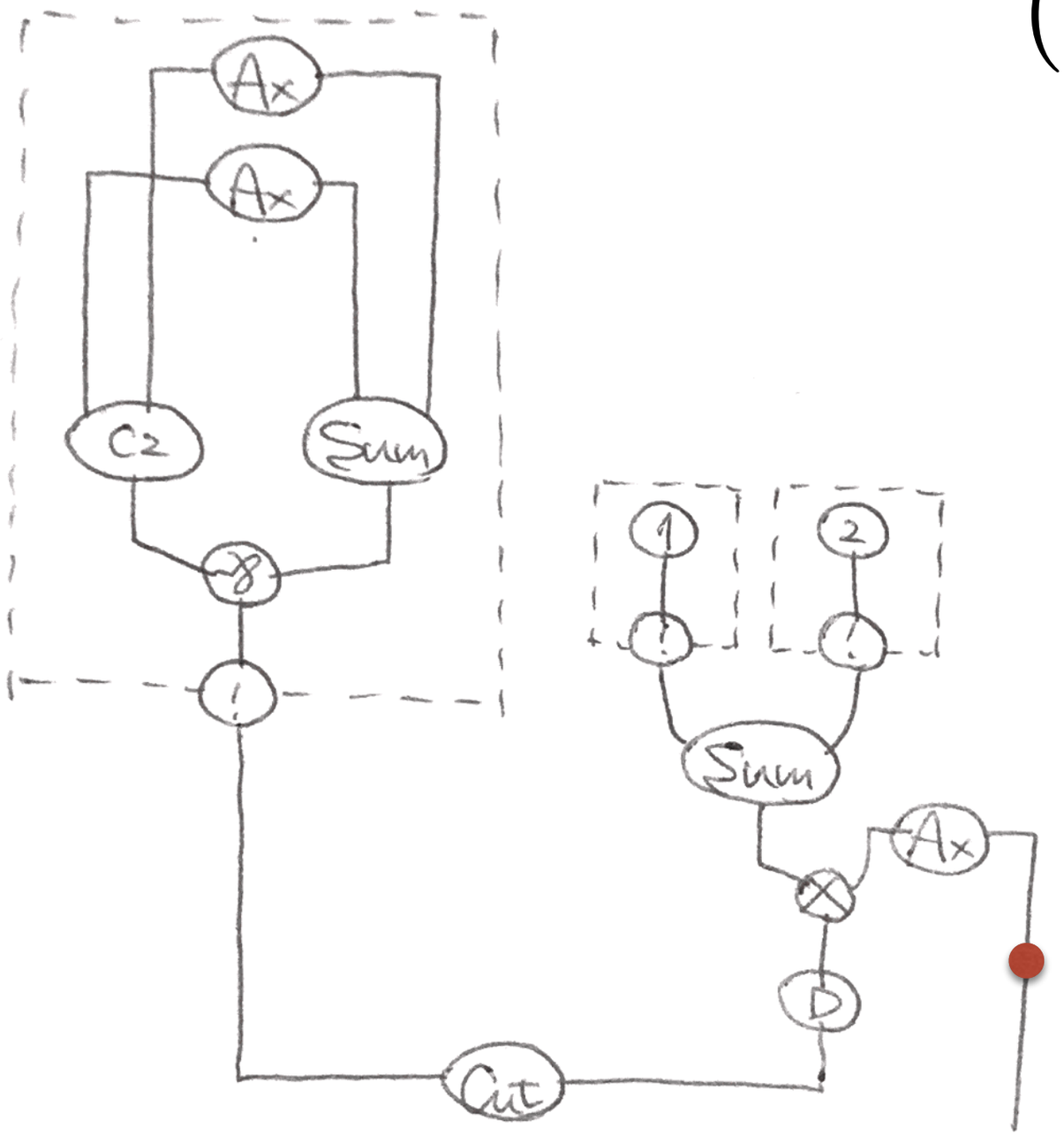
q
3
q
3

6



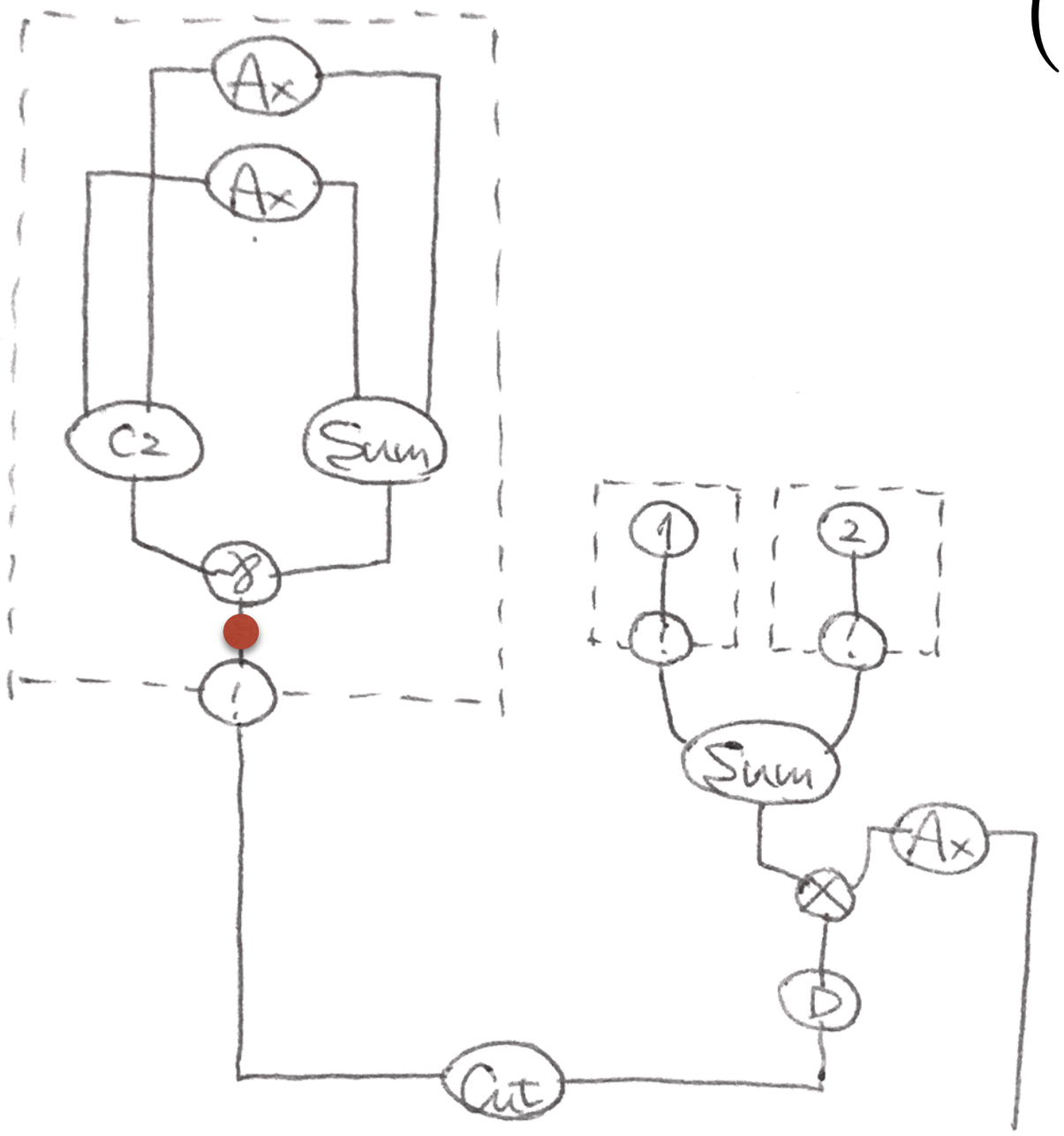
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



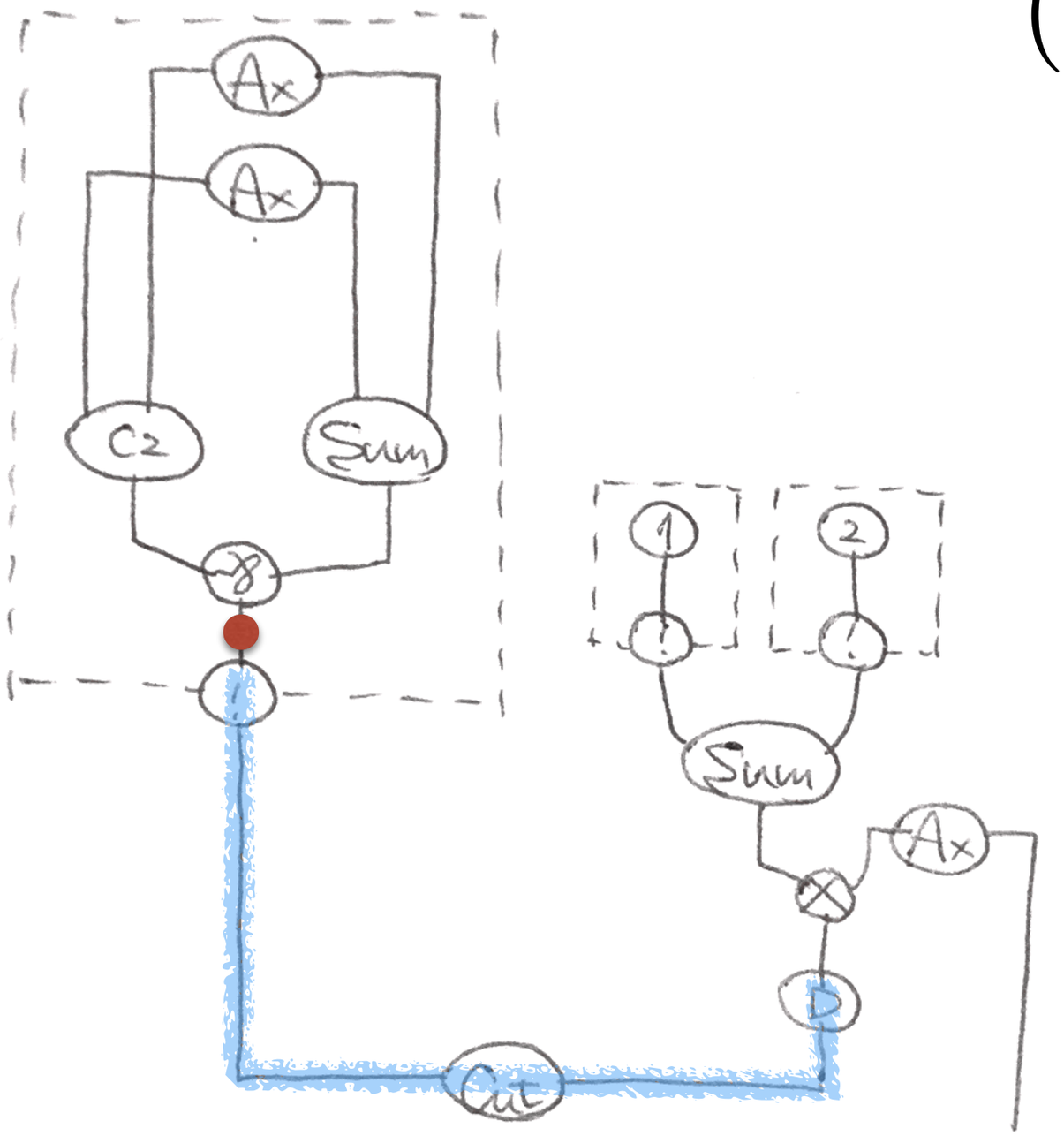
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



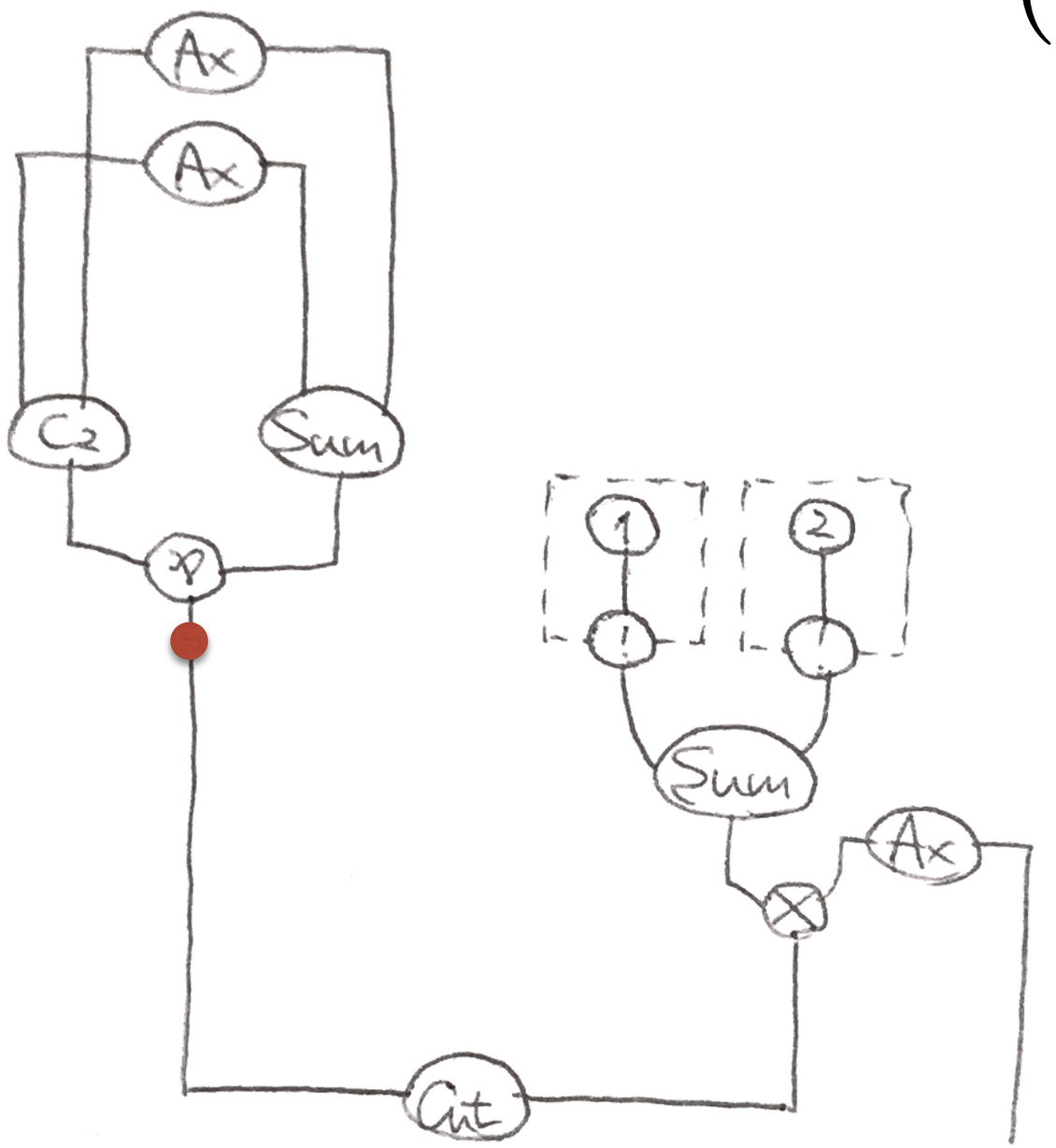
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



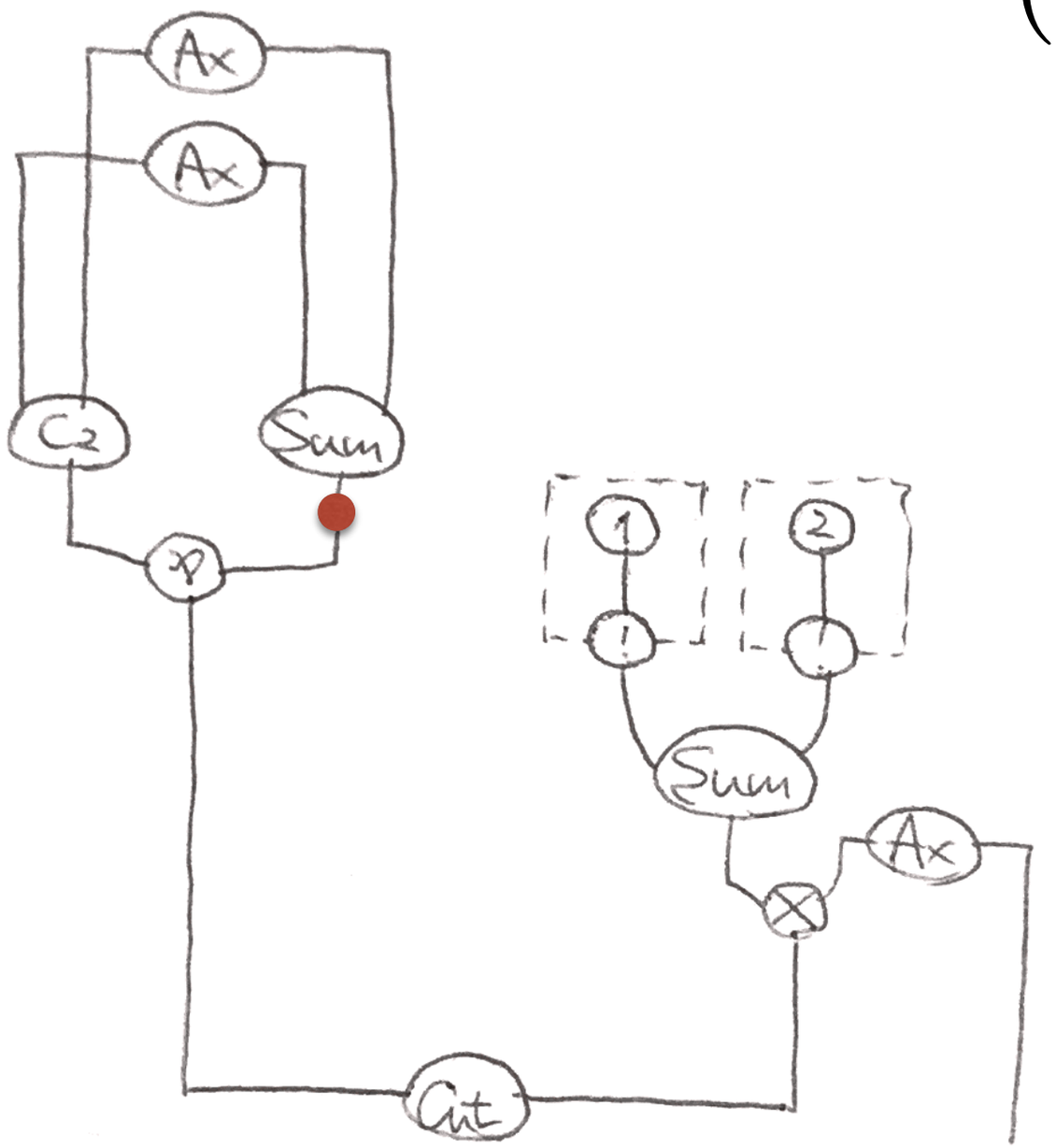
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



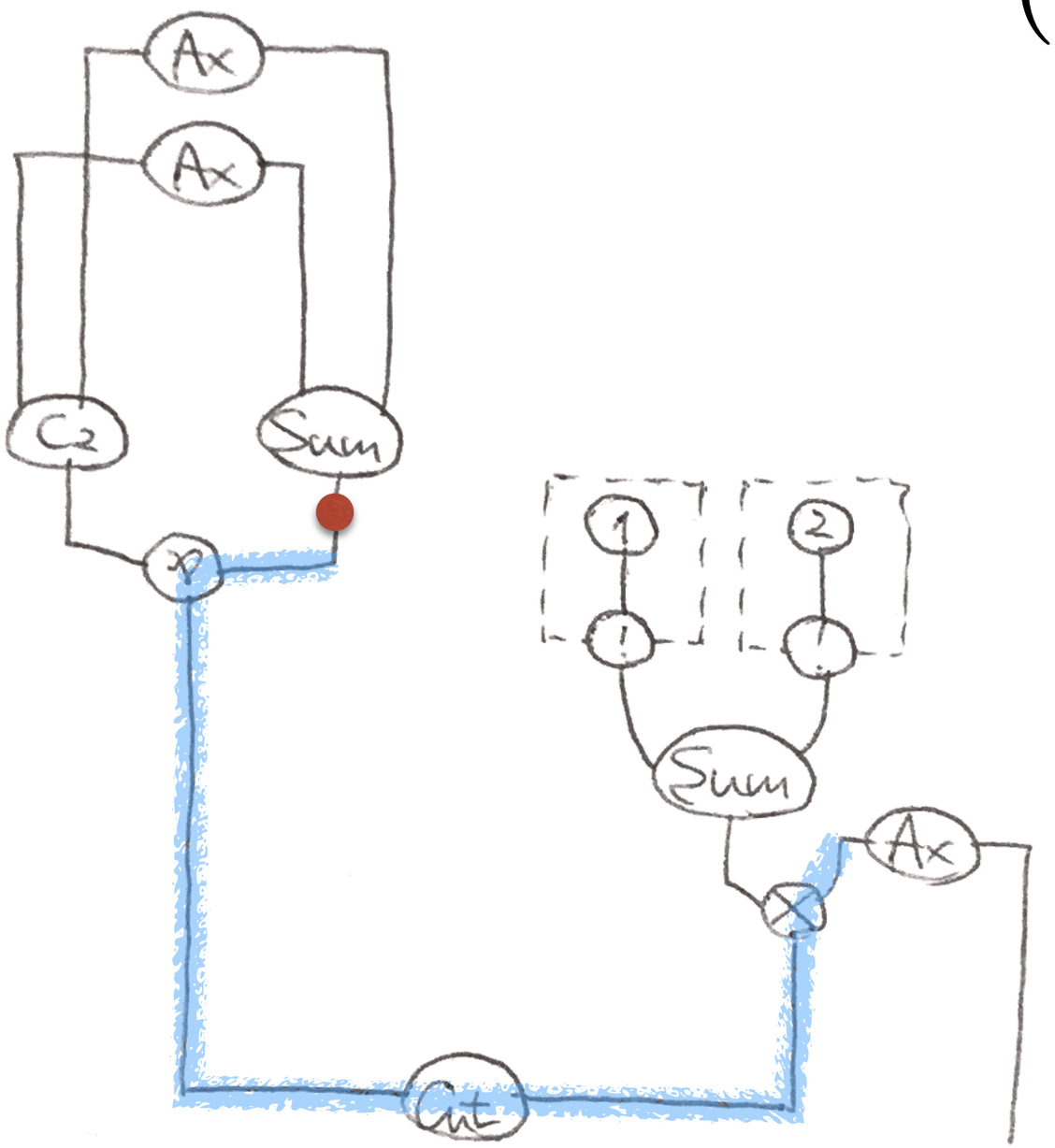
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



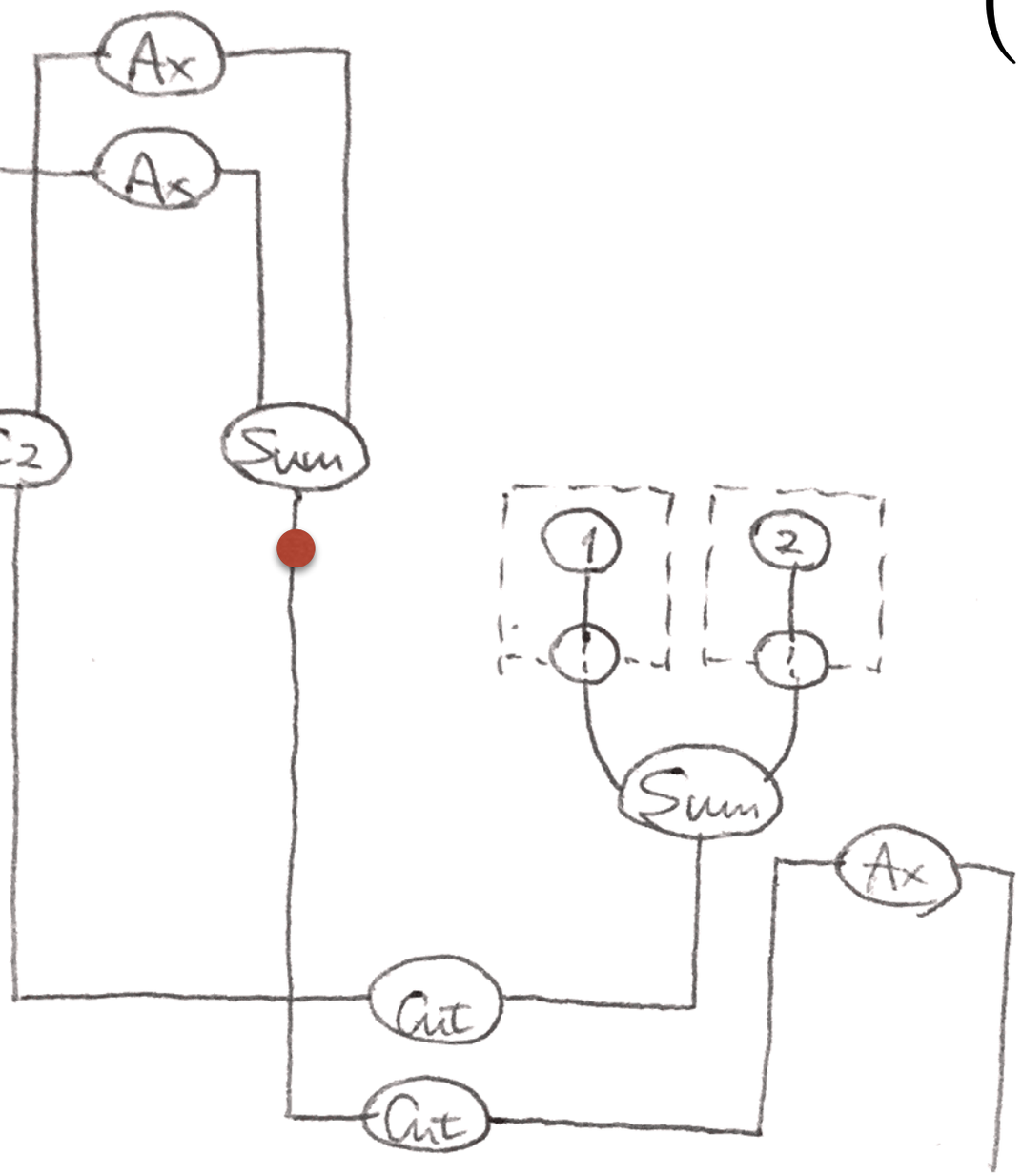
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



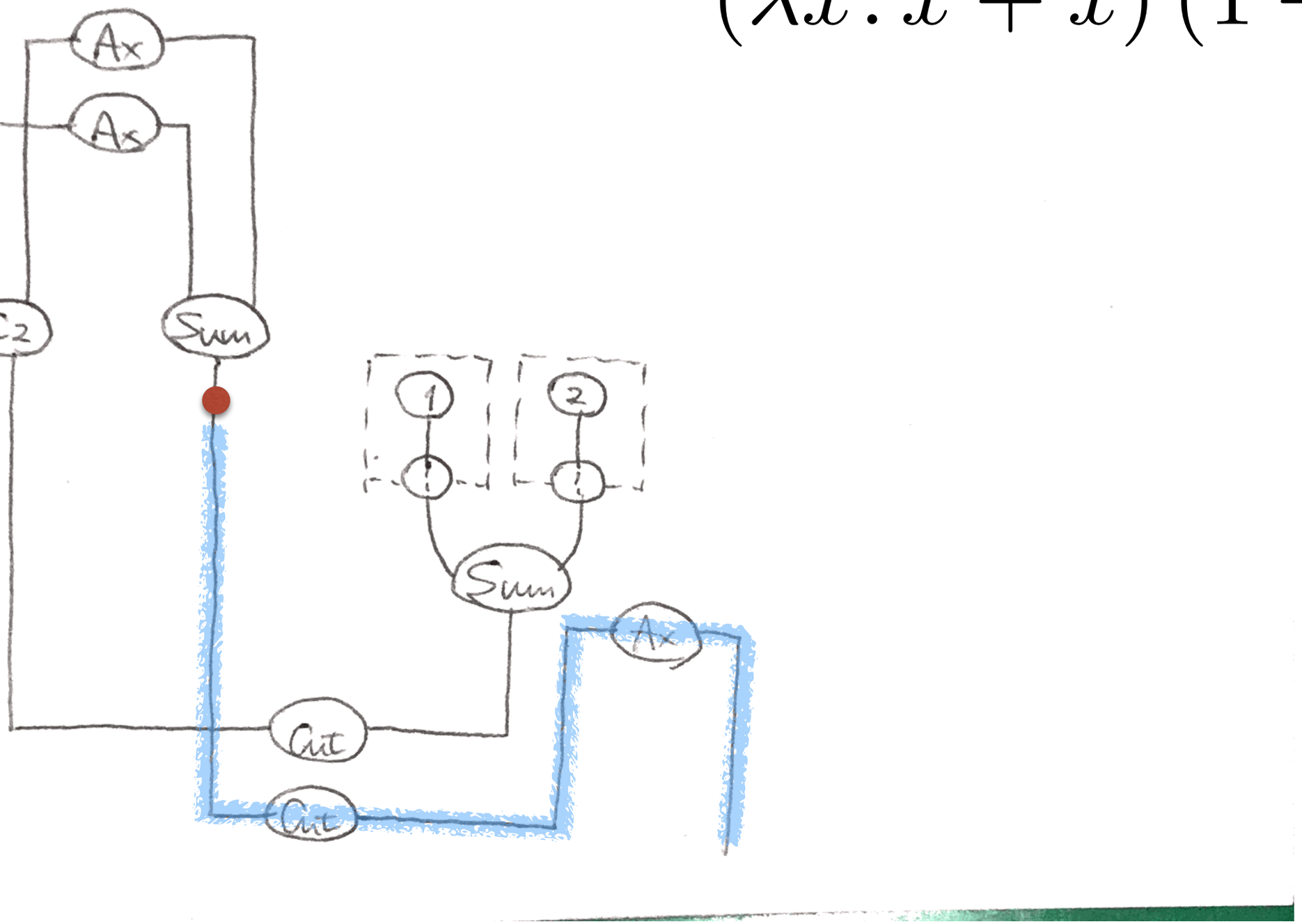
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



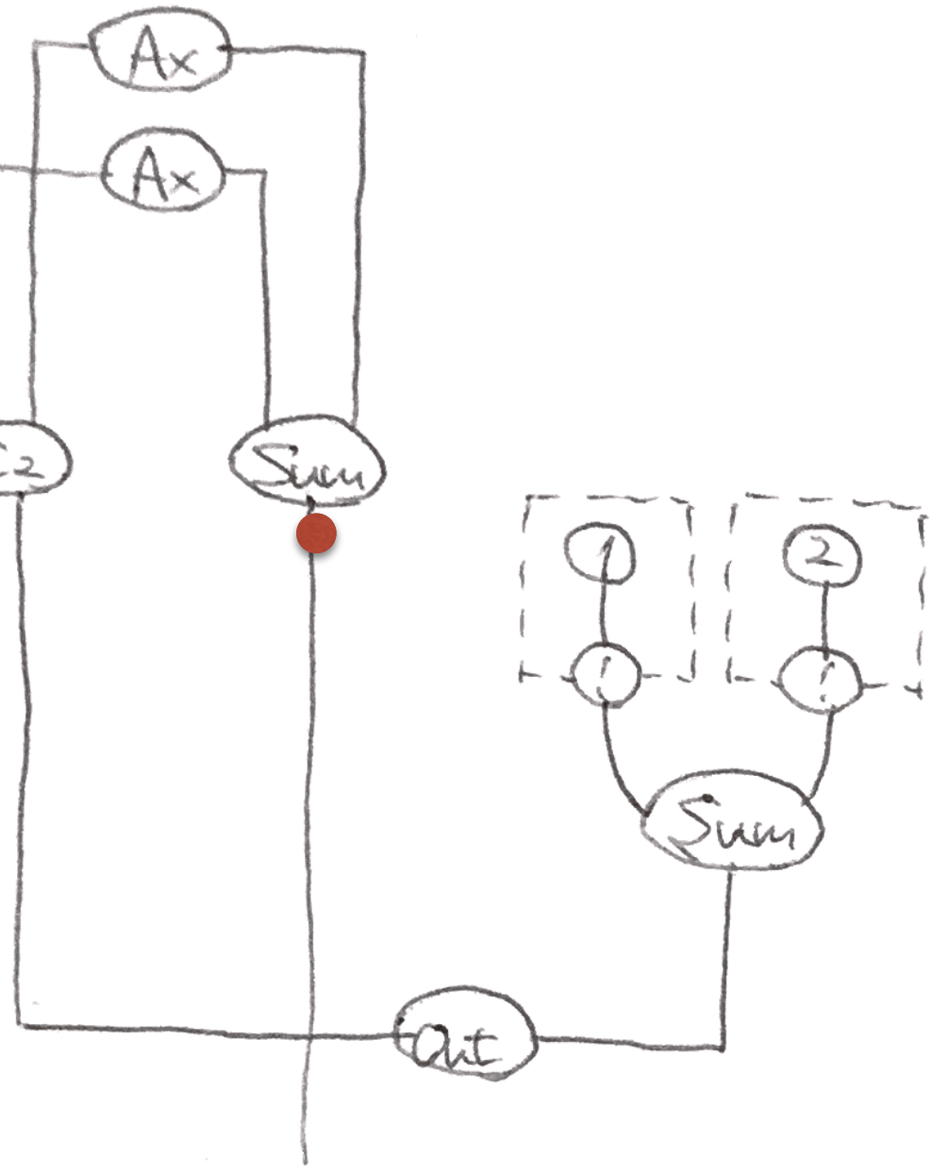
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



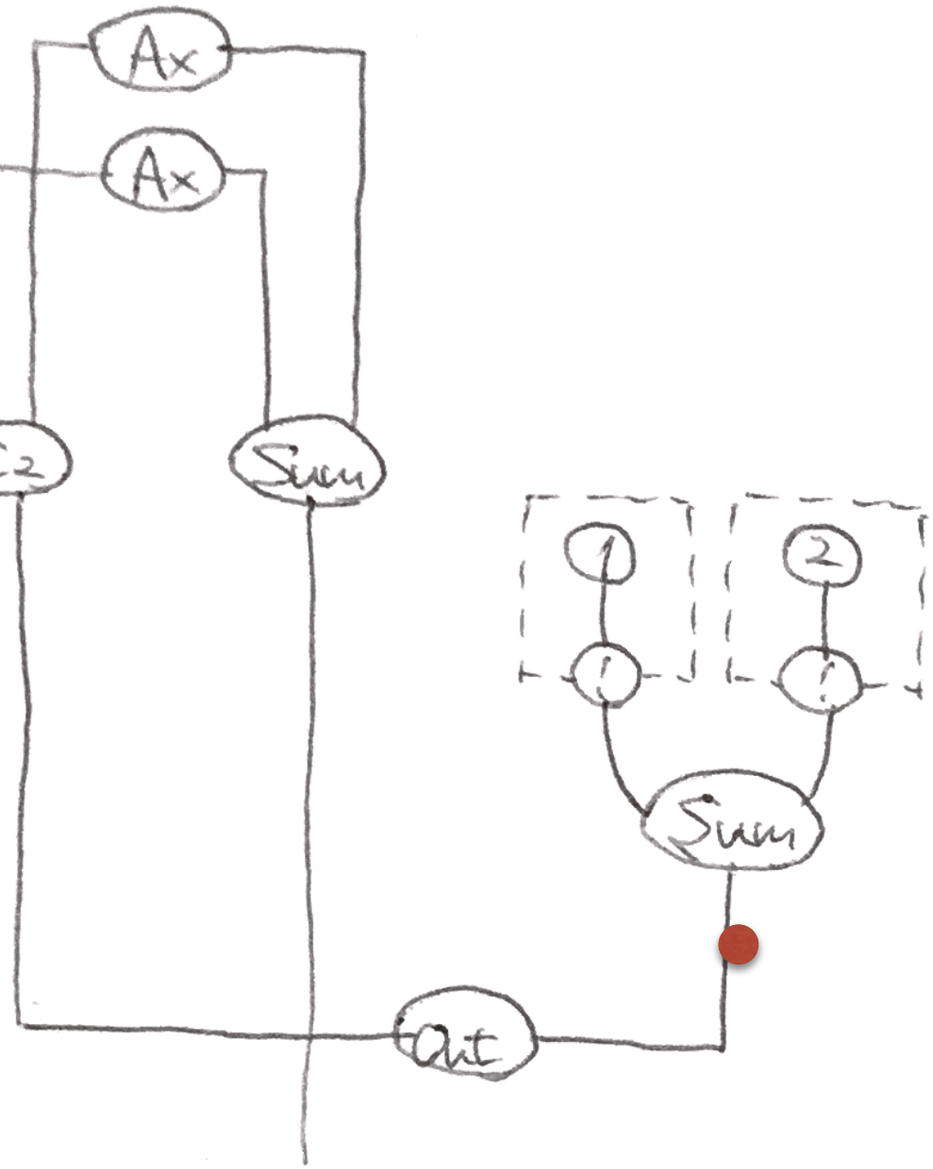
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



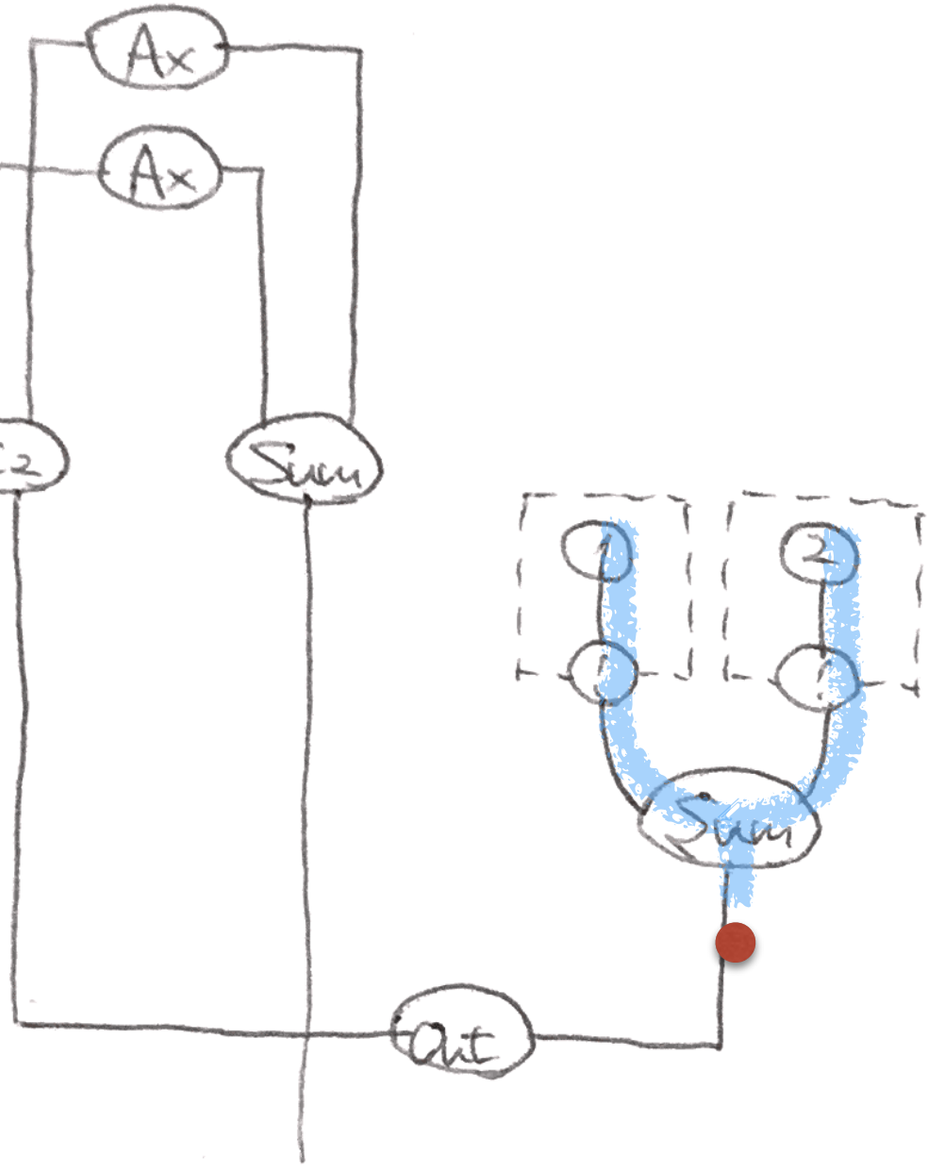
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



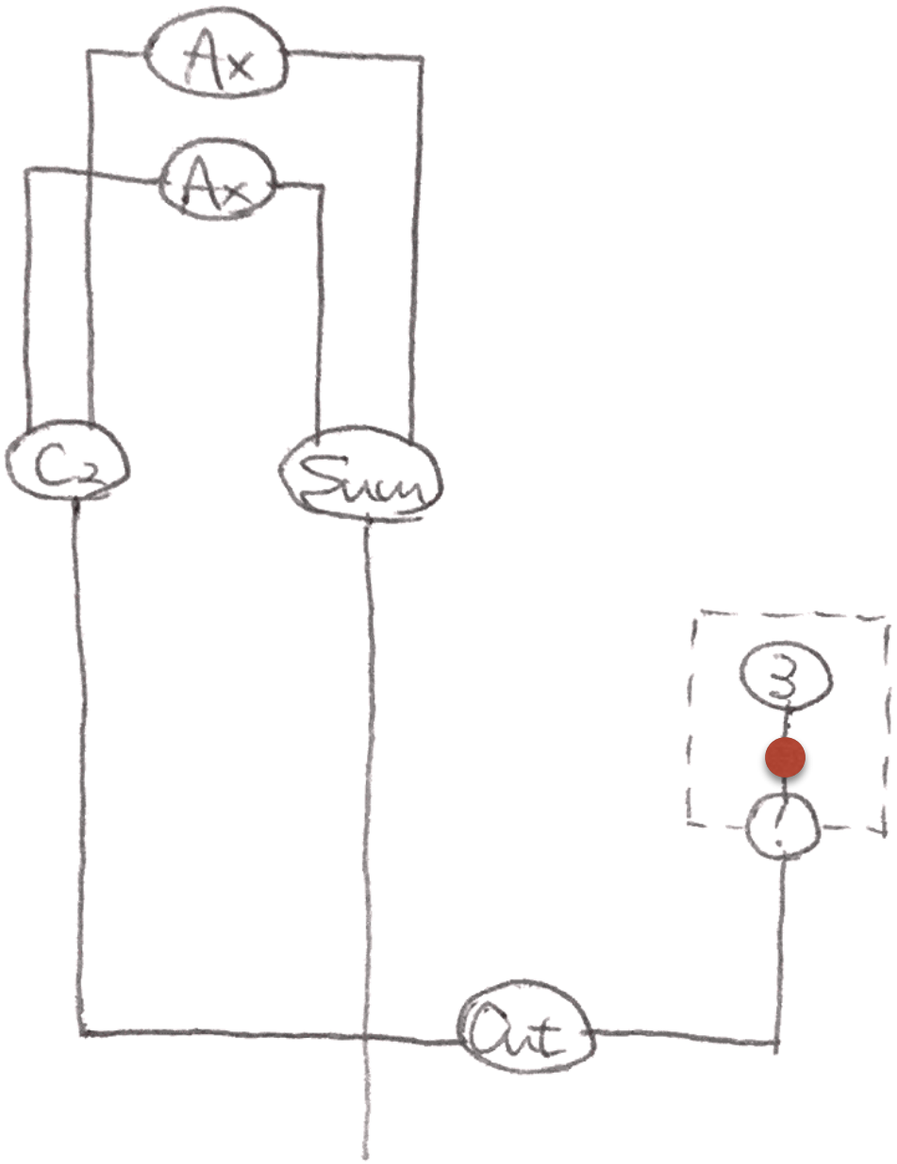
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



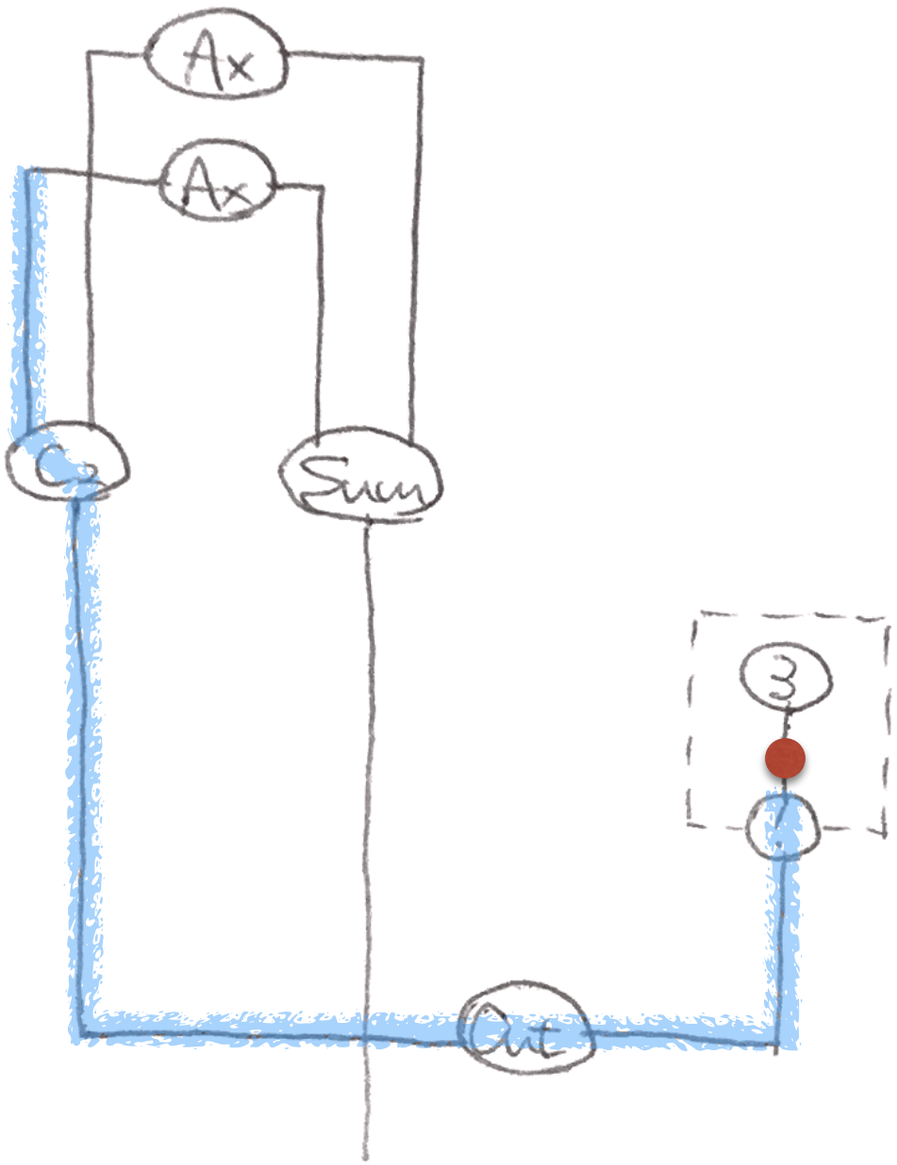
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



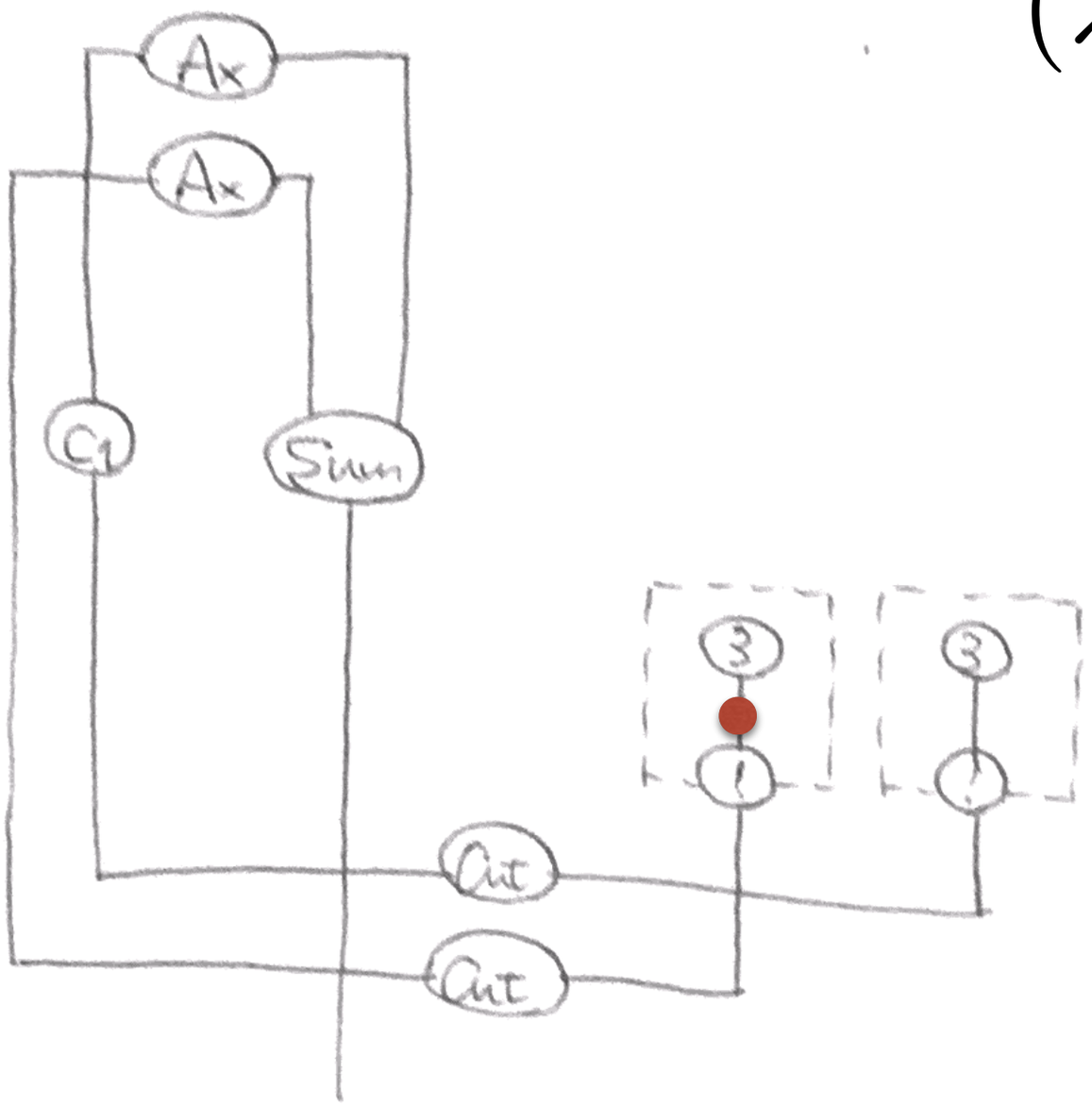
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



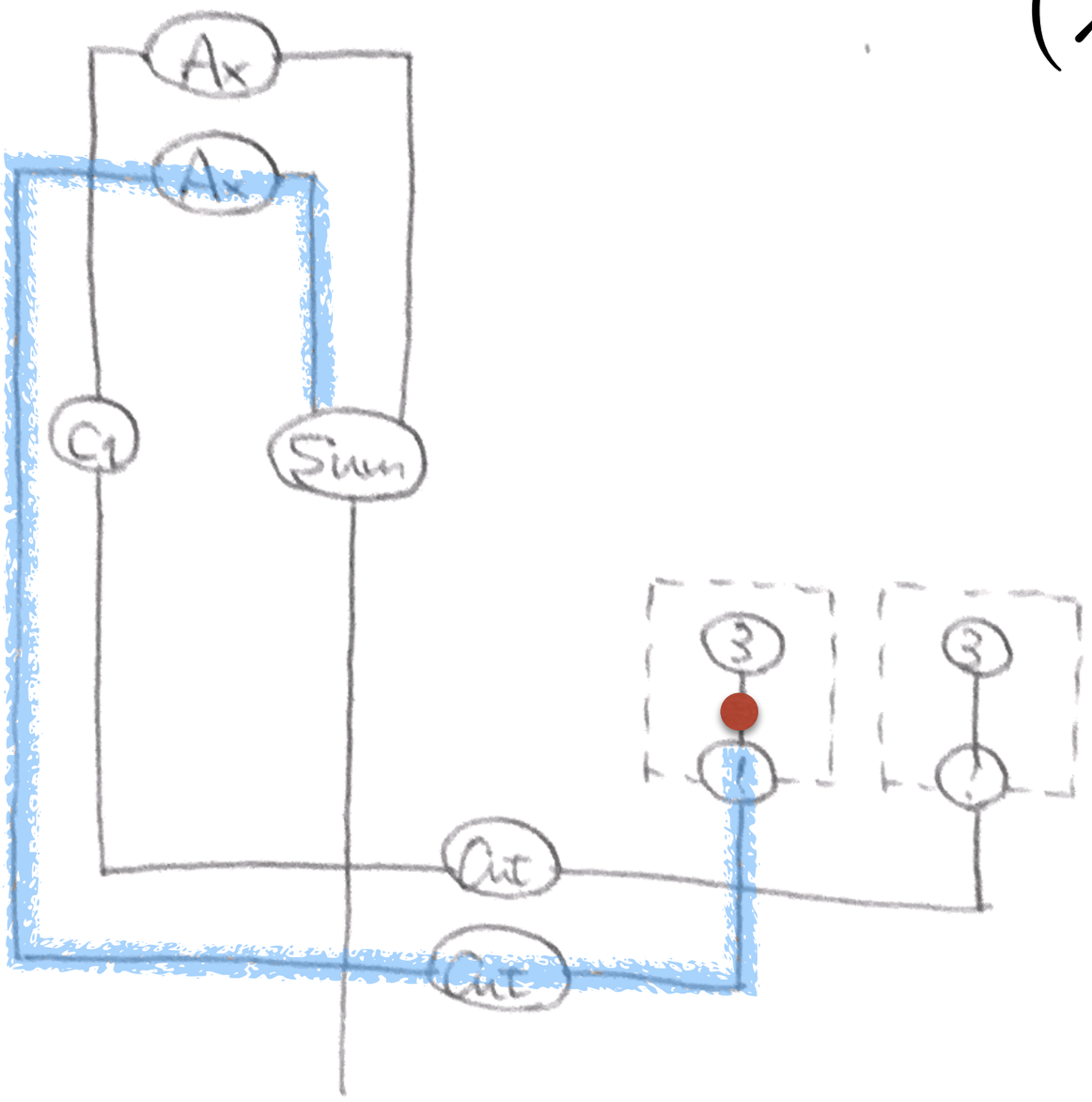
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



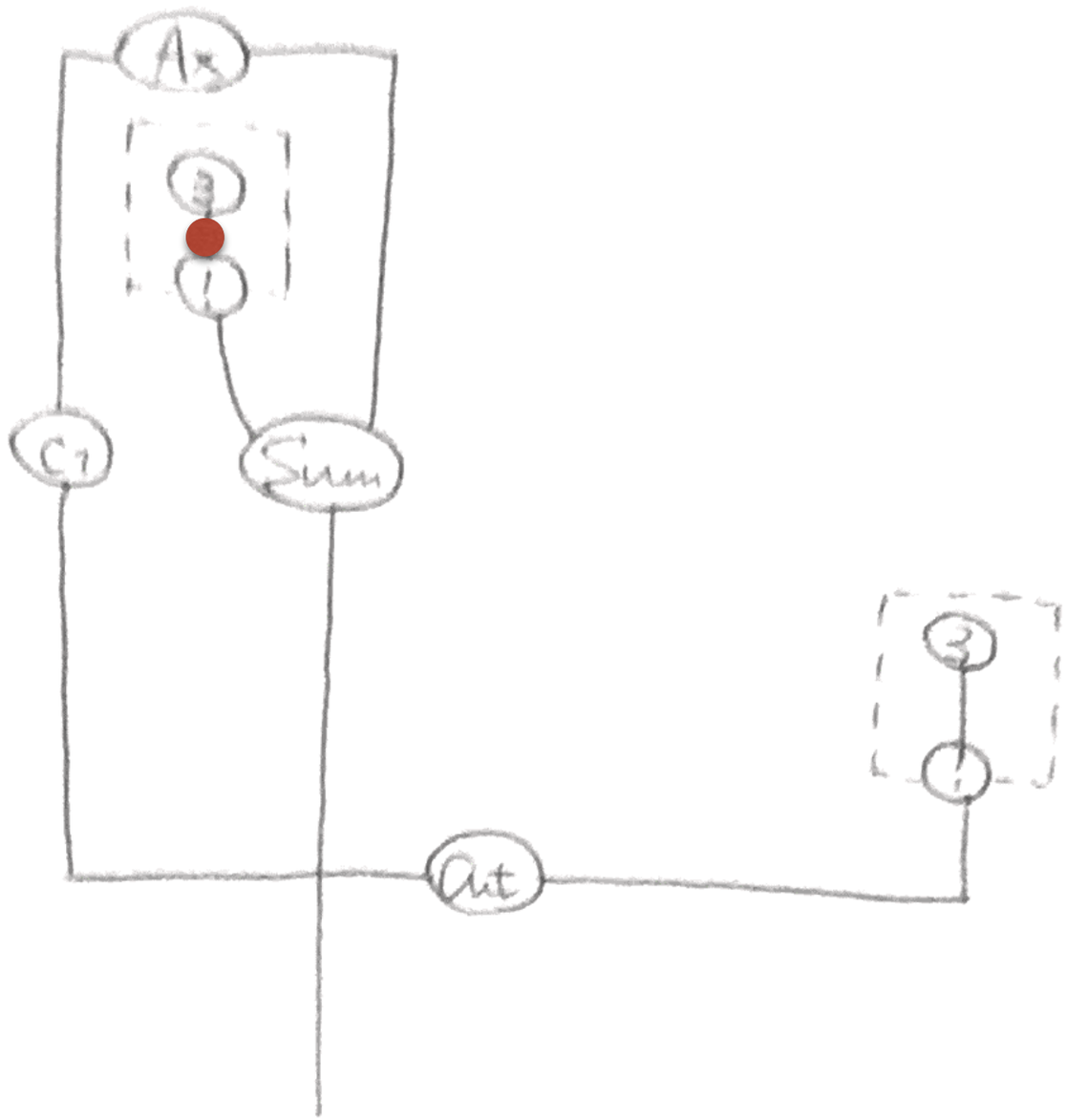
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



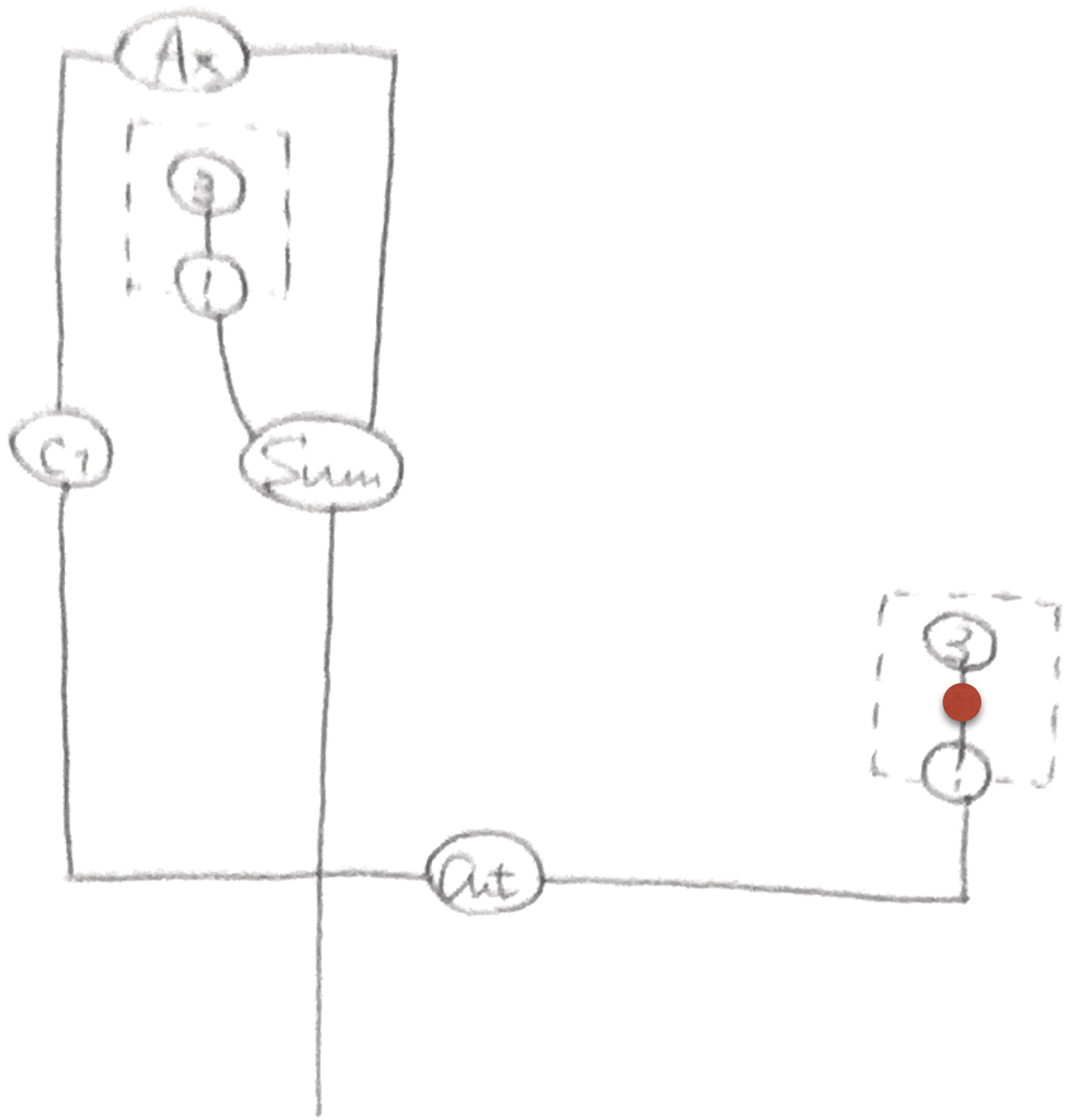
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



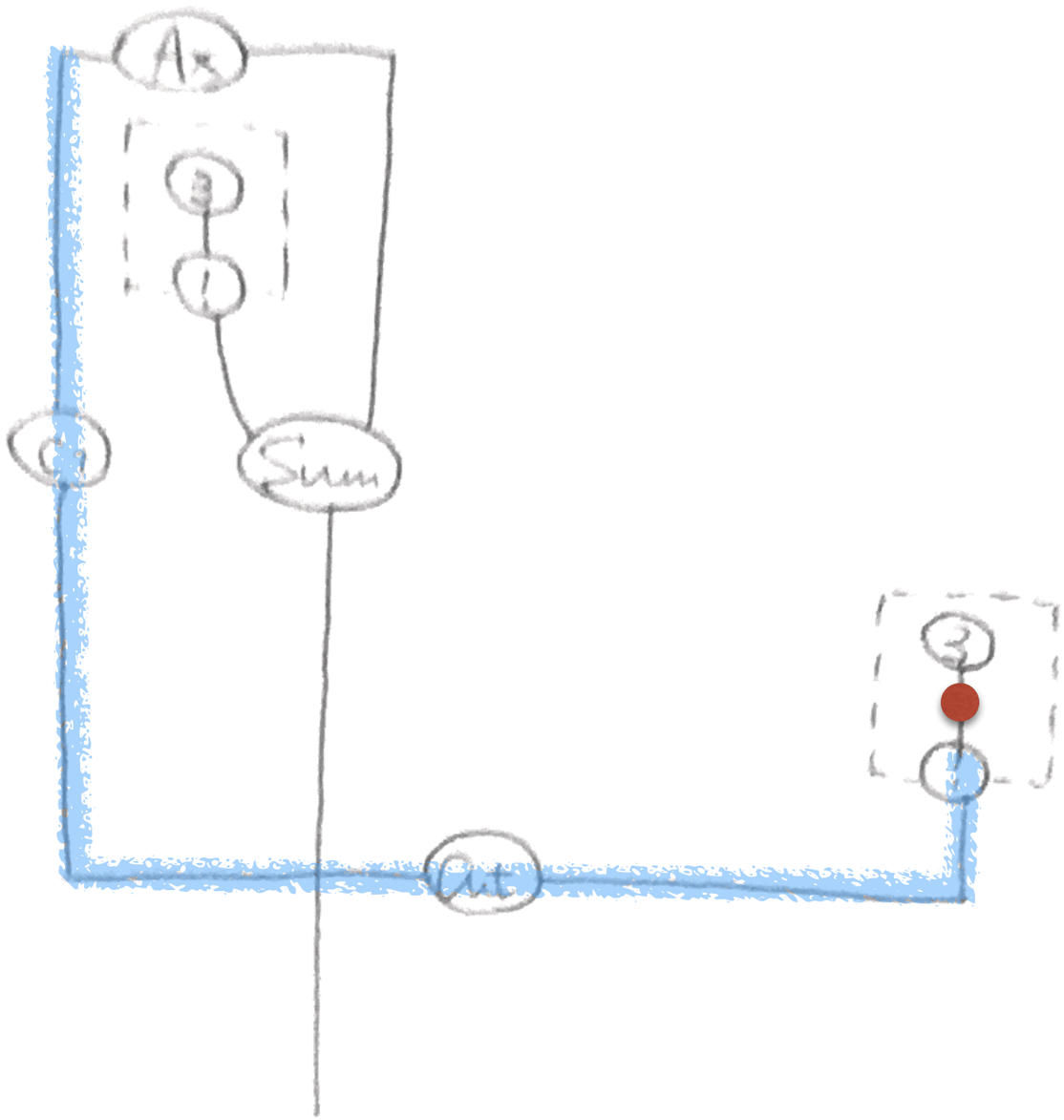
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



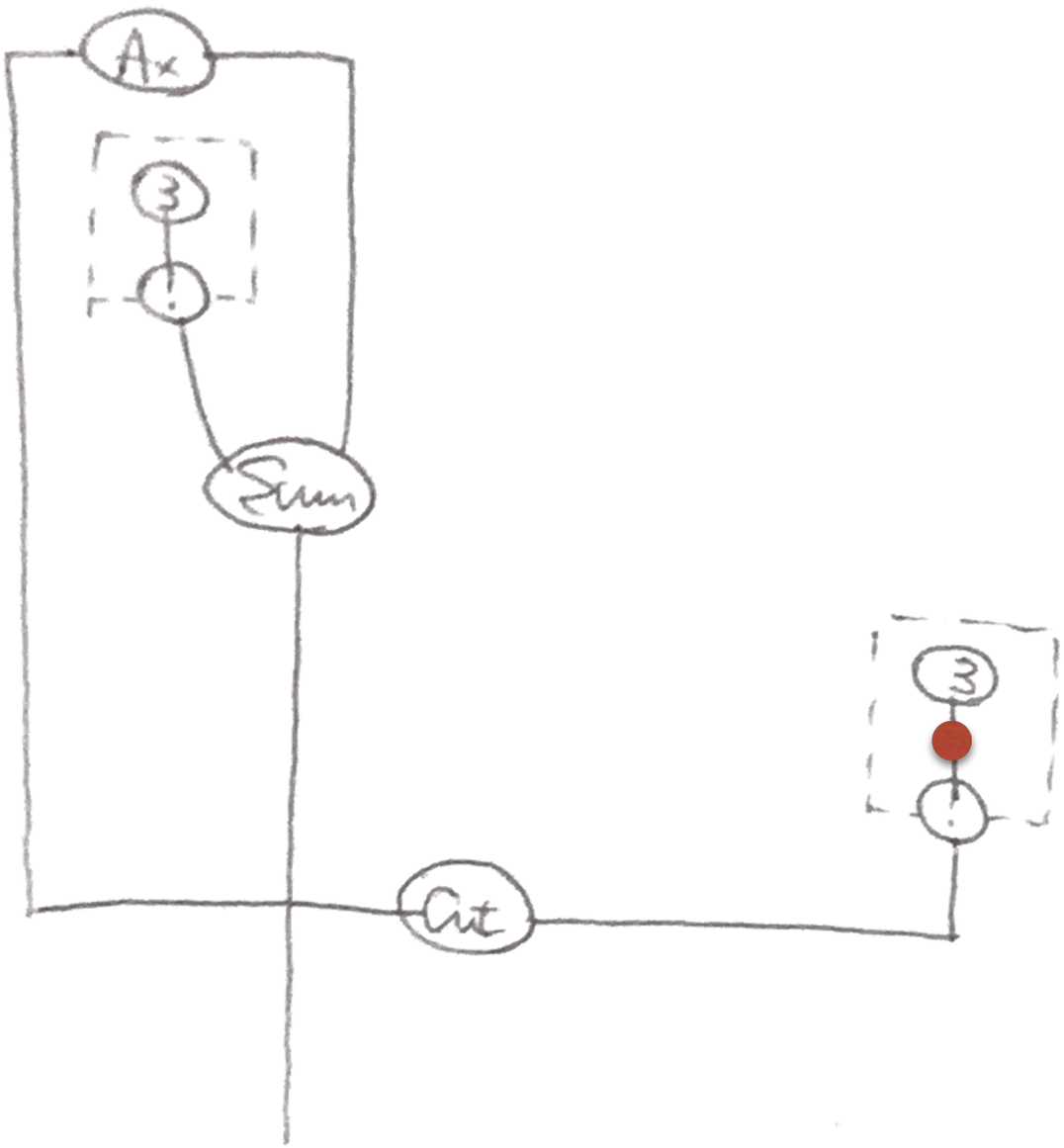
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



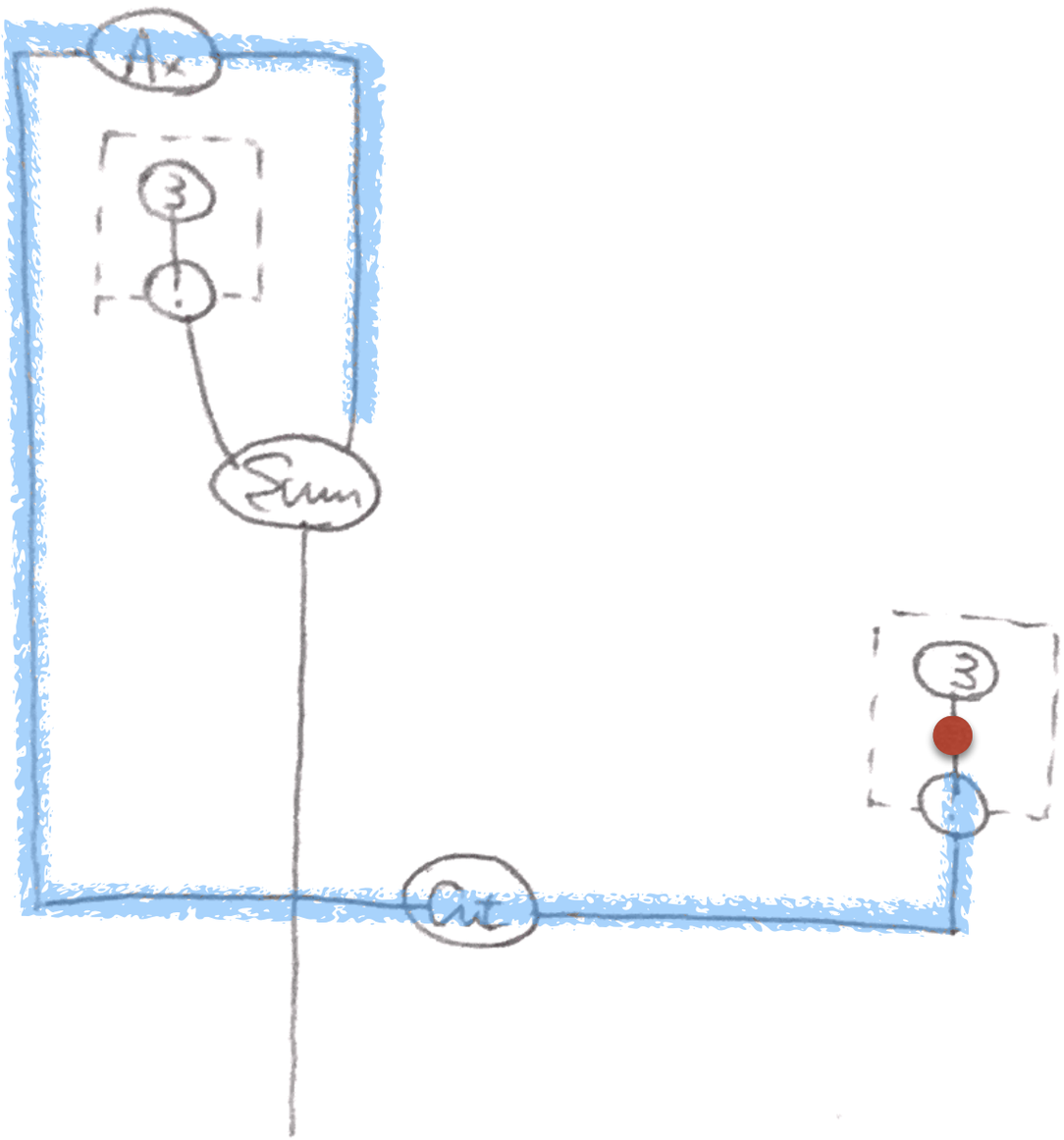
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$

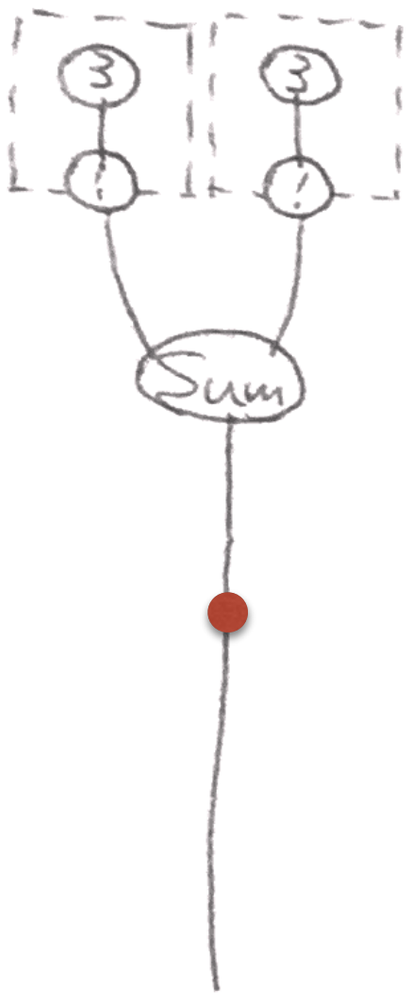


Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$

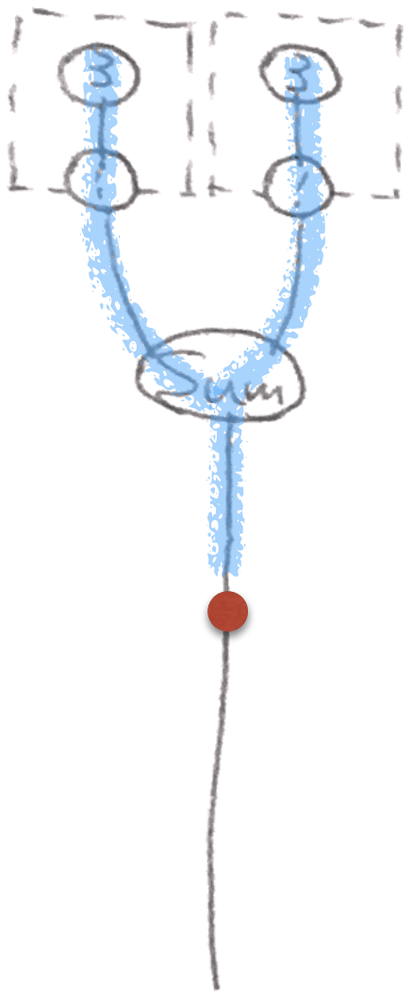
Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$

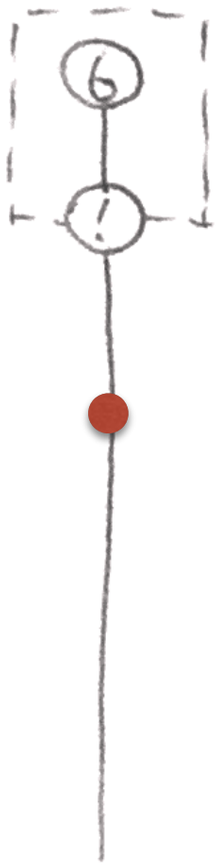


Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$

Avoid re-evaluation: dynamic rewrite

$$(\lambda x\,.\,x + x)\,(1 + 2)$$



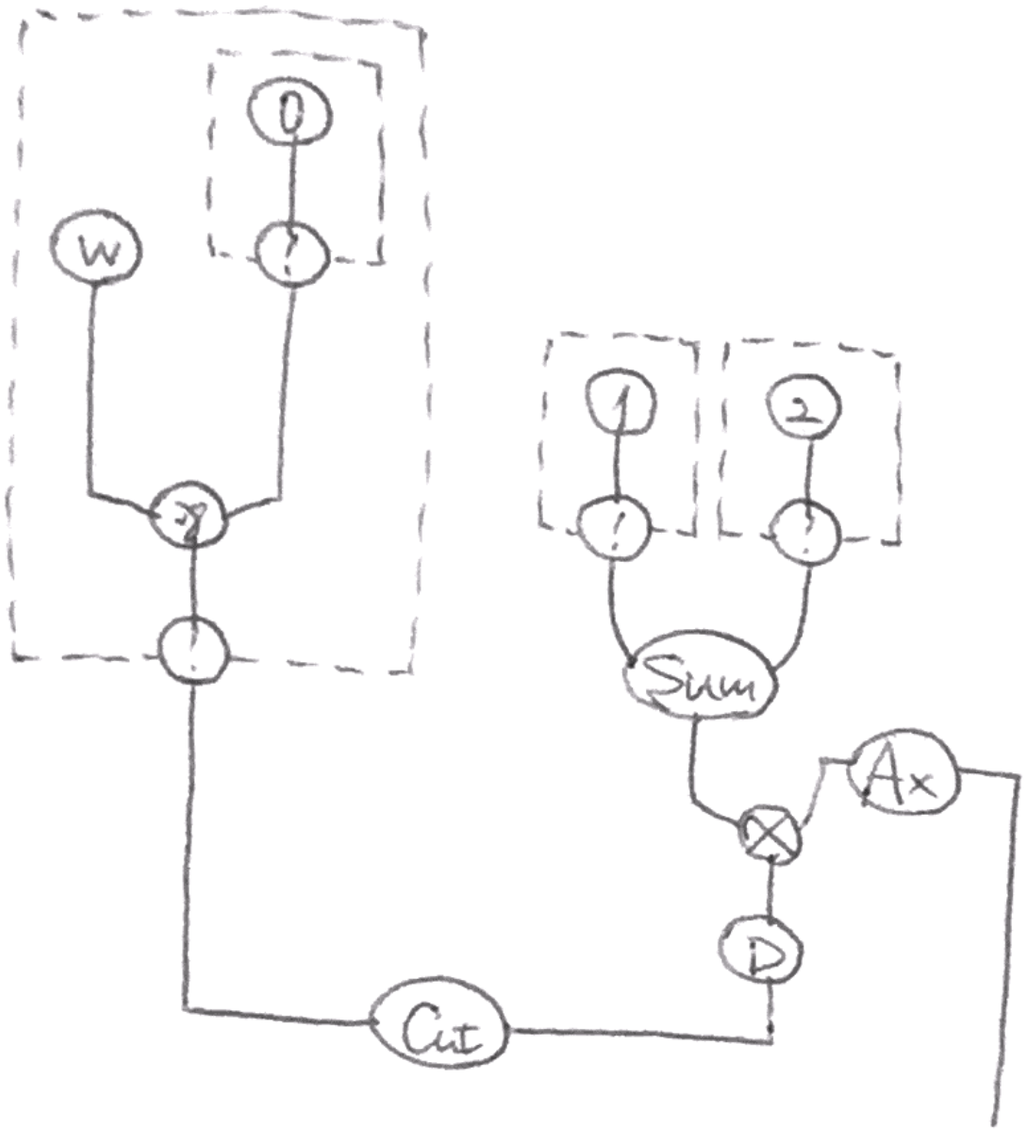
Dynamic GoI machine

- ~~call-by-name~~ **call-by-need**
  - a token on a ~~*static*~~ graph
    - dynamic rewrites
- ~~space~~ **time** efficiency

avoid repeated evaluation

force evaluation of function arguments

Force evaluation of arguments



$$(\lambda x.\,0)\,(1+2)$$

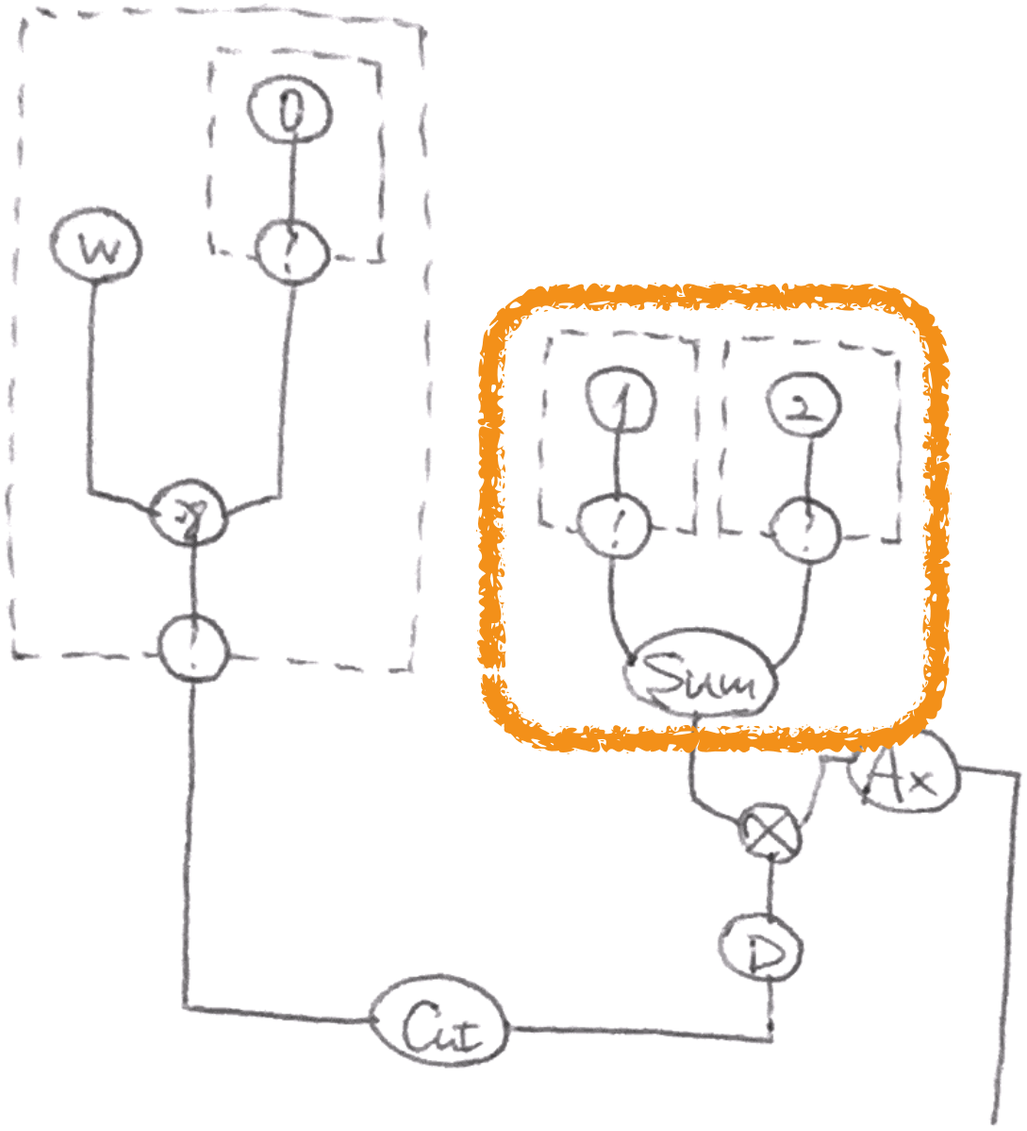
q

~~q~~

~~3~~

6

Force evaluation of arguments



$$(\lambda x.\,0)\,(1+2)$$

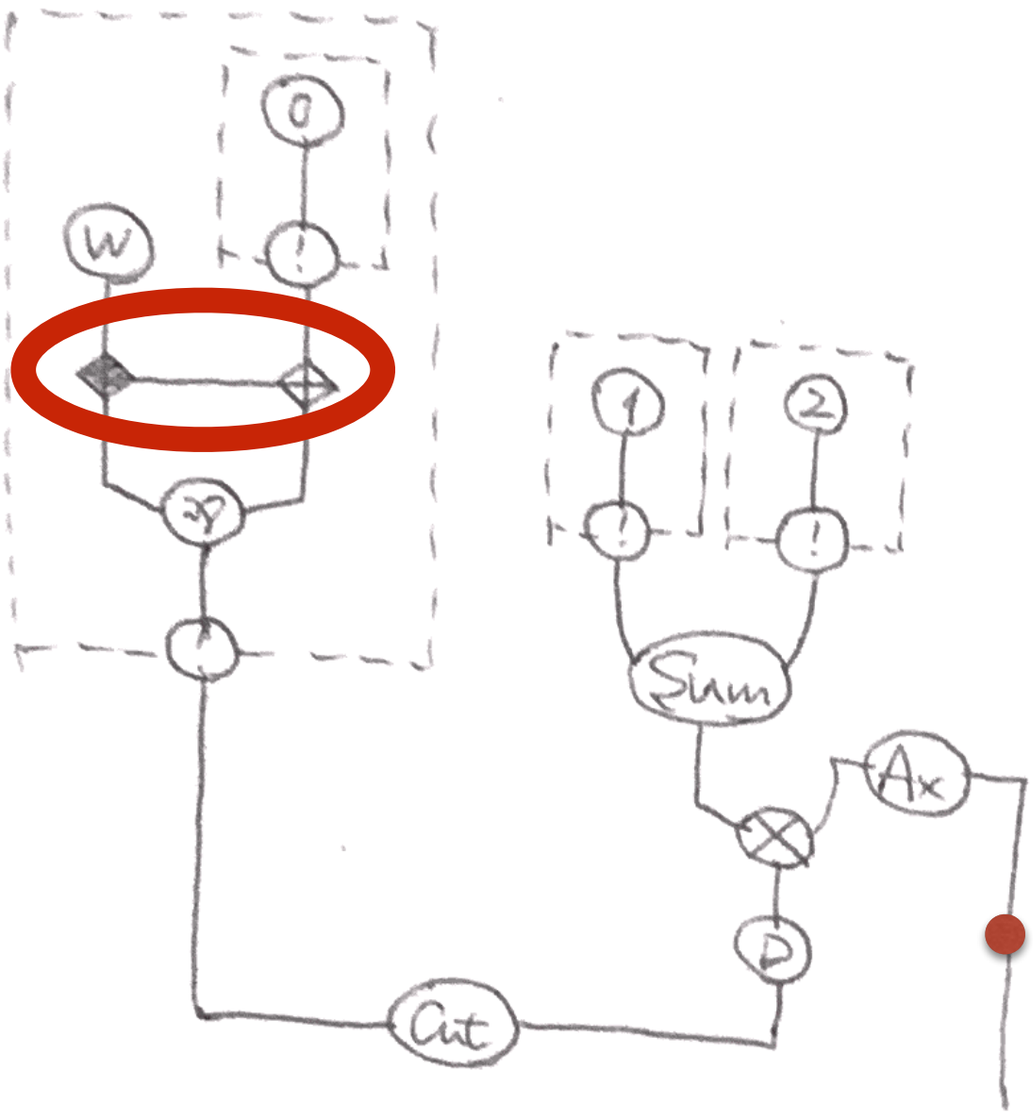
q

~~q~~
~~3~~

6

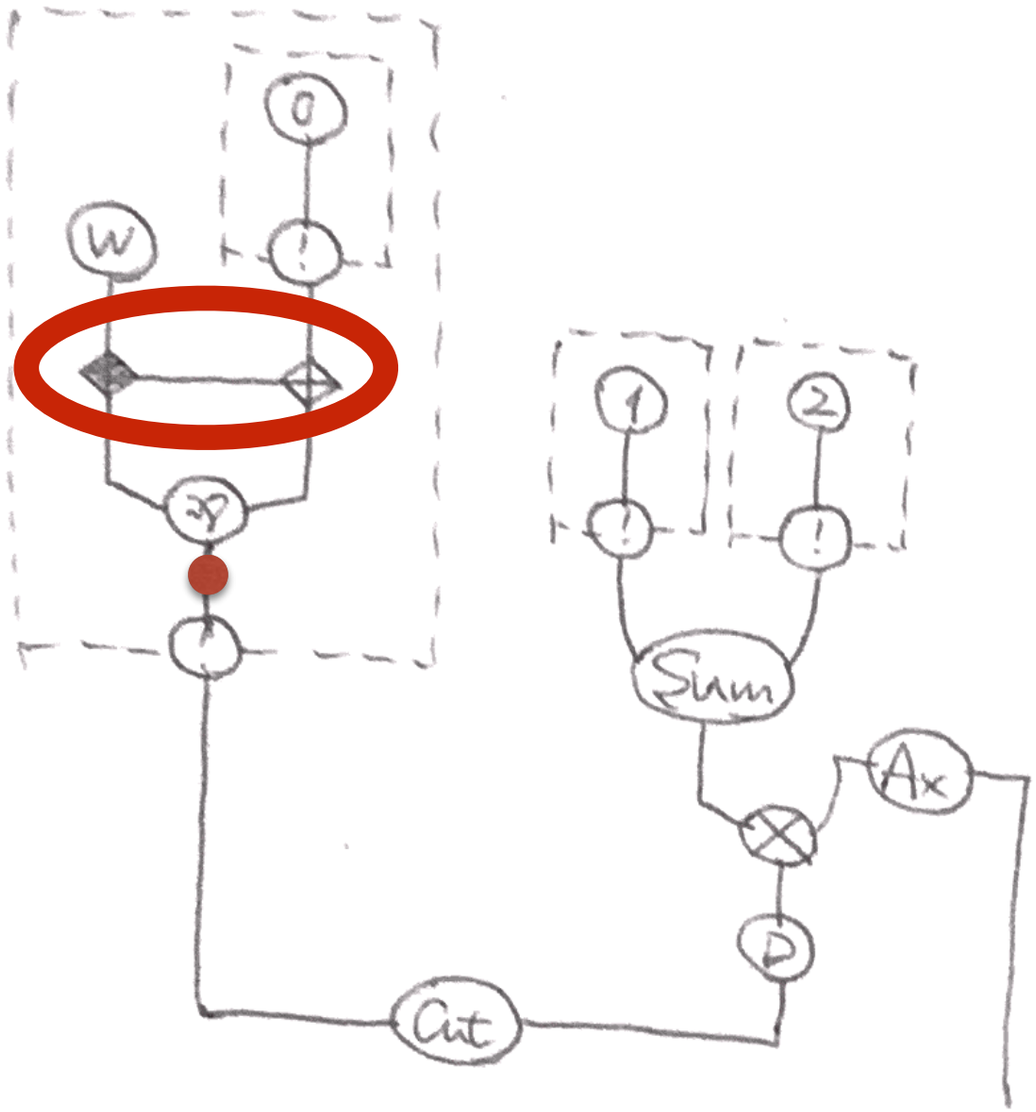
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



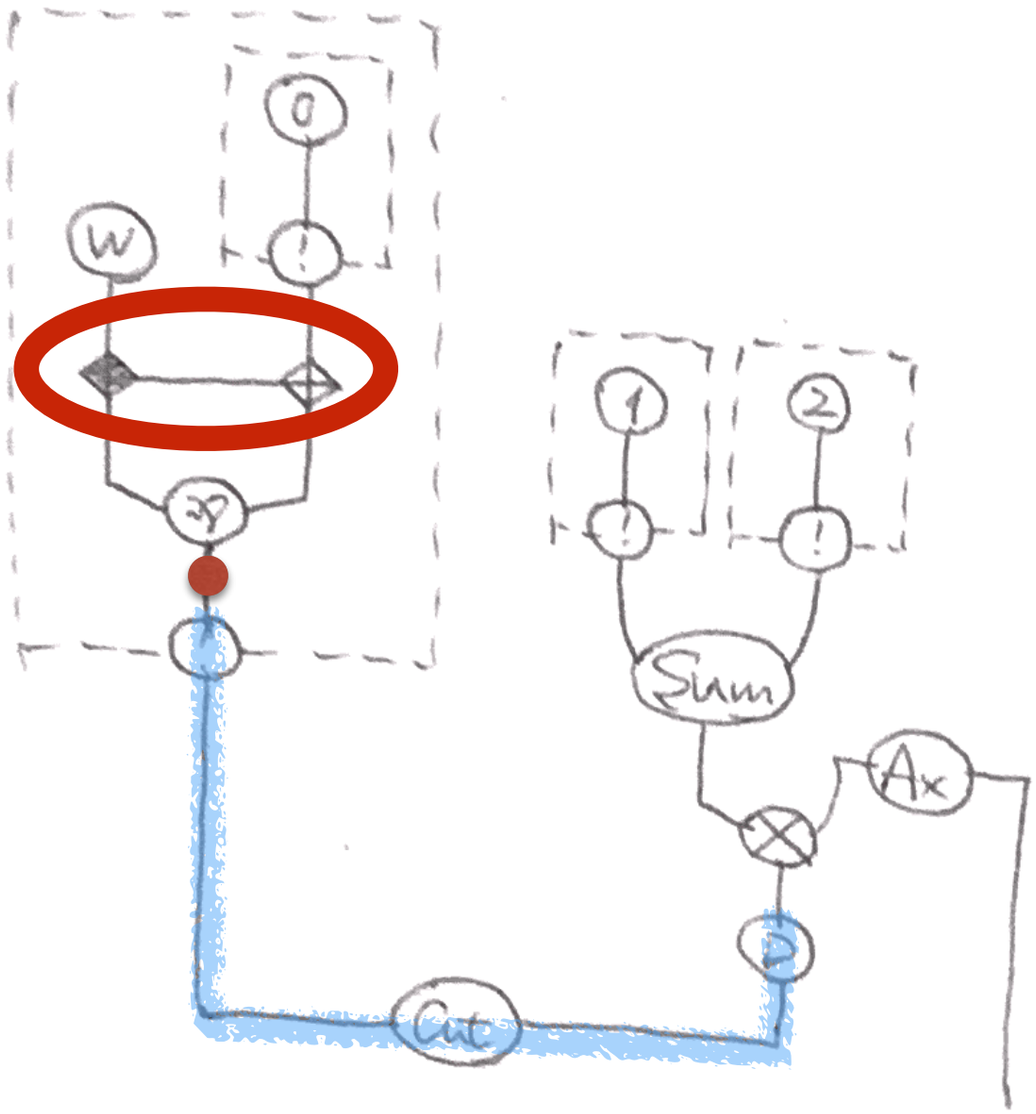
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



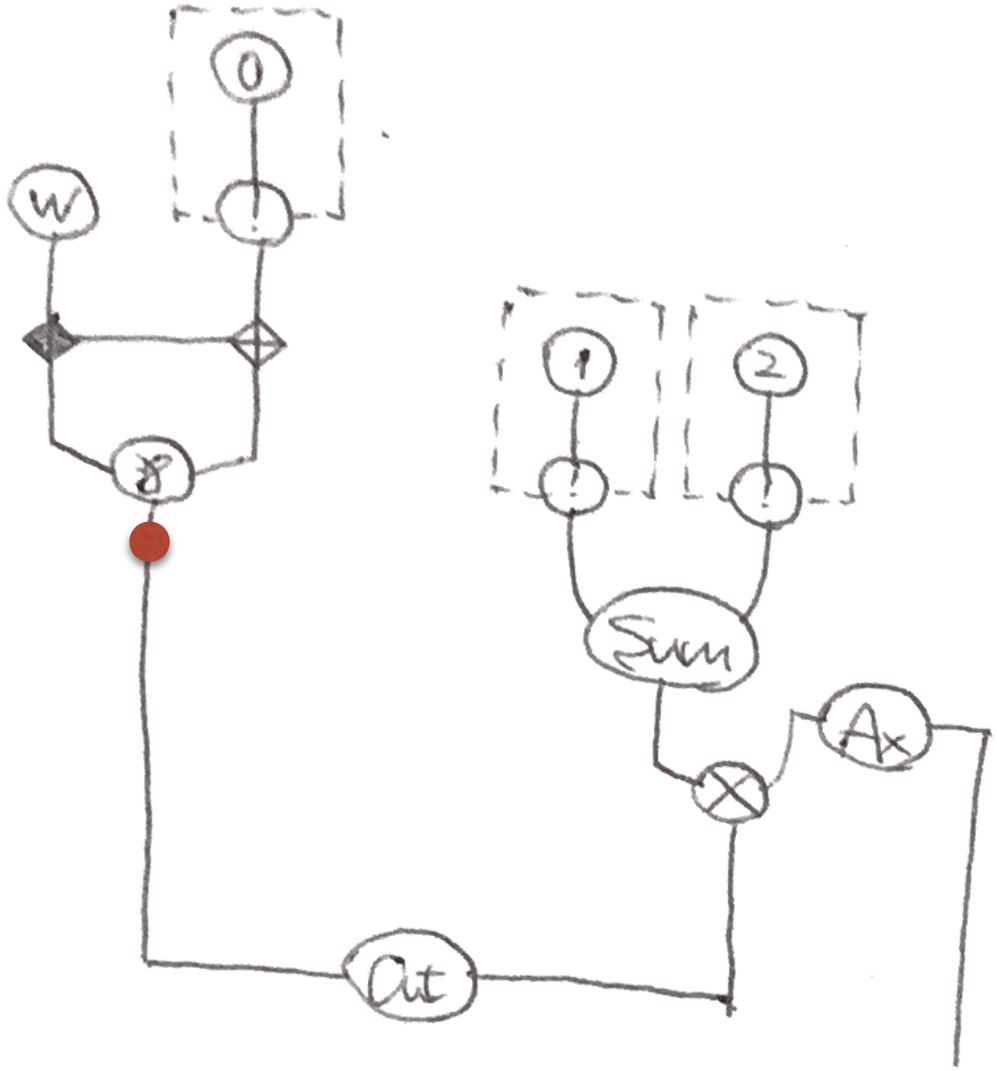
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



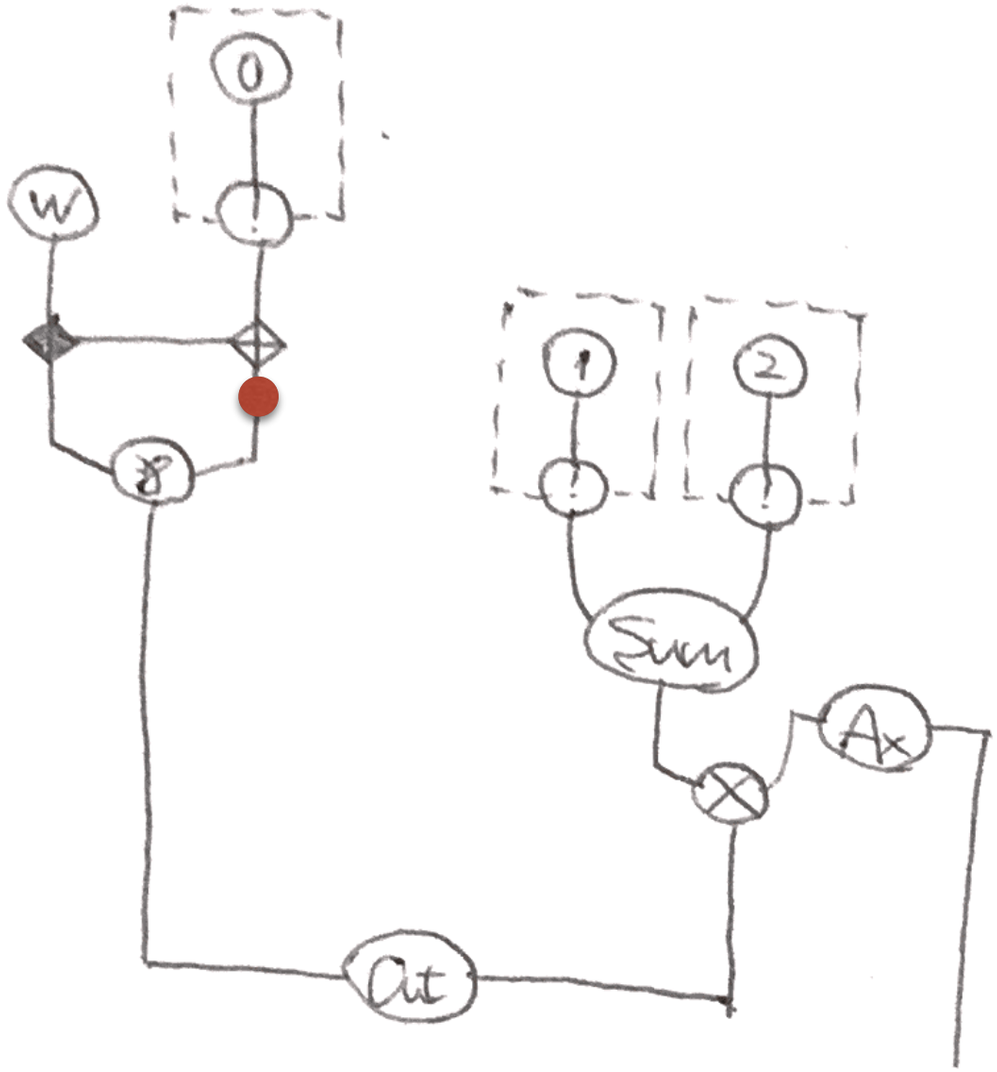
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



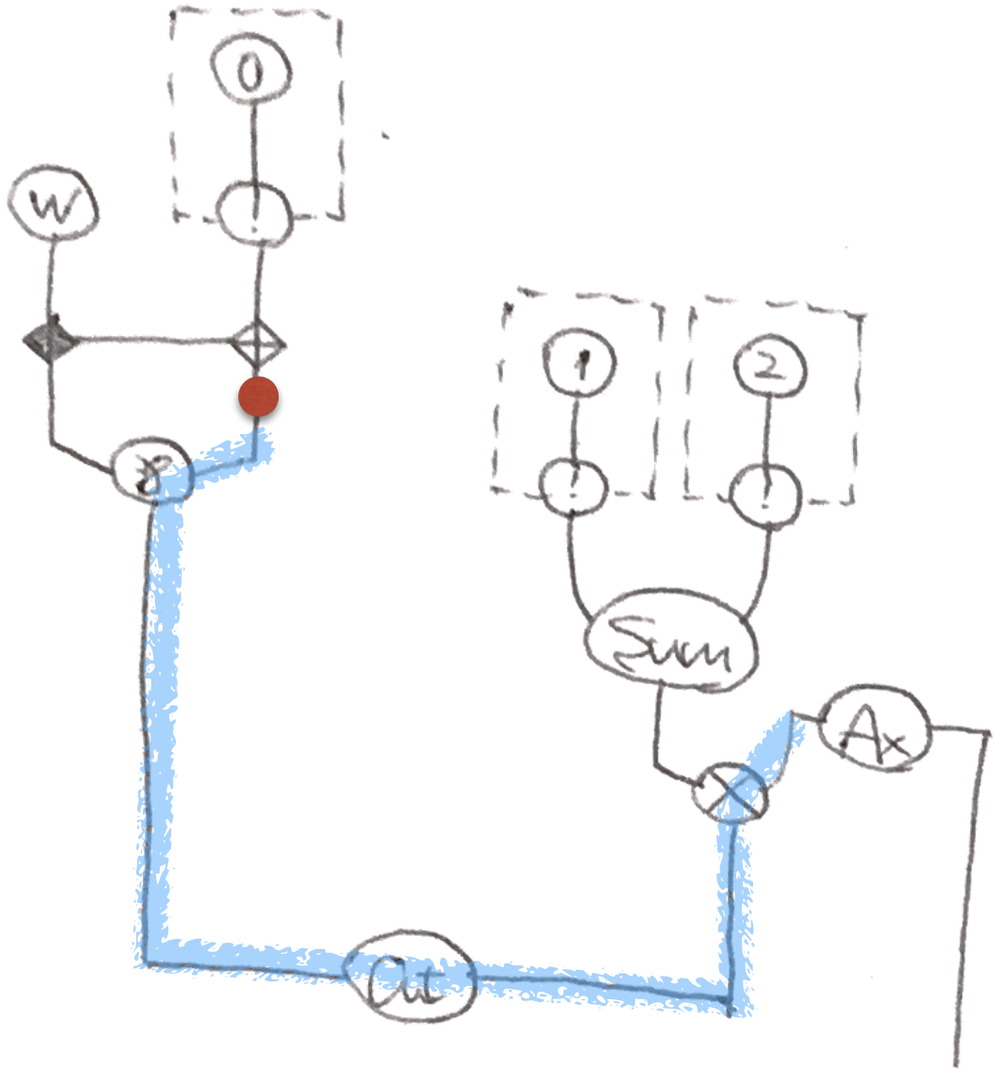
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



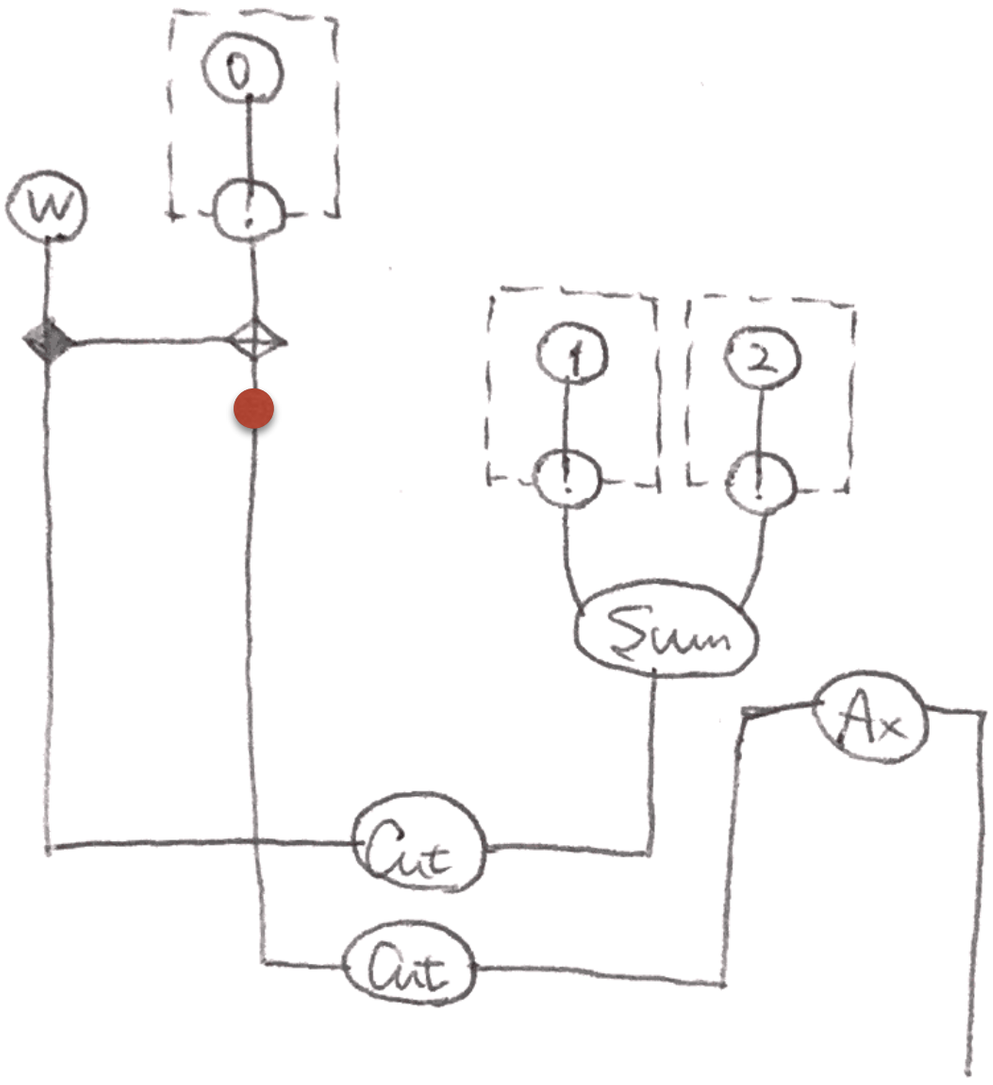
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



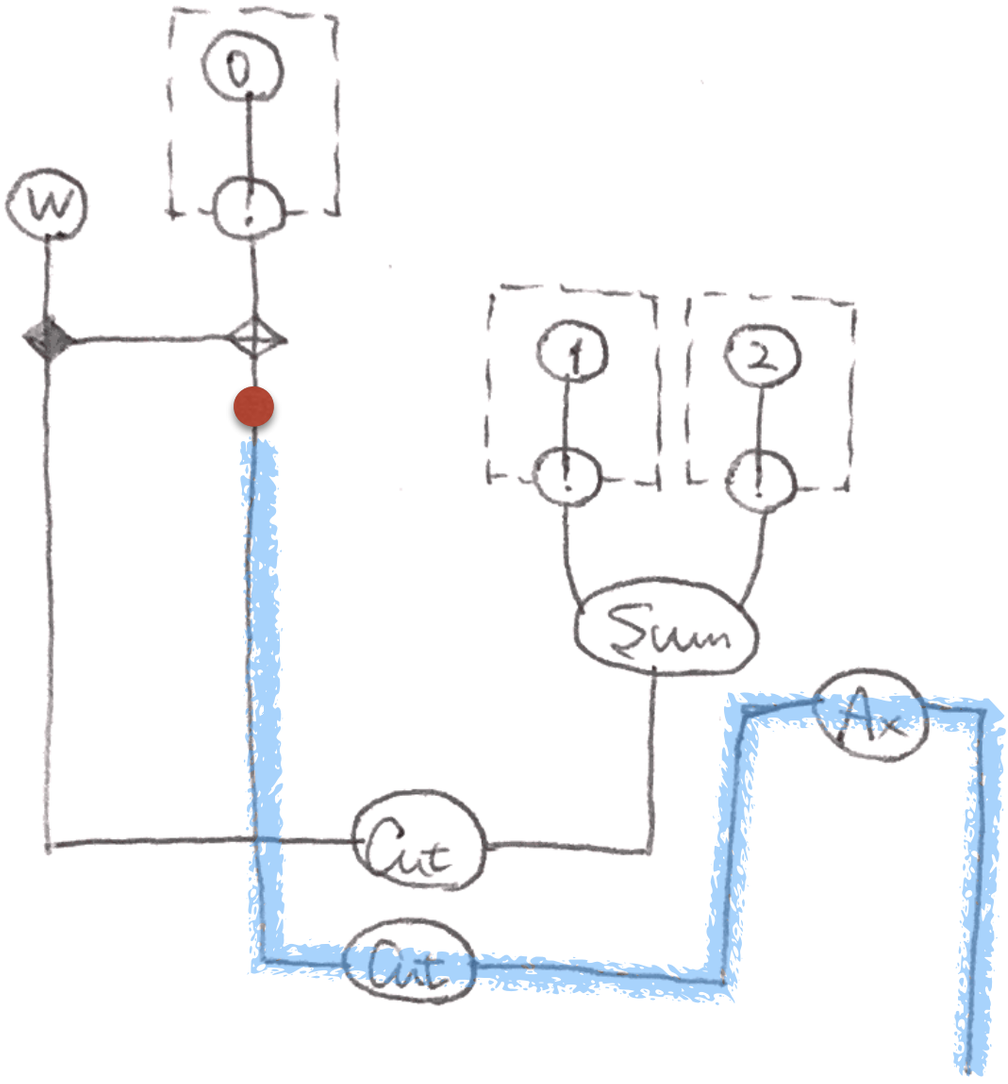
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



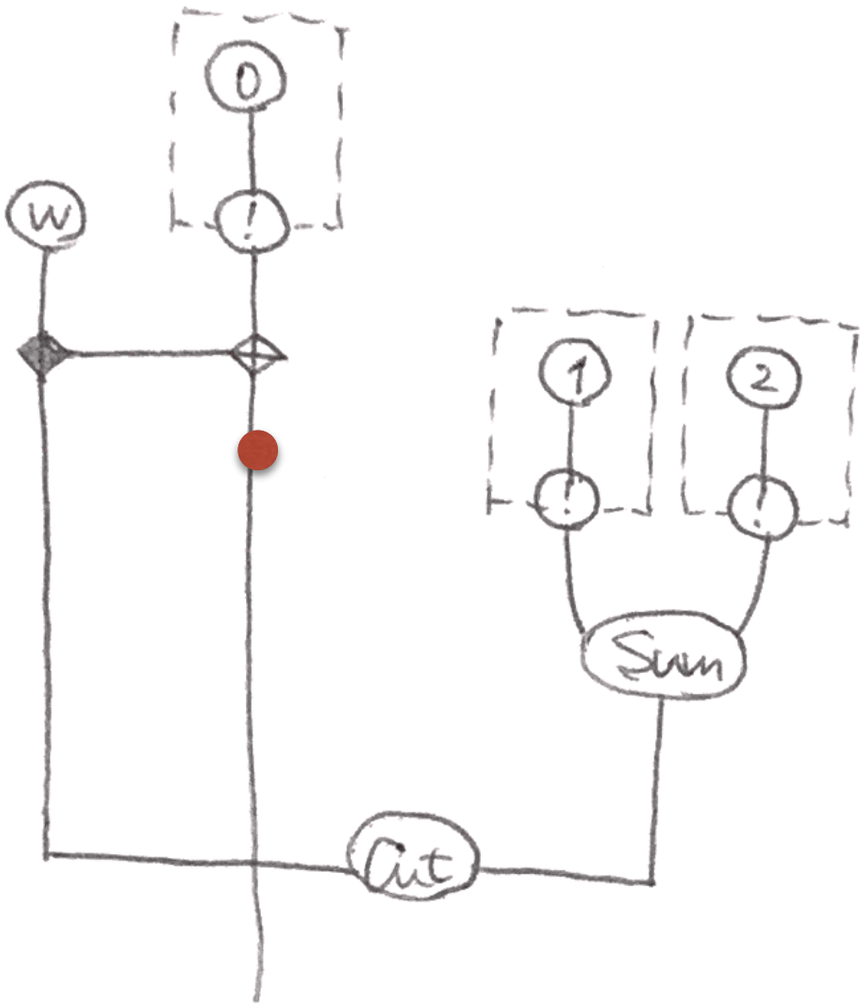
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



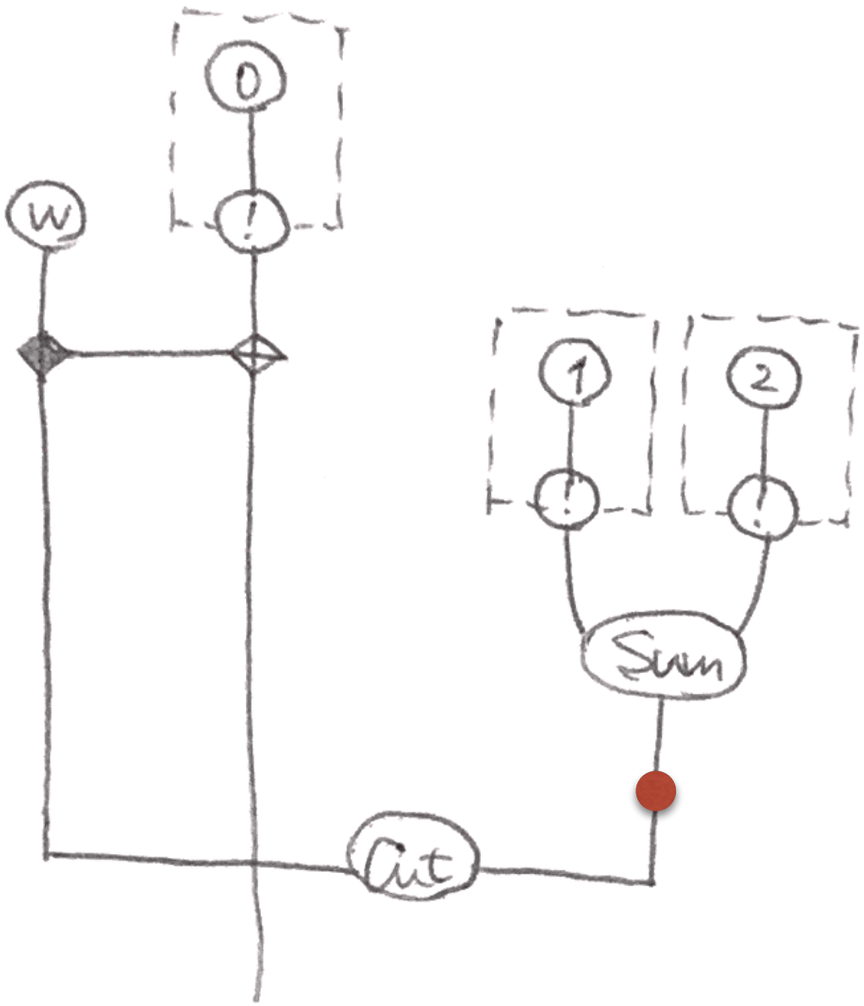
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



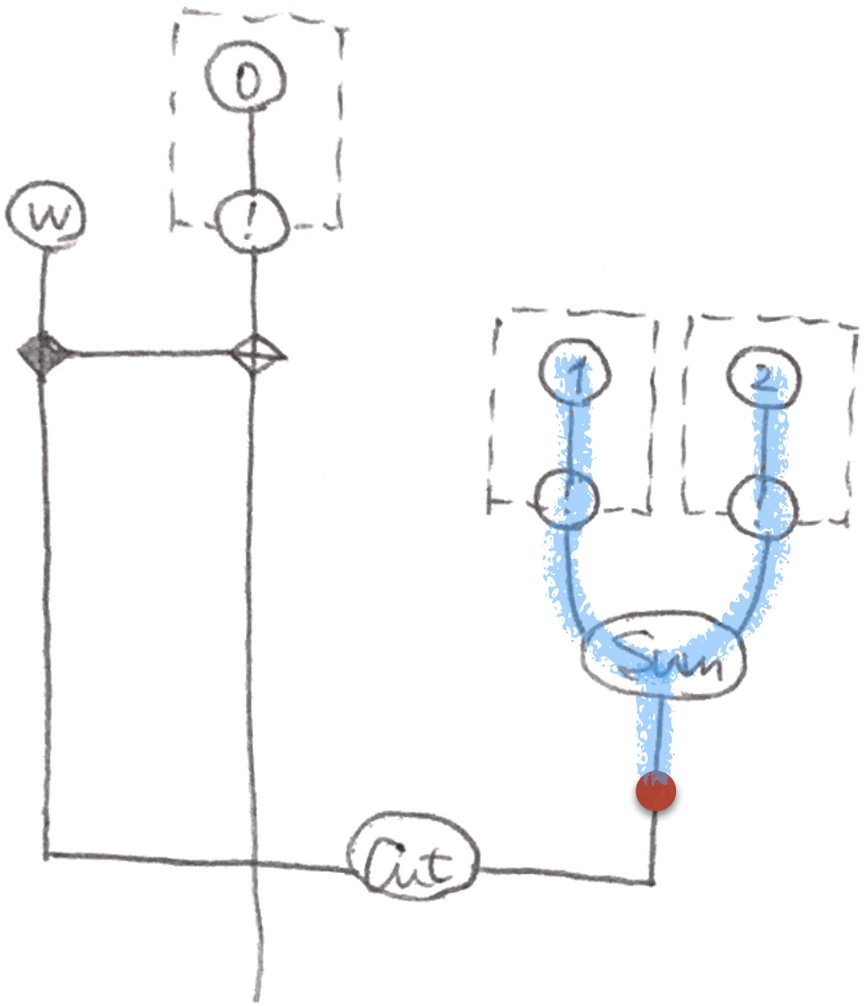
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



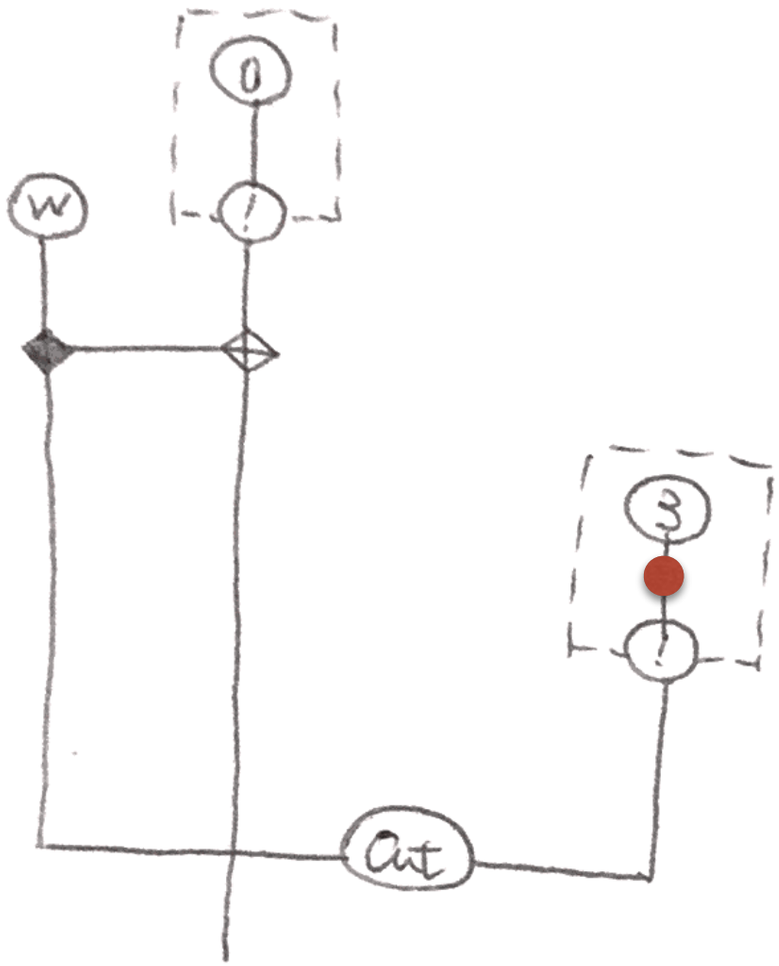
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



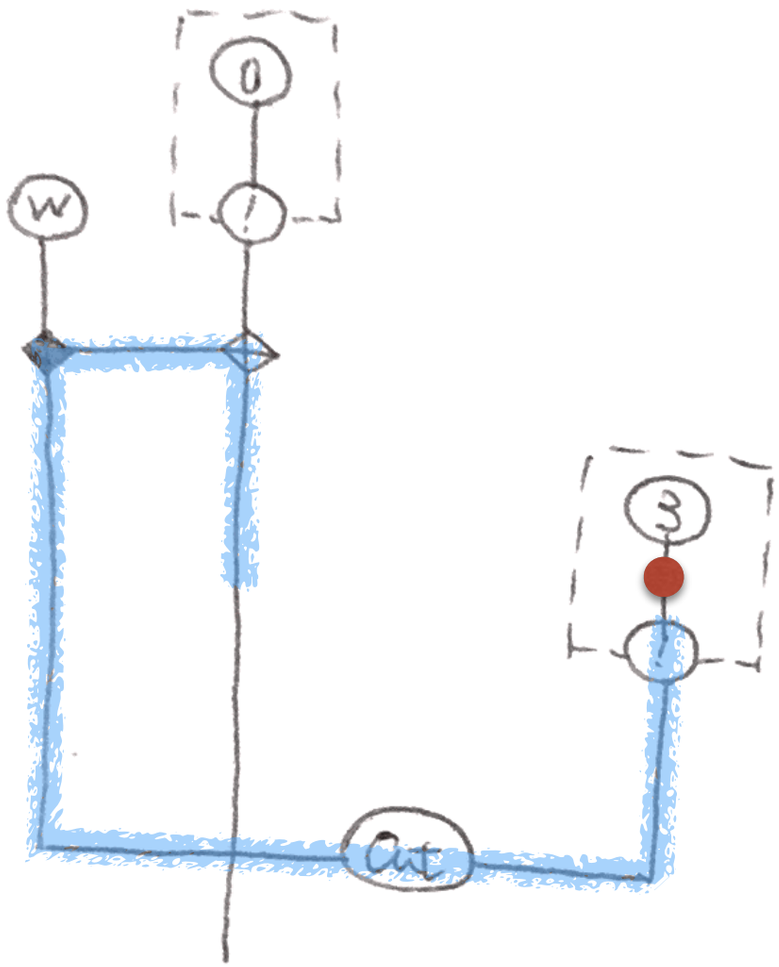
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



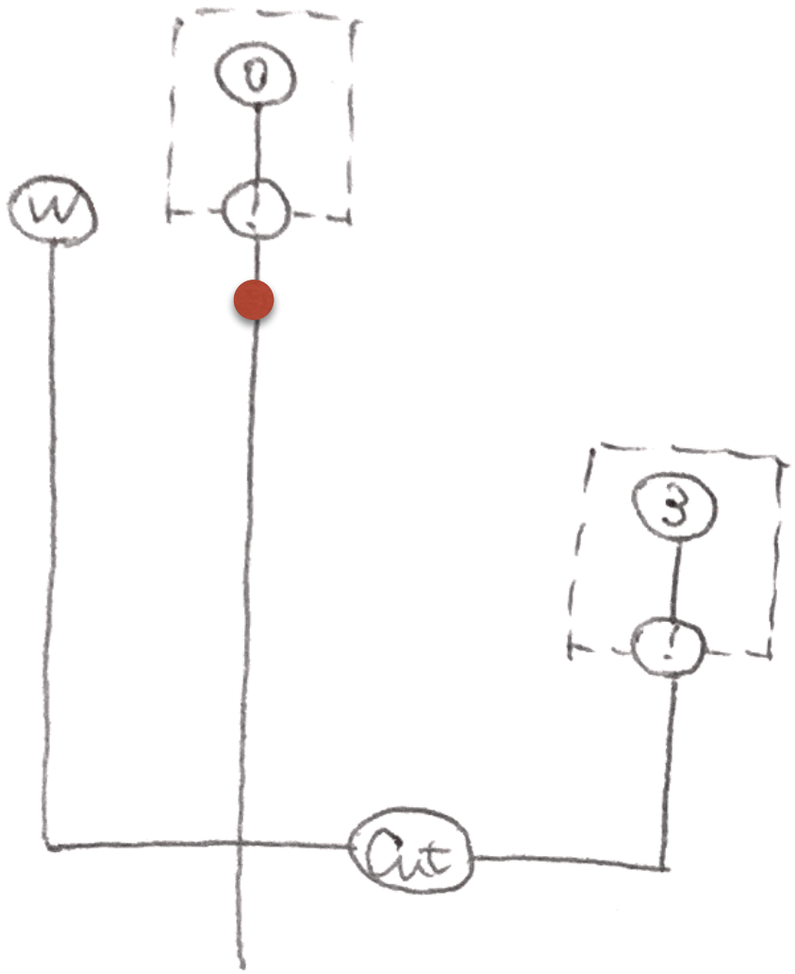
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



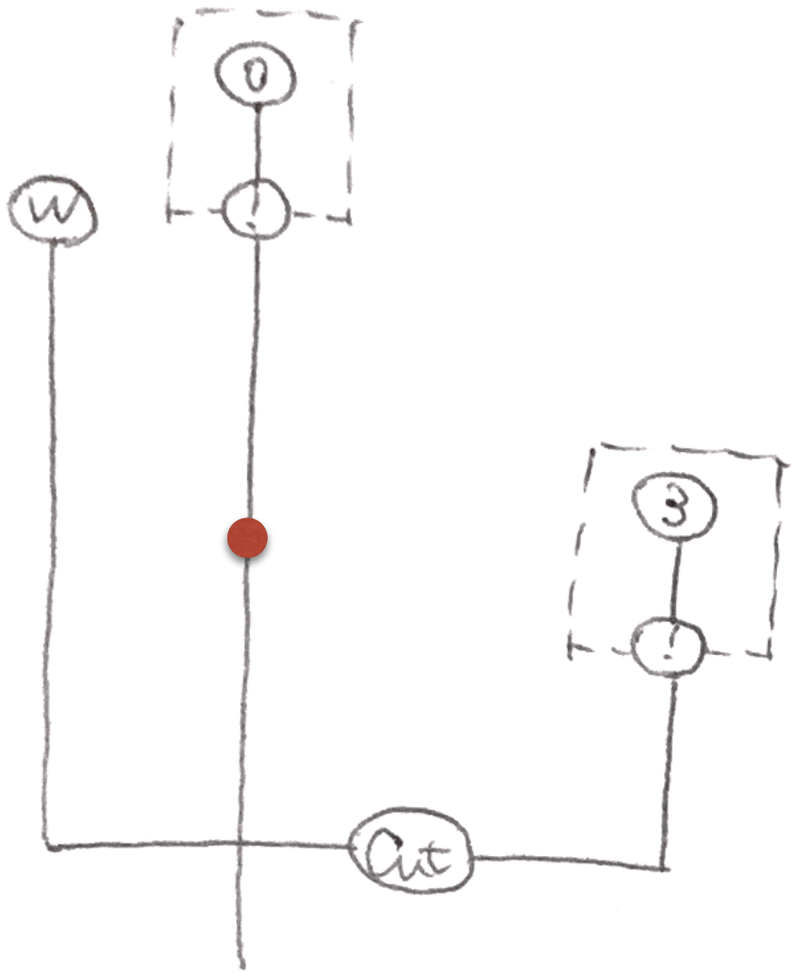
Force evaluation of arguments: checkpoint

$$(\lambda x.\,0)\,(1+2)$$



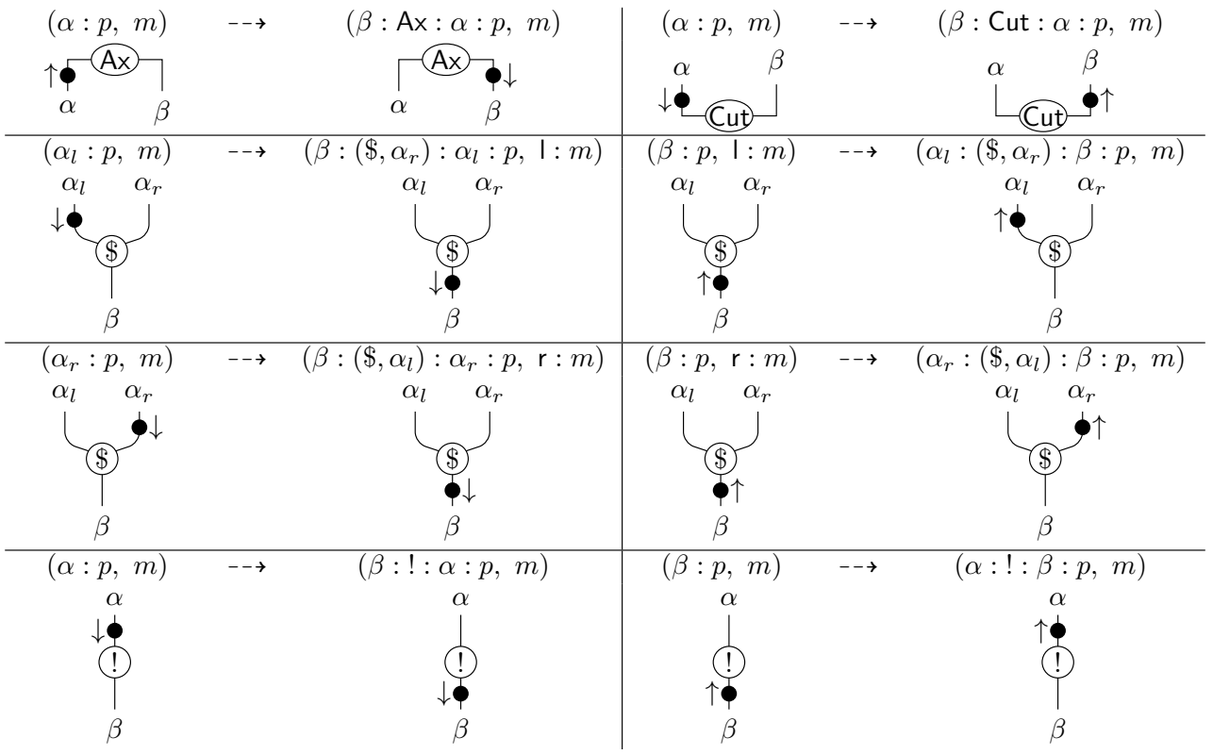
Force evaluation of arguments: checkpoint

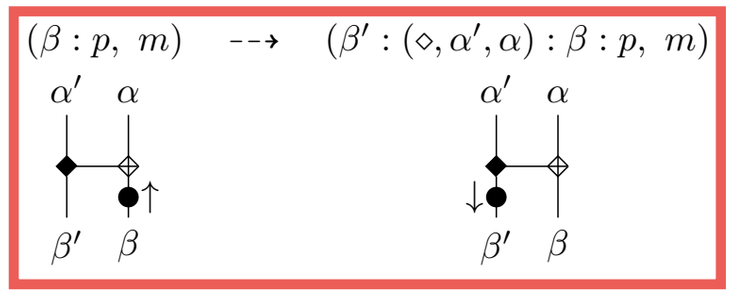
$$(\lambda x.\,0)\,(1+2)$$

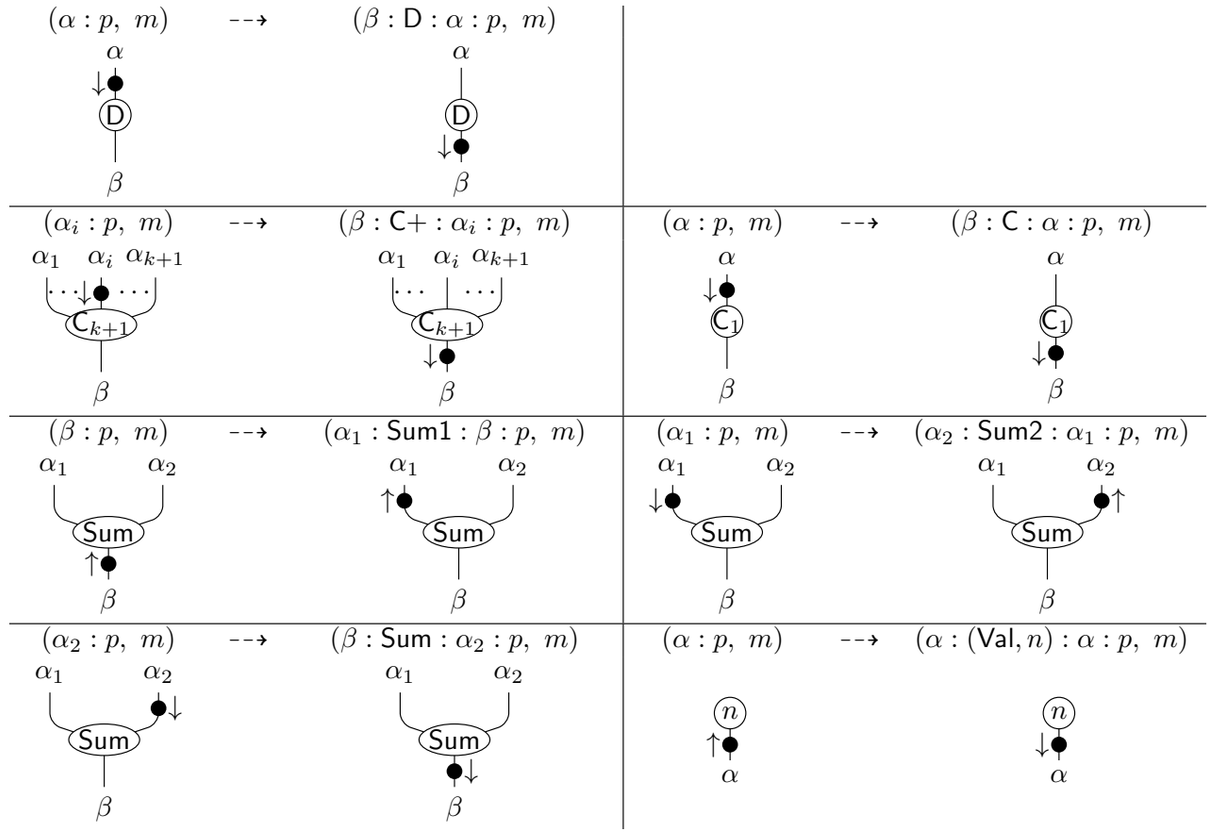


Dynamic GoI machine, extended

"move" transition $((G, \ell_e, \ell_b), p, d, m) \dashrightarrow ((G, \ell_e, \ell_b), p', d', m')$



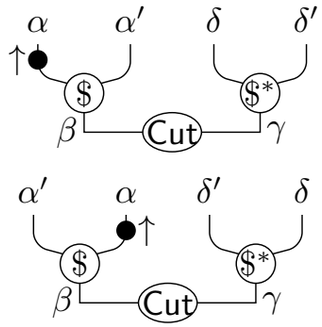




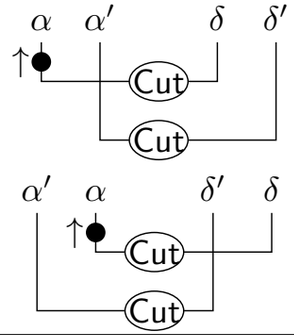
Dynamic GoI machine, extended

"rewrite" transition $((G, \ell_e, \ell_b), p, d, m) \rightsquigarrow ((G', \ell'_e, \ell'_b), p', d', m)$

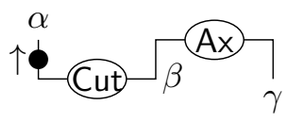
$\alpha : (\$, \alpha') : \beta : \mathsf{Cut} : \gamma : (\$^*, \delta') : \delta : p \quad \rightsquigarrow$



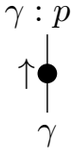
$\alpha : \mathsf{Cut} : \delta : p$



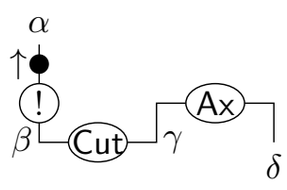
$\alpha : \mathsf{Cut} : \beta : \mathsf{Ax} : \gamma : p \quad \rightsquigarrow$



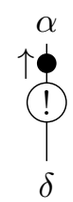
$\gamma : p$



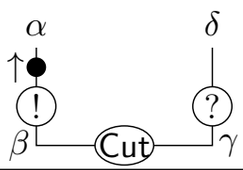
$\alpha : ! : \beta : \mathsf{Cut} : \gamma : \mathsf{Ax} : \delta : p \quad \rightsquigarrow$



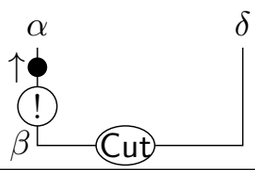
$\alpha : ! : \delta : p$



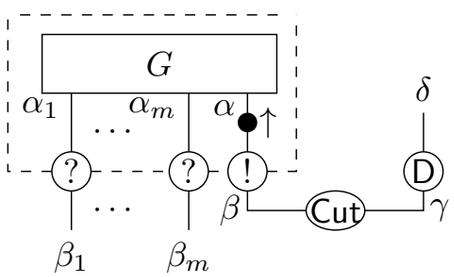
$\alpha : ! : \beta : \mathsf{Cut} : \gamma : ? : \delta : p \quad \rightsquigarrow$



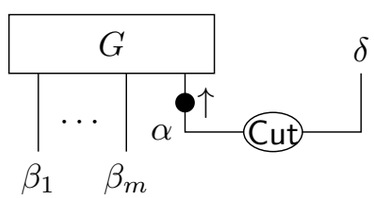
$\alpha : ! : \mathsf{Cut} : \delta : p$



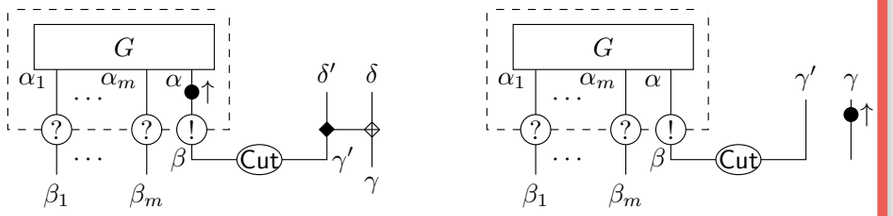
$\alpha : ! : \beta : \mathsf{Cut} : \gamma : \mathsf{D} : \delta : p \quad \rightsquigarrow$



$\alpha : \mathsf{Cut} : \delta : p$

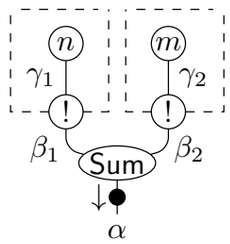


$\alpha : ! : \beta : \mathsf{Cut} : \gamma' : (\diamond, \delta', \delta) : \gamma : p \quad \rightsquigarrow$

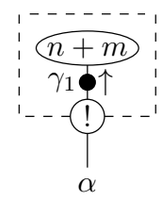


$\gamma : p$

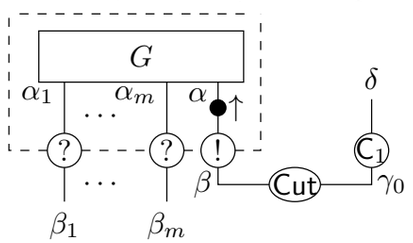
$\alpha : \mathsf{Sum} : \beta_2 : ! : \gamma_2 : (\mathsf{Val}, m)$
$: \gamma_2 : ! : \beta_2 : \mathsf{Sum2} : \beta_1 : ! : \gamma_1 : (\mathsf{Val}, n) \quad \rightsquigarrow$
$: \gamma_1 : ! : \beta_1 : \mathsf{Sum1} : \alpha : p$



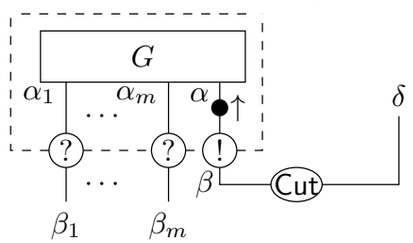
$\gamma_1 : ! : \alpha : p$



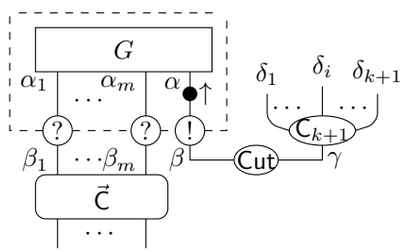
$\alpha : ! : \beta : \mathsf{Cut} : \gamma : \mathsf{C} : \delta : p \quad \rightsquigarrow$



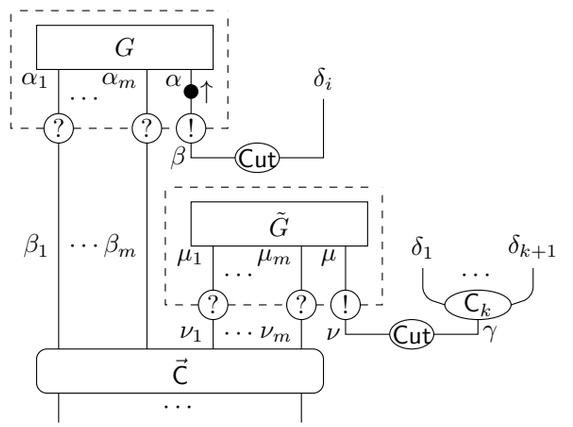
$\alpha : ! : \beta : \mathsf{Cut} : \delta : p$



$\alpha : ! : \beta : \mathsf{Cut} : \gamma : \mathsf{C}+ : \delta_i : p \quad \rightsquigarrow$



$\alpha : ! : \beta : \mathsf{Cut} : \delta_i : p$



“Linear-cost” simulation

- Dynamic GoI machine $\quad \to \overset{\text{def.}}{\Longleftrightarrow} \begin{cases} \rightsquigarrow & \text{if a rewrite is possible} \\ \dashrightarrow & \text{if no rewrite but a move is possible} \end{cases}$

- Call-by-value storeless abstract machine

$$\langle V, E\rangle_t \to_{\mathrm{val}} \langle E, V\rangle_c$$
$$\langle M\ N, E\rangle_t \to_{\mathrm{val}} \langle M, E[[\ ]\ N]\rangle_t$$
$$\langle M + N, E\rangle_t \to_{\mathrm{val}} \langle M, E[[\ ] + N]\rangle_t$$
$$\langle x, E_1[\texttt{let}\ x = V\ \texttt{in}\ E_2]\rangle_t \to_{\mathrm{val}} \langle E_1[\texttt{let}\ x = V\ \texttt{in}\ E_2], V\rangle_c$$

$$\langle [\ ], A[V]\rangle_c \to_{\mathrm{val}} \langle A[V]\rangle_a$$
$$\langle E[[\ ]\ N], A[\lambda x.\, M]\rangle_c \to_{\mathrm{val}} \langle N, E[A[\texttt{let}\ x' := [\ ]\ \texttt{in}\ M[x'/x]]]\rangle_t$$
$$\langle E[[\ ] + N], A[\underline{n}]\rangle_c \to_{\mathrm{val}} \langle N, E[A[\underline{n} + [\ ]]]\rangle_t$$
$$\langle E[\underline{n} + [\ ]], A[\underline{m}]\rangle_c \to_{\mathrm{val}} \langle E, A[\underline{n+m}]\rangle_c$$
$$\langle E[\texttt{let}\ x = V'\ \texttt{in}\ [\ ]], A[V]\rangle_c \to_{\mathrm{val}} \langle E, \texttt{let}\ x = V'\ \texttt{in}\ A[V]\rangle_c$$
$$\langle E[\texttt{let}\ x := [\ ]\ \texttt{in}\ M], A[V]\rangle_c \to_{\mathrm{val}} \langle M, E[A[\texttt{let}\ x = V\ \texttt{in}\ [\ ]]]\rangle_t$$

Theorem A.2. There exists a binary relation $\ddagger$ that satisfies

$$c \xrightarrow{k}_{\mathrm{val}} c' \ \wedge\ c \ddagger (G, p, d, m)$$
$$\Longrightarrow\ (G, p, d, m) \to \overset{\mathcal{O}(k)}{\cdots} \to (G', p', d', m') \ \wedge\ c' \ddagger (G', p', d', m')\ .$$

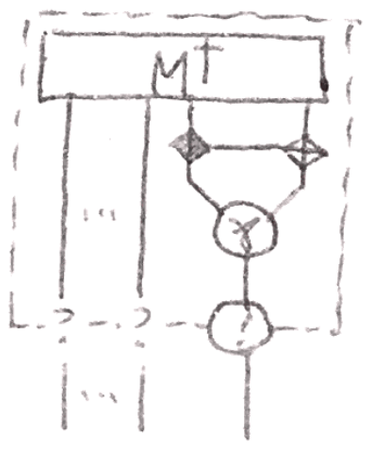
“Linear-cost” simulation

Theorem A.2. There exists a binary relation $\ddagger$ that satisfies

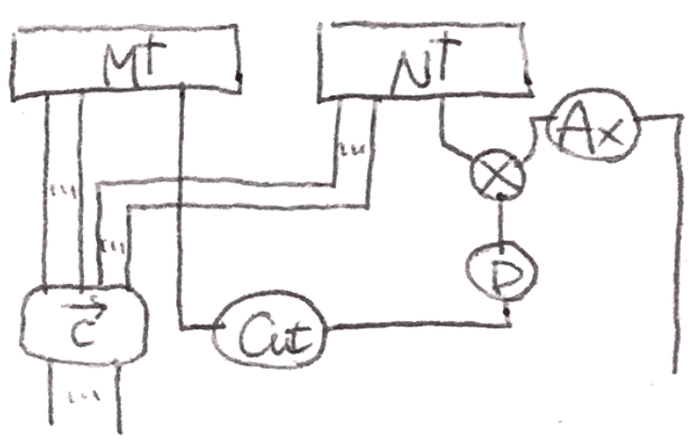
$$c \xrightarrow{k}_{\mathrm{val}} c' \ \wedge \ c \ddagger (G, p, d, m)$$
$$\Longrightarrow \ (G, p, d, m) \twoheadrightarrow^{\mathcal{O}(k)} (G', p', d', m') \ \wedge \ c' \ddagger (G', p', d', m') \ .$$

$x^\dagger :=$

$(\lambda x.\, M)^\dagger :=$



$(M\ N)^\dagger :=$



Dynamic GoI machine

avoid repeated evaluation

~~call-by-name~~ **call-by-need & call-by-value**

force evaluation of function arguments

- a token on a ~~*static*~~ graph
  - dynamic rewrites & “checkpoint” mechanism
- ~~space~~ **time** efficiency
- automated graph rewriter