SSA is Freyd Categories

Jad Ghalayini

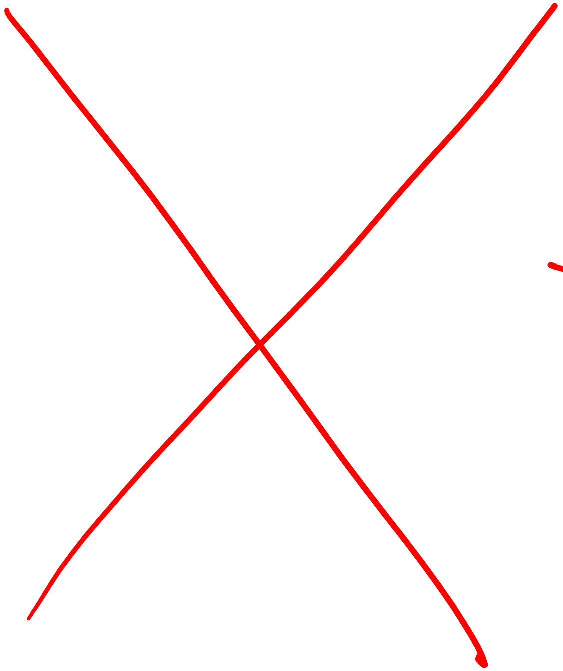
Neel Krishnaswami

2024-02-21T14:00Z
TUPLE'24
University of Edinburgh

Part I: What is SSA?

```
fn f(x: int, y: int) {
  z = y + y;
  y = x + 1;
  x = x - y;
  z = z + y;
  x = x + 1;
  y = y + x;
  z = x + 1;
  y = y + z;
  return y;
}
```

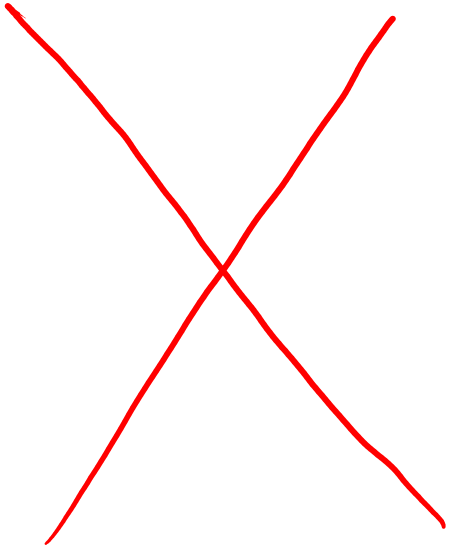
```
fn f(x: int, y: int) {
  z = y + y;
  y = x + 1;
  x = x - y;
  z = z + y;
  x = x + 1;
  y = y + x;
  z = x + 1;
  y = (x + 1) + z;
  return y;
}
```



```
fn f(x: int, y: int) {
    z = y + y;
    y = x + 1;
    x = x - y;
    z = z + y;
    x = x + 1;
  → y = y + x;
    z = x + 1;
    y = (x + 1) + z;
    return y;
}
```

```
fn f(x: int, y: int) {
  z = y + y;
  y = x + 1;
  x = x - y;
  z = z + y;
  x = x + 1;
  y = y + x;
  z = x + 1;
  y = y + z;
  return y;
}
```

```
fn f(x: int, y: int) {
  z = y + y;
  y = x + 1;
  x = x - y;
  z = z + (x + 1);
  x = x + 1;
  y = y + x;
  z = x + 1;
  y = y + z;
  return y;
}
```



```
fn f(x: int, y: int) {
  z = y + y;
  y = x + 1;
→ x = x - y;
  z = z + (x + 1);
  x = x + 1;
  y = y + x;
  z = x + 1;
  y = y + z;
  return y;
}
```

```
fn f(x0: int, y0: int) {
  z0 = y0 + y0;
  y1 = x0 + 1;
  x1 = x0 - y1;
  z1 = z0 + y1;
  x2 = x1 + 1;
  y2 = y1 + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

$y1 \neq y2$

✗

```
fn f(x0: int, y0: int) {
  z0 = y0 + y0;
  y1 = x0 + 1;
  x1 = x0 - y1;
  z1 = z0 + y1;
  x2 = x1 + 1;
  y2 = y1 + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  z0 = y0 + y0;
  y1 = x0 + 1;
  x1 = x0 - y1;
  z1 = z0 + (x0 + 1);
  x2 = x1 + 1;
  y2 = y1 + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  z0 = y0 + y0;
  y1 = x0 + 1;
  x1 = x0 - y1;
  z1 = z0 + (x0 + 1);
  x2 = x1 + 1;
  y2 = y1 + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  z0 = y0 + y0;
  y1 = x0 + 1;
  x1 = x0 - (x0 + 1);
  z1 = z0 + (x0 + 1);
  x2 = x1 + 1;
  y2 = (x0 + 1) + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  z0 = y0 + y0;
  y1 = x0 + 1;
  x1 = -1 ;
  z1 = z0 + (x0 + 1);
  x2 = x1 + 1;
  y2 = (x0 + 1) + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  ~~z0 = y0 + y0;~~
  ~~y1 = x0 + 1;~~
  x1 = -1;
  ~~z1 = z0 + (x0 + 1);~~
  x2 = x1 + 1;
  y2 = (x0 + 1) + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  x1 = -1;
  x2 = x1 + 1;
  y2 = (x0 + 1) + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  x2 = -1 + 1;
  y2 = (x0 + 1) + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  x2 = 0;
  y2 = (x0 + 1) + x2;
  z2 = x2 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  y2 = (x0 + 1) + 0;
  z2 = 0 + 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  y2 = x0 + 1;
  z2 = 1;
  y3 = y2 + z2;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  y3 = (x0 + 1) + 1;
  return y3;
}
```

```
fn f(x0: int, y0: int) {
  return x0 + 2;
}
```

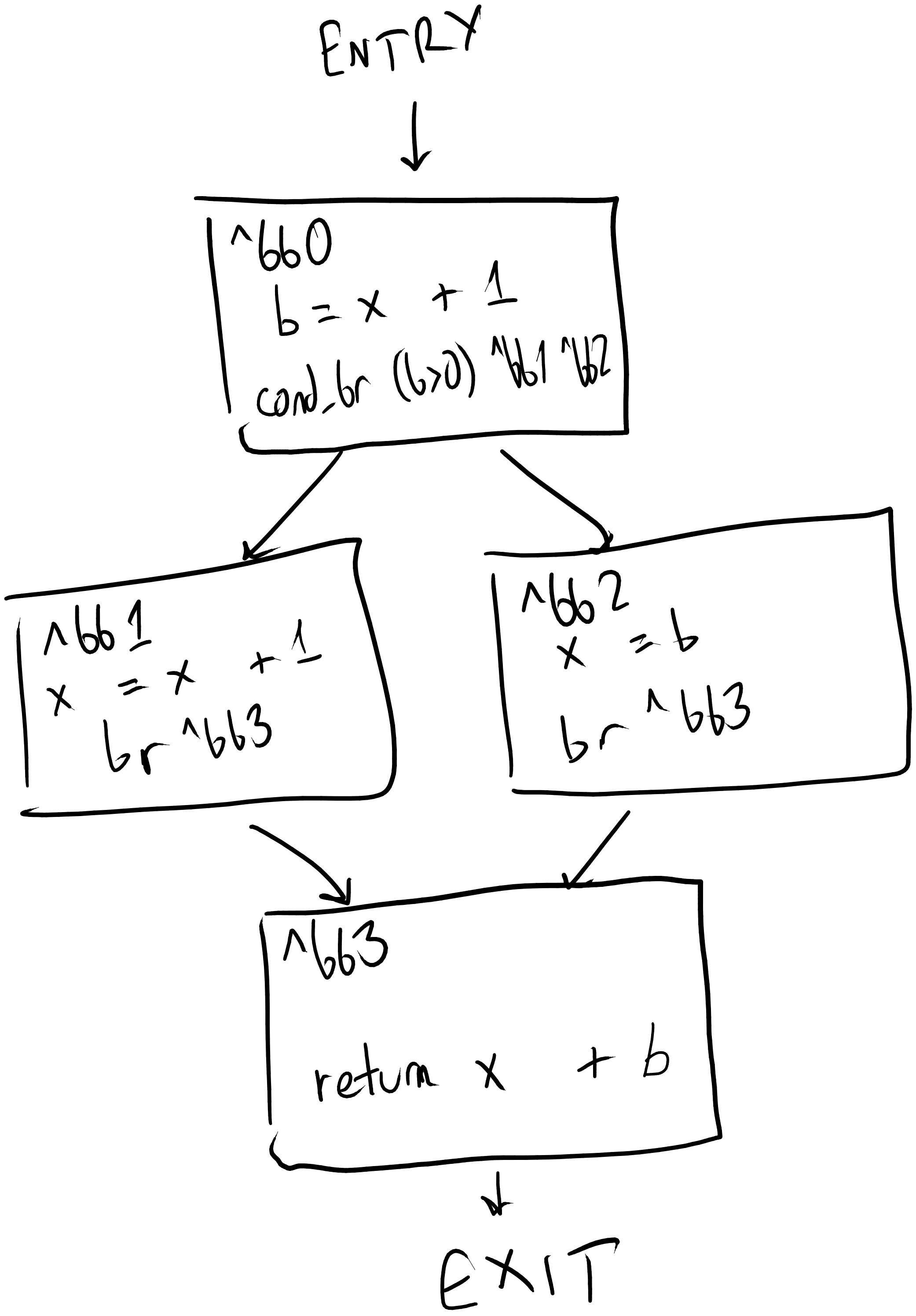
Static Single Assignment Property

$\implies$

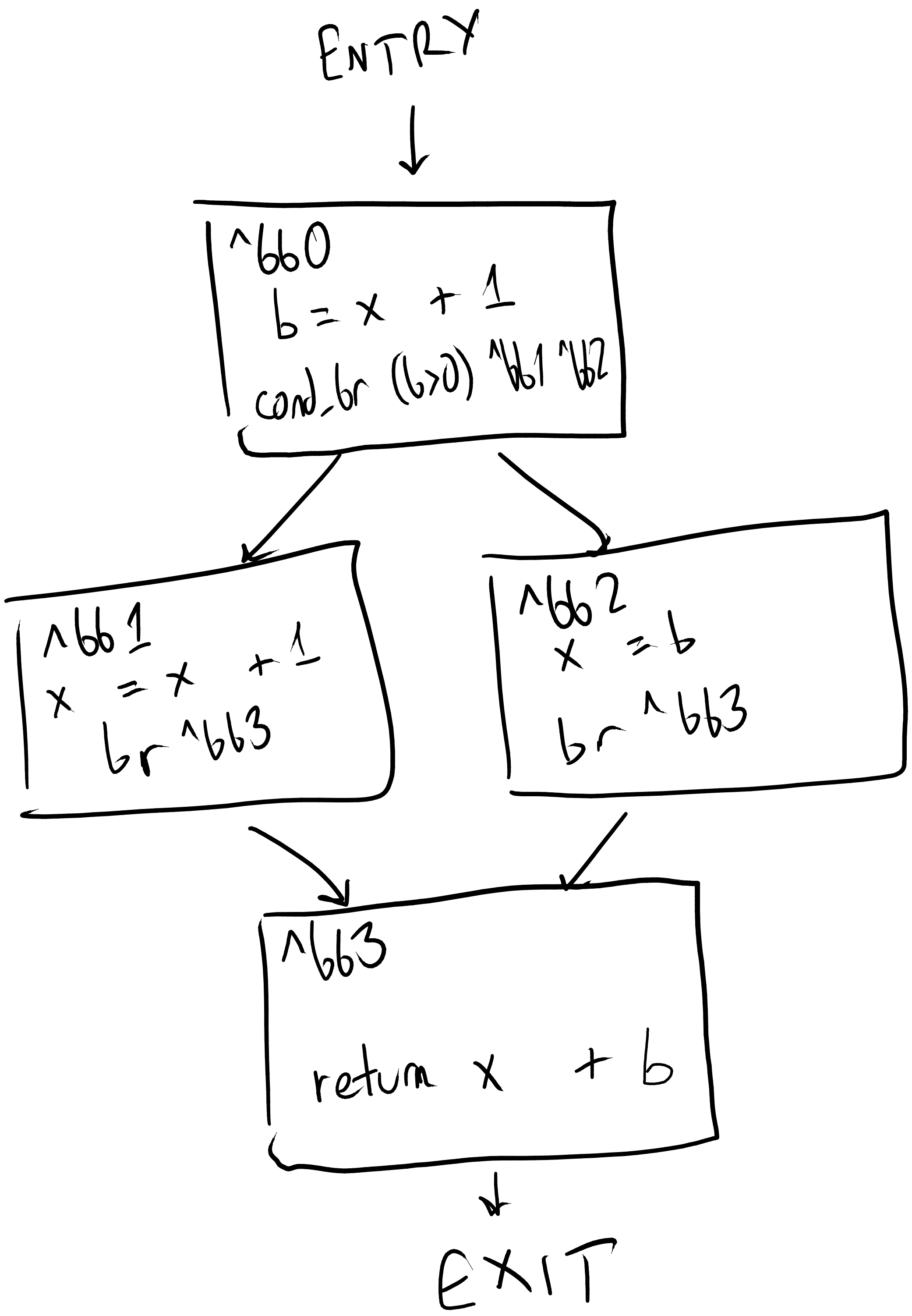
Algebraic Reasoning

```
fn f(x: int) {
  b = x + 1;
  if b > 0 {
    x = x + 1;
  } else {
    x = b;
  }
  return x + b;
}
```

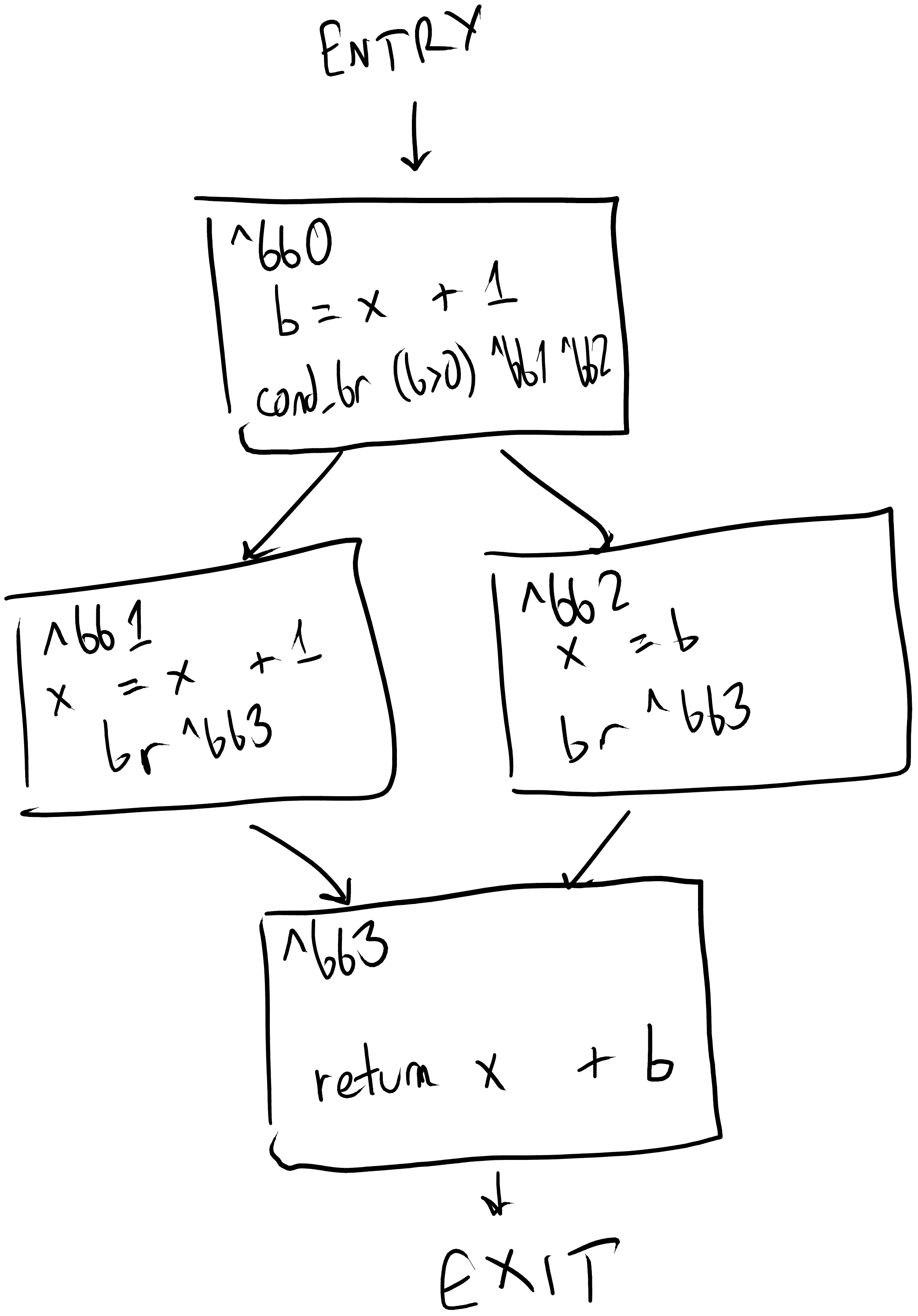
```
fn f(x: int) {
  b = x + 1;
  if b > 0 {
    x = x + 1;
  } else {
    x = b;
  }
  return x + b;
}
```



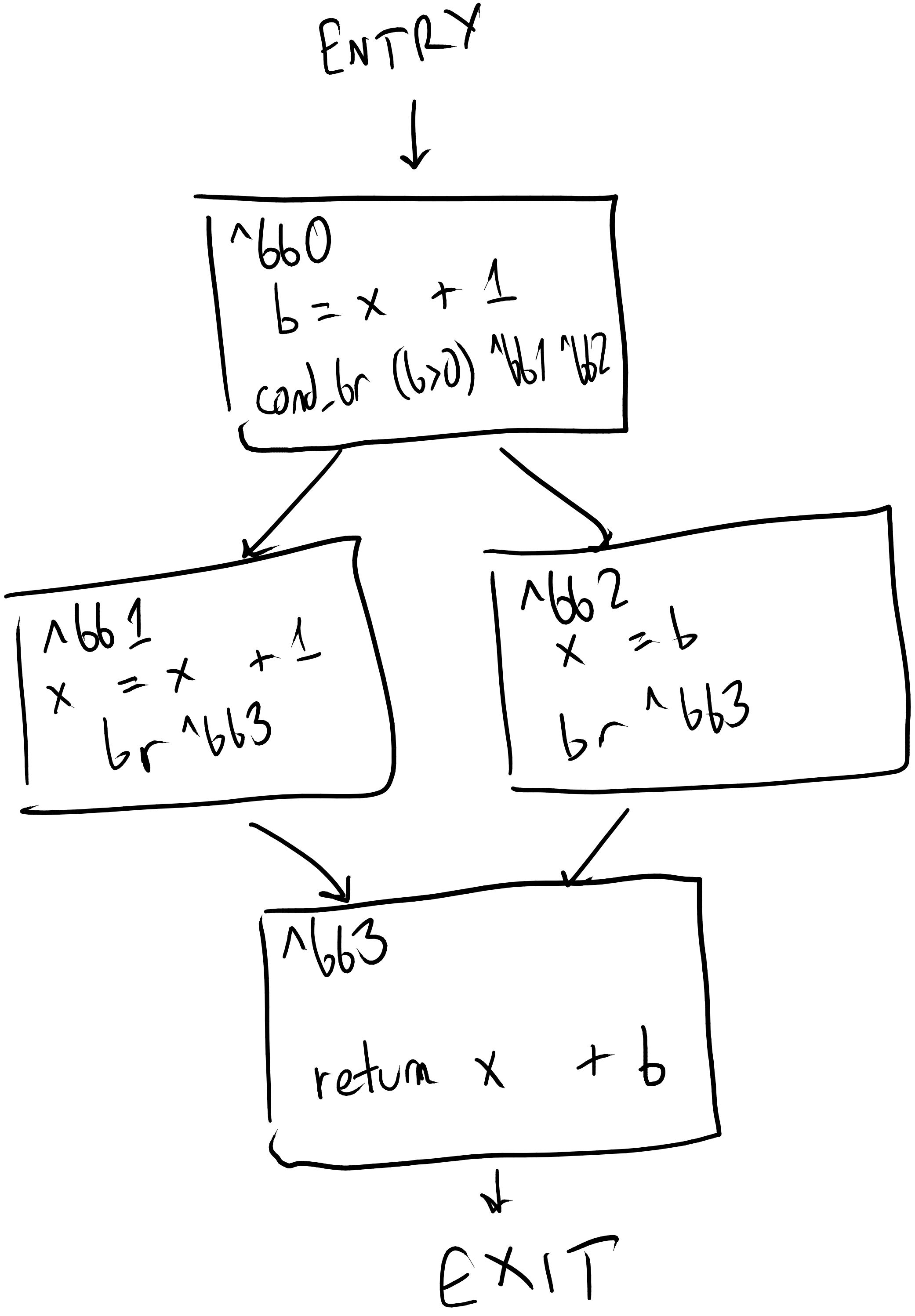
```
fn f(x0: int) {
  b = x0 + 1;
  if b > 0 {
    x1 = x0 + 1;
  } else {
    x2 = b;
  }
  return x? + b;
}
```



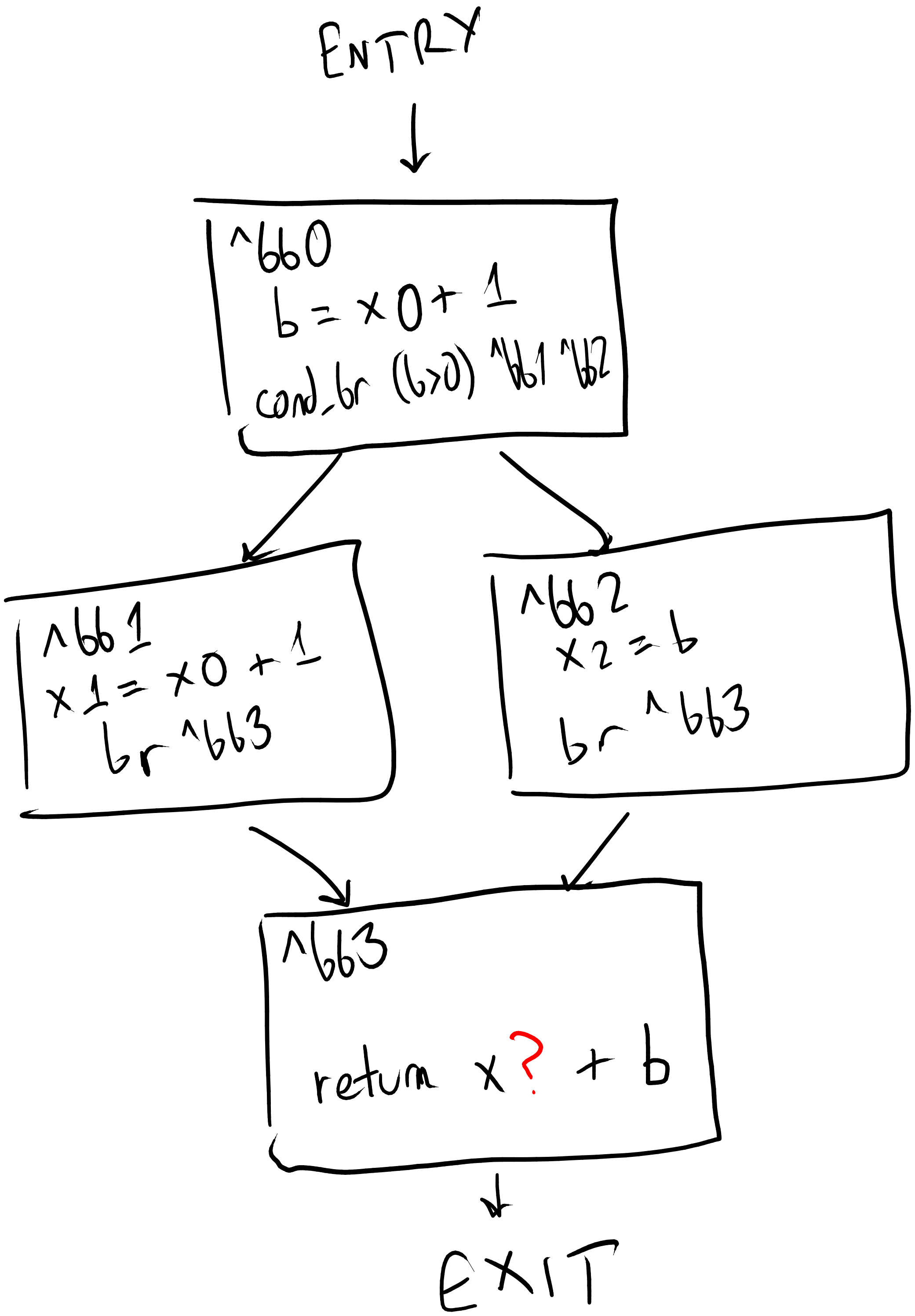
```
fn f(x: int) {
  b = x + 1;
  if b > 0 {
    x = x + 1;
  } else {
    x = b;
  }
  return x + b;
}
```



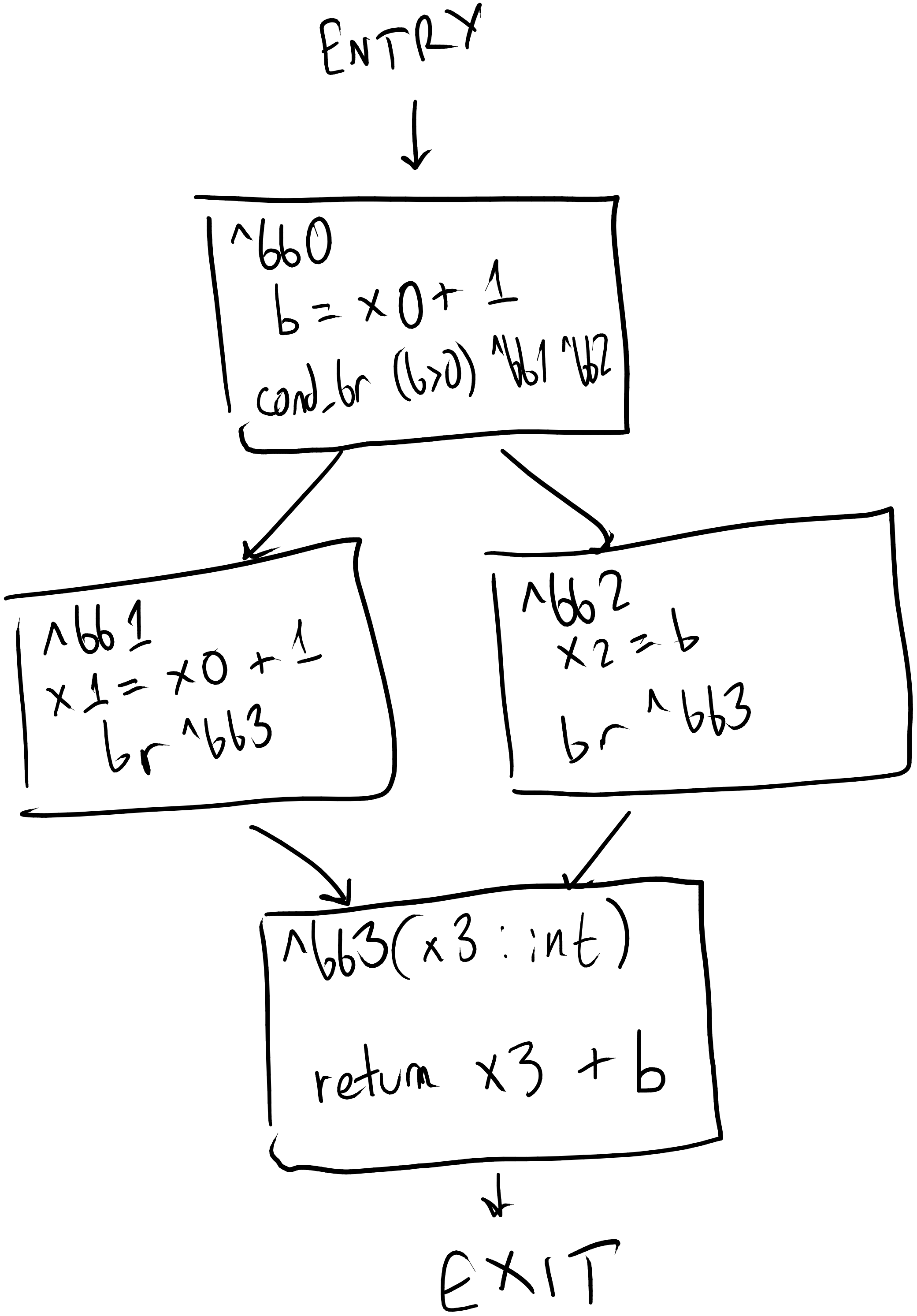
```
fn f(x0: int) {
^bb0:
  b = x + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x = x + 1;
  br ^bb3
^bb2:
  x = b;
  br ^bb3
^bb3:
  return x + b;
}
```



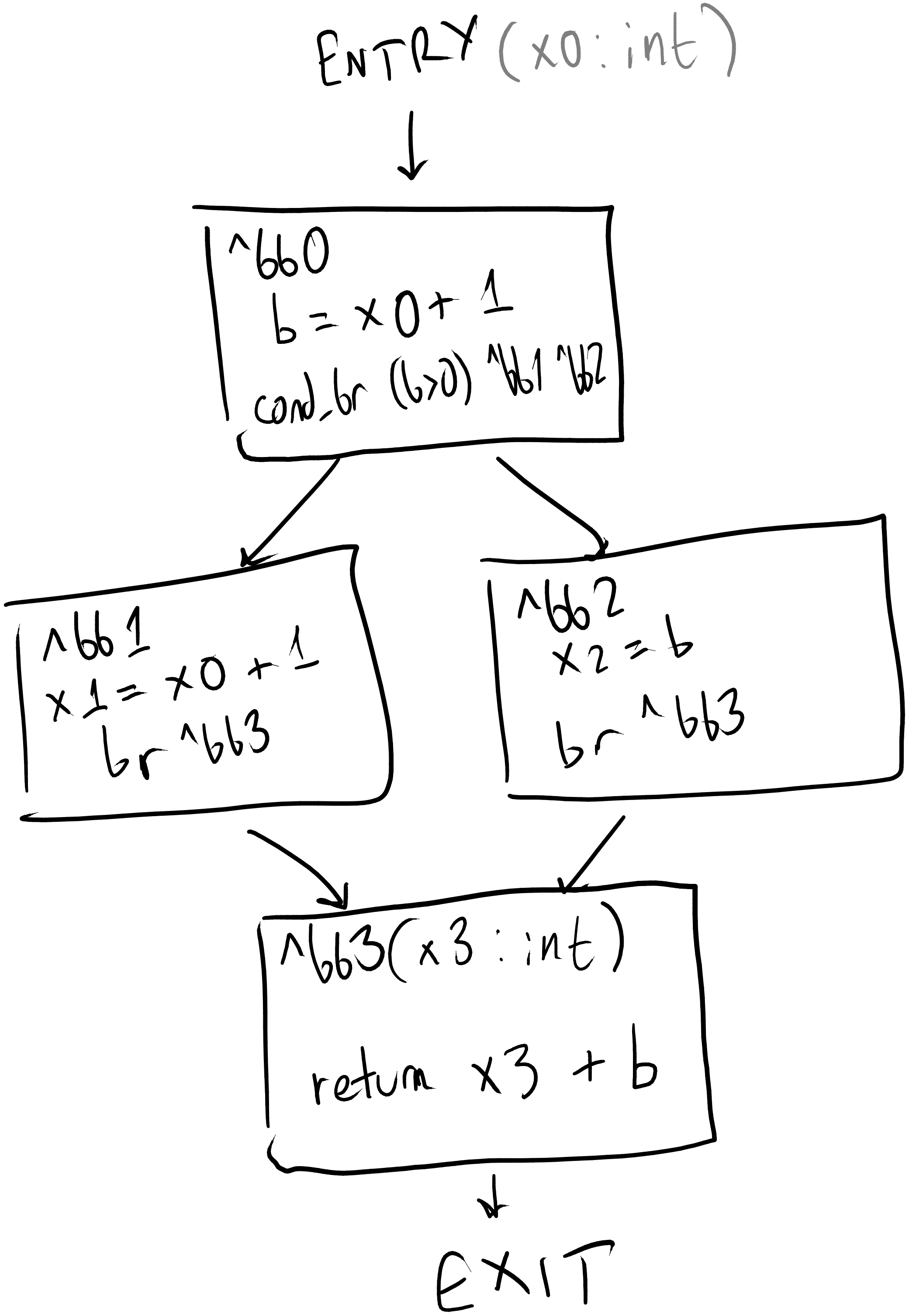
```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3:
  return x?+ b;
}
```



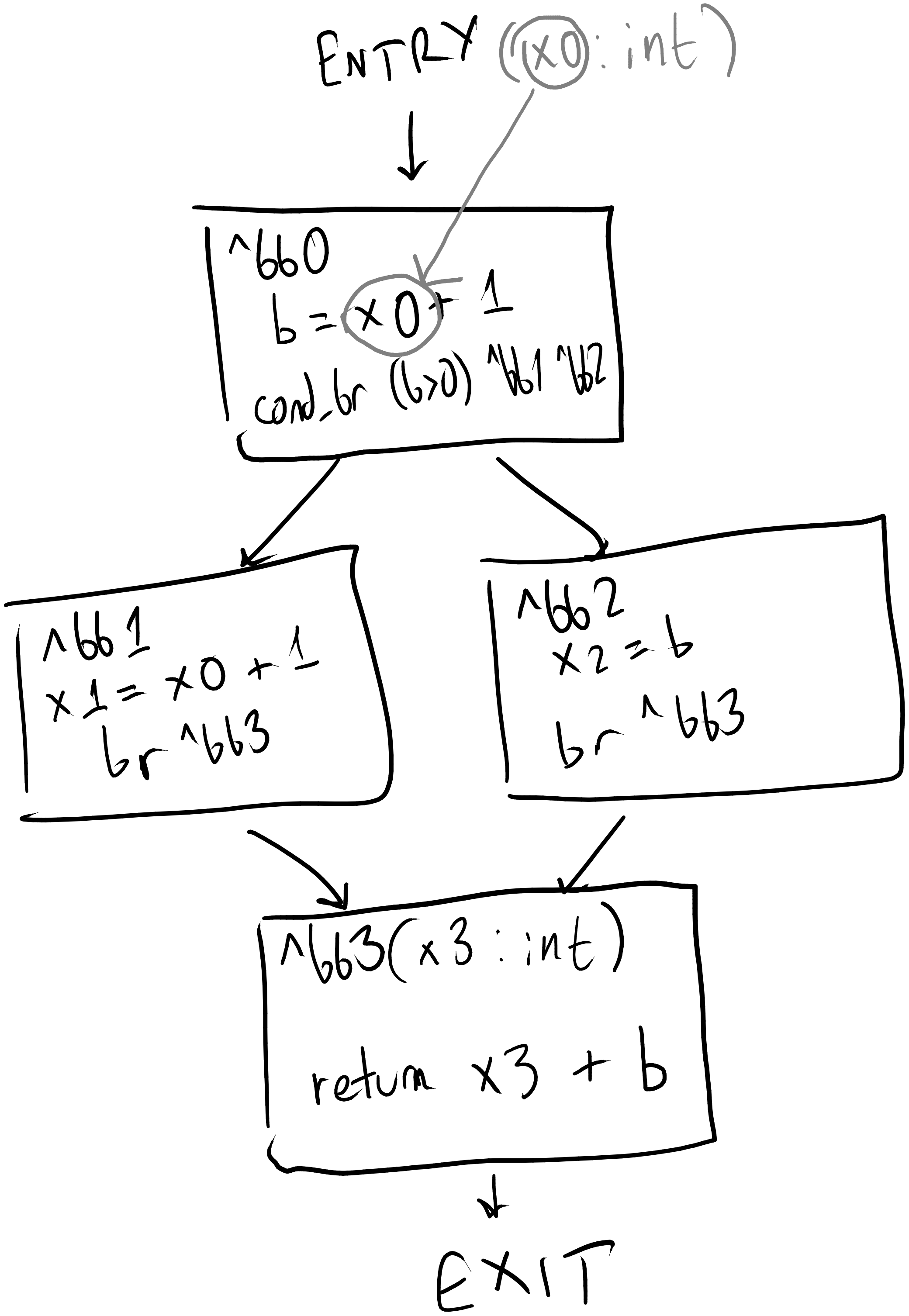
```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



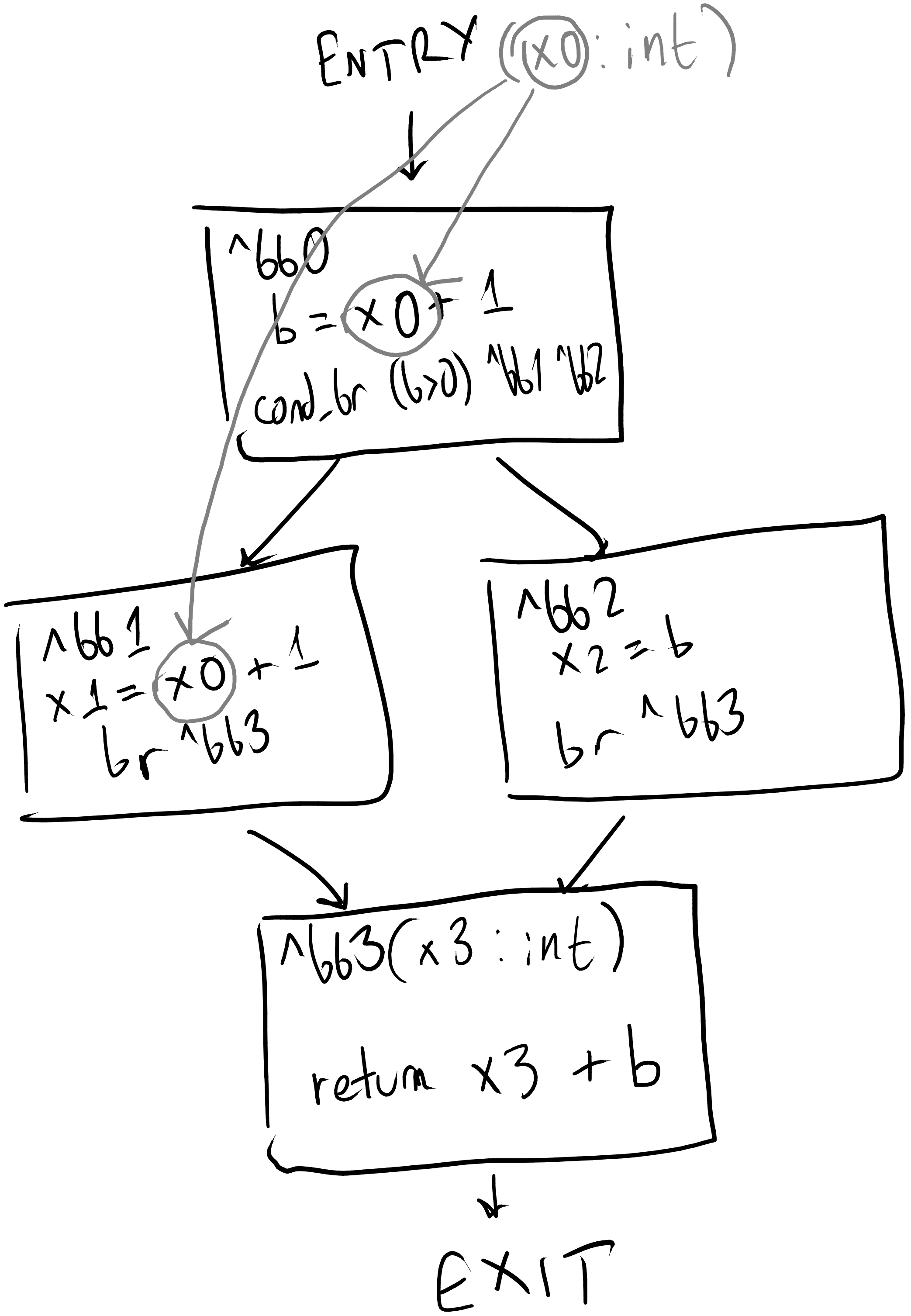
```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



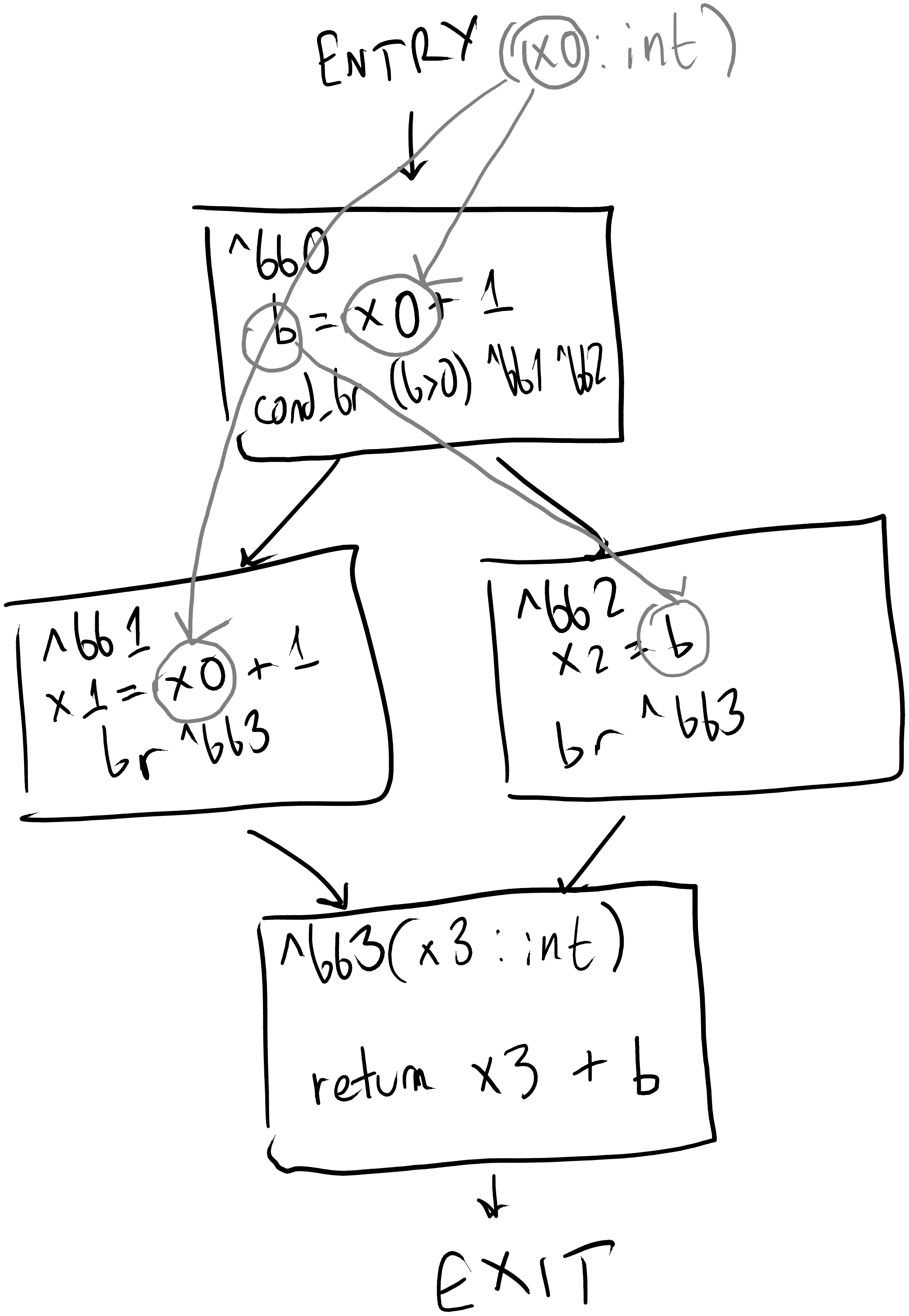
```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



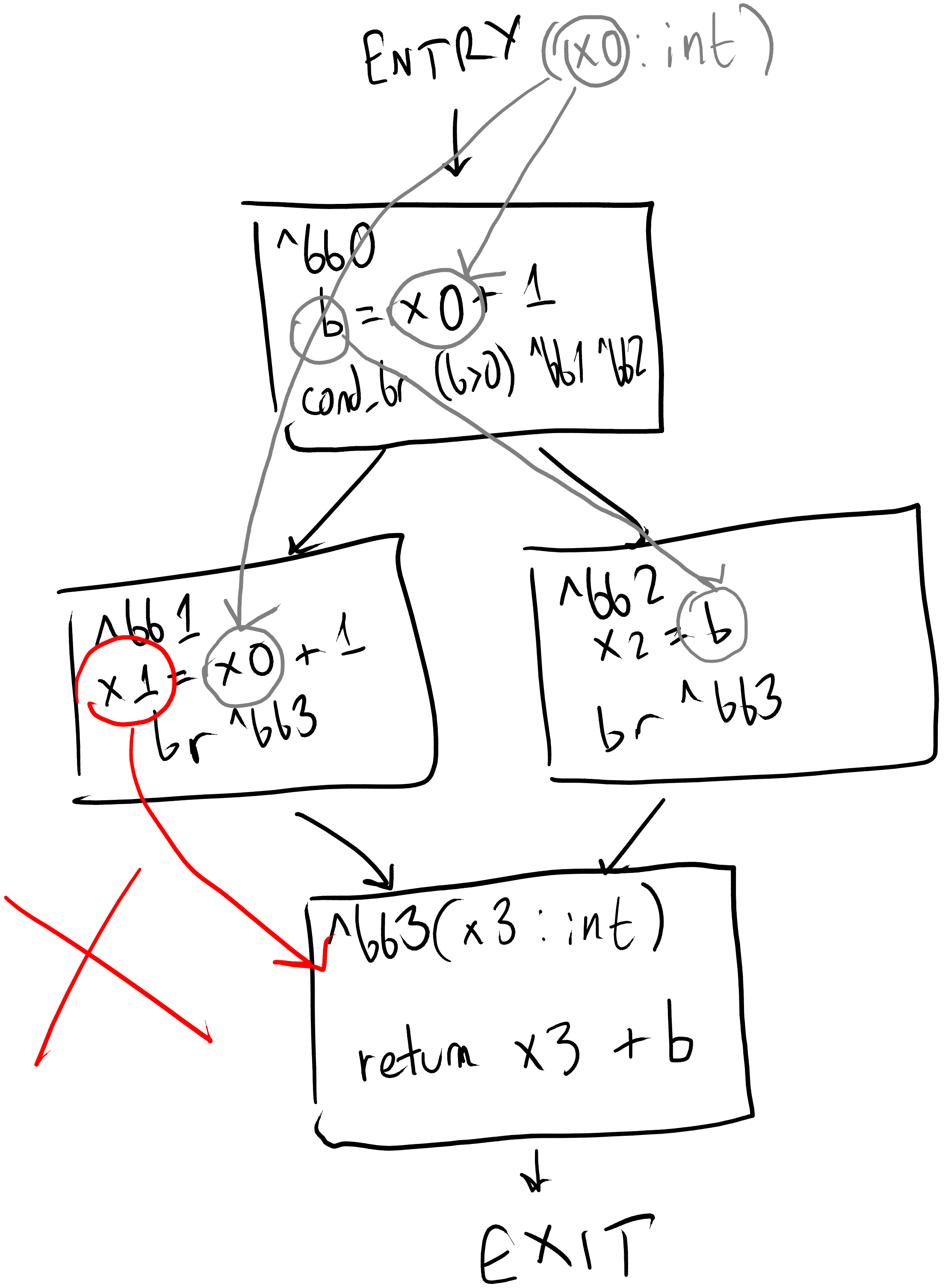
```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



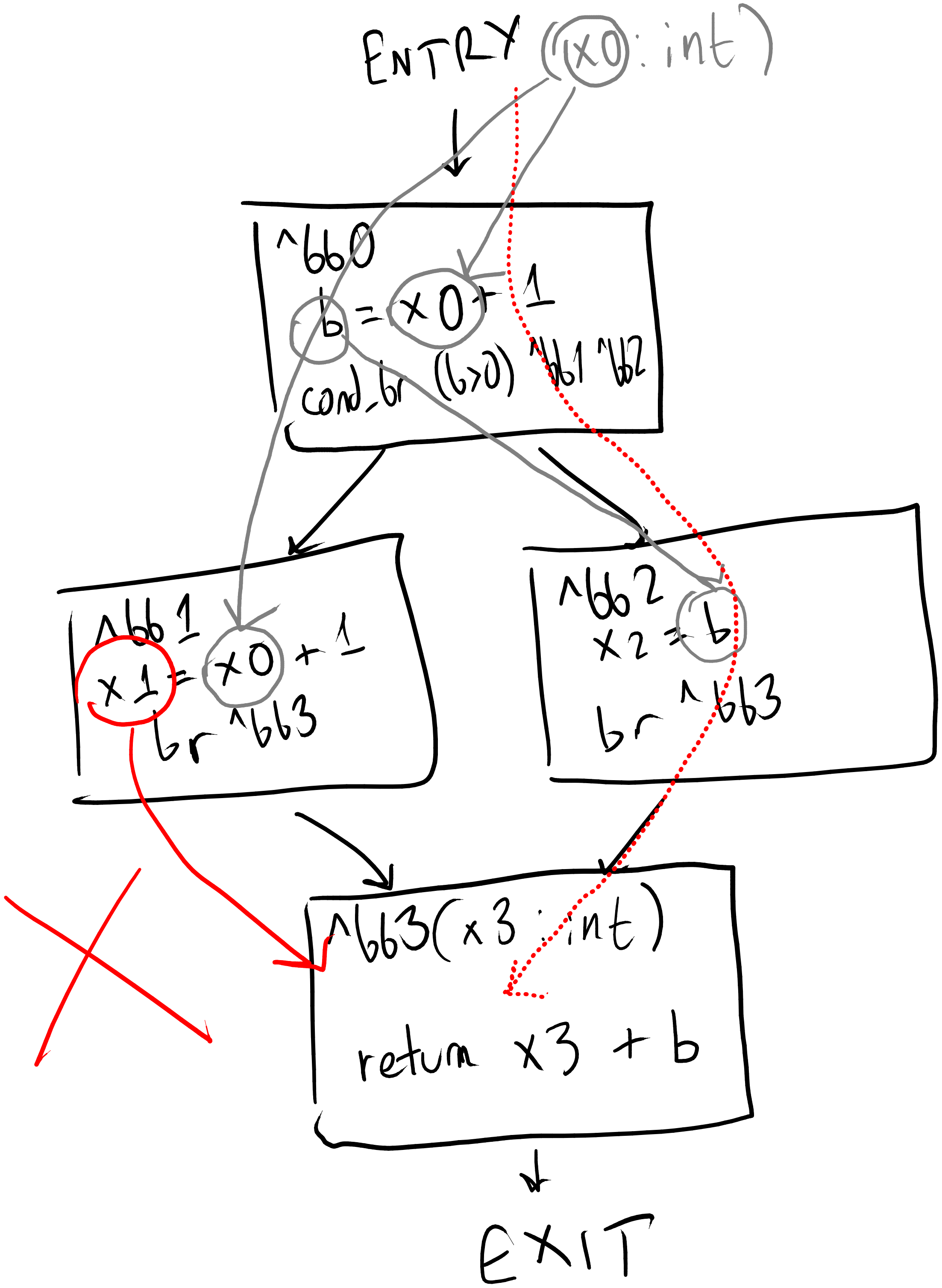
```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



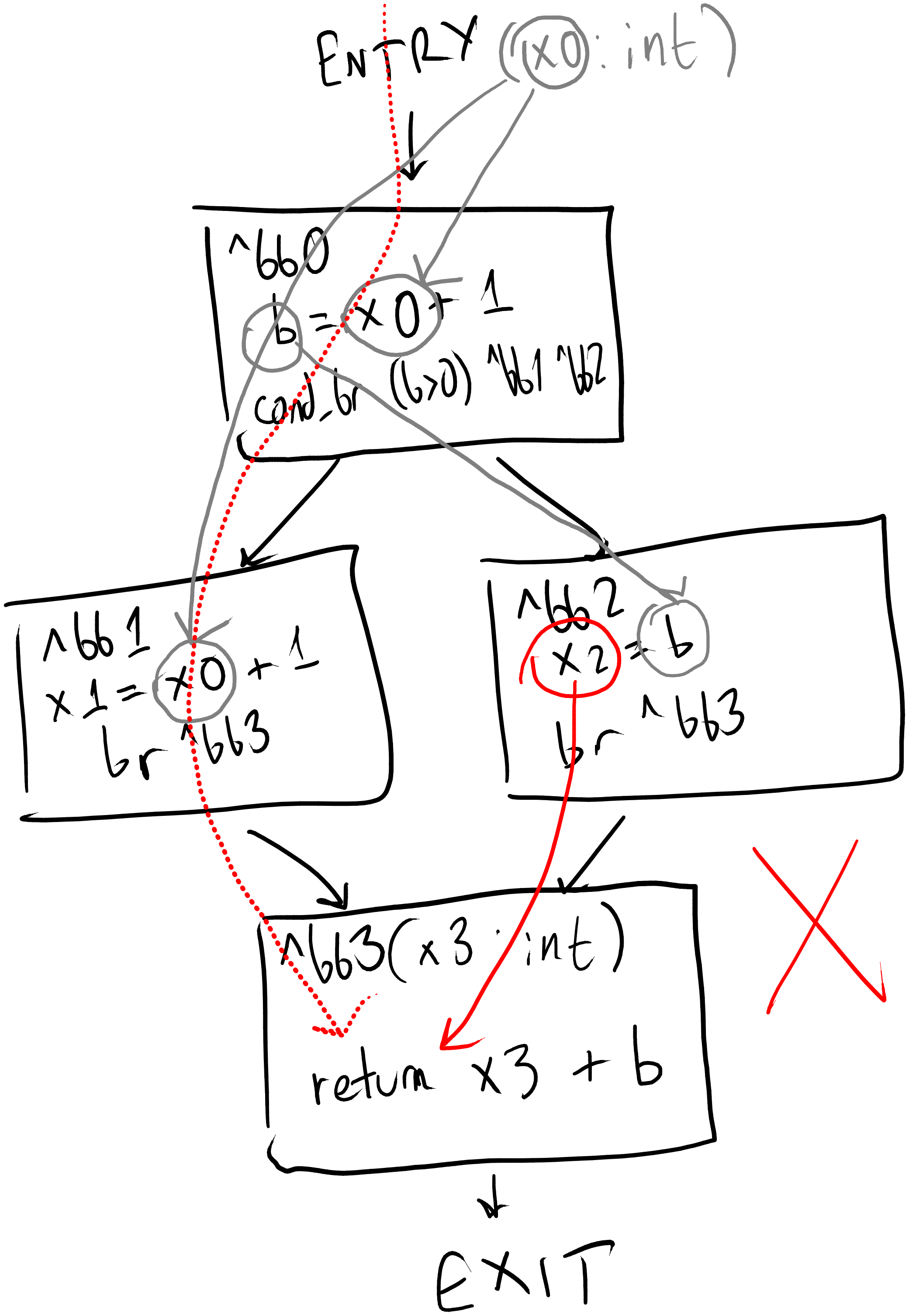
```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



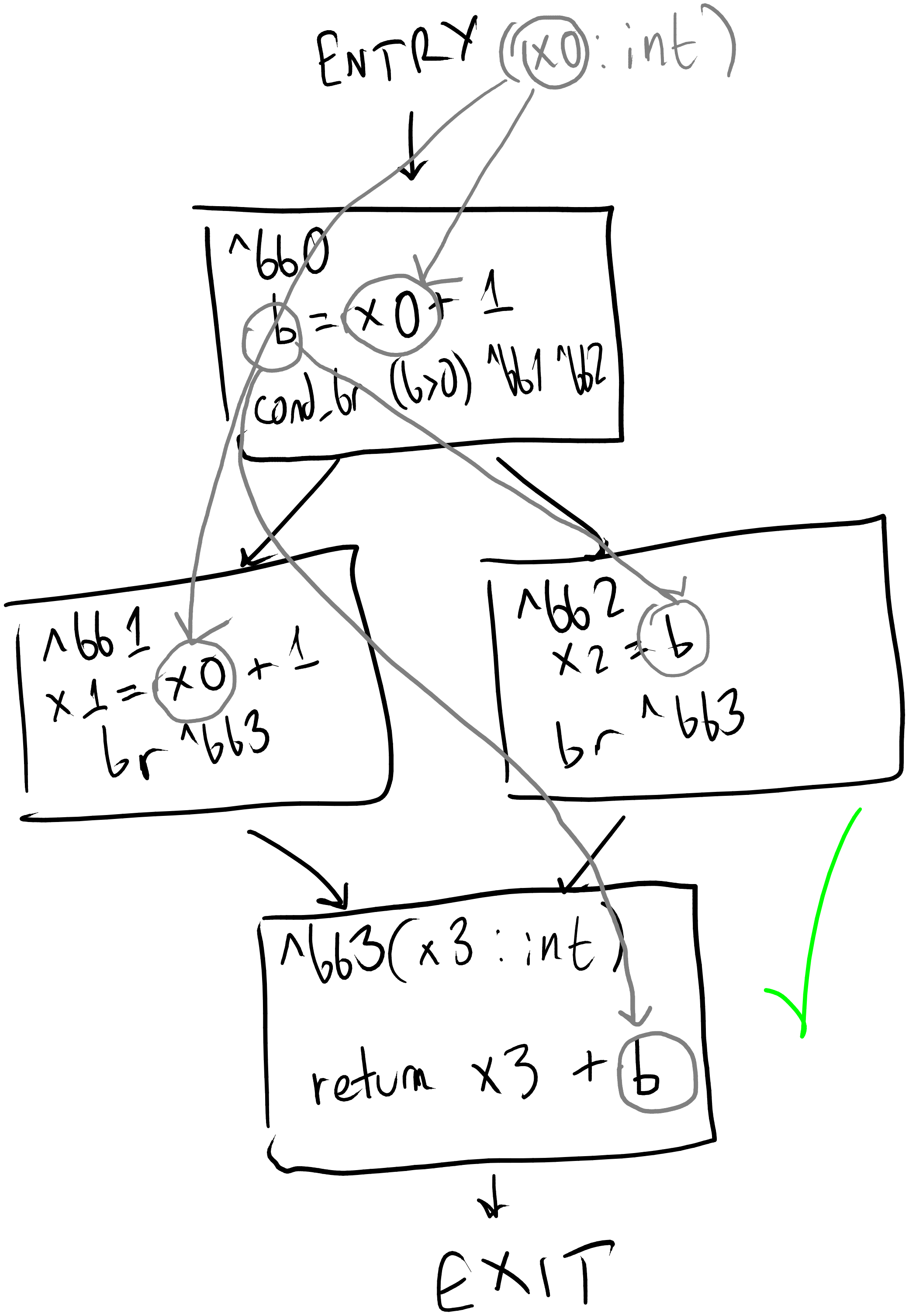
```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



```
fn f(x0: int) {
^bb0:
  b = x0 + 1;
  cond_br (b > 0) ^bb1 ^bb2
^bb1:
  x1 = x0 + 1;
  br ^bb3
^bb2:
  x2 = b;
  br ^bb3
^bb3(x3: int):
  return x3 + b;
}
```



Part II: Semantics of SSA

SSA Recap

SSA Recap

Instructions

```
x = a + b
y = call f x
```

SSA Recap

Instructions

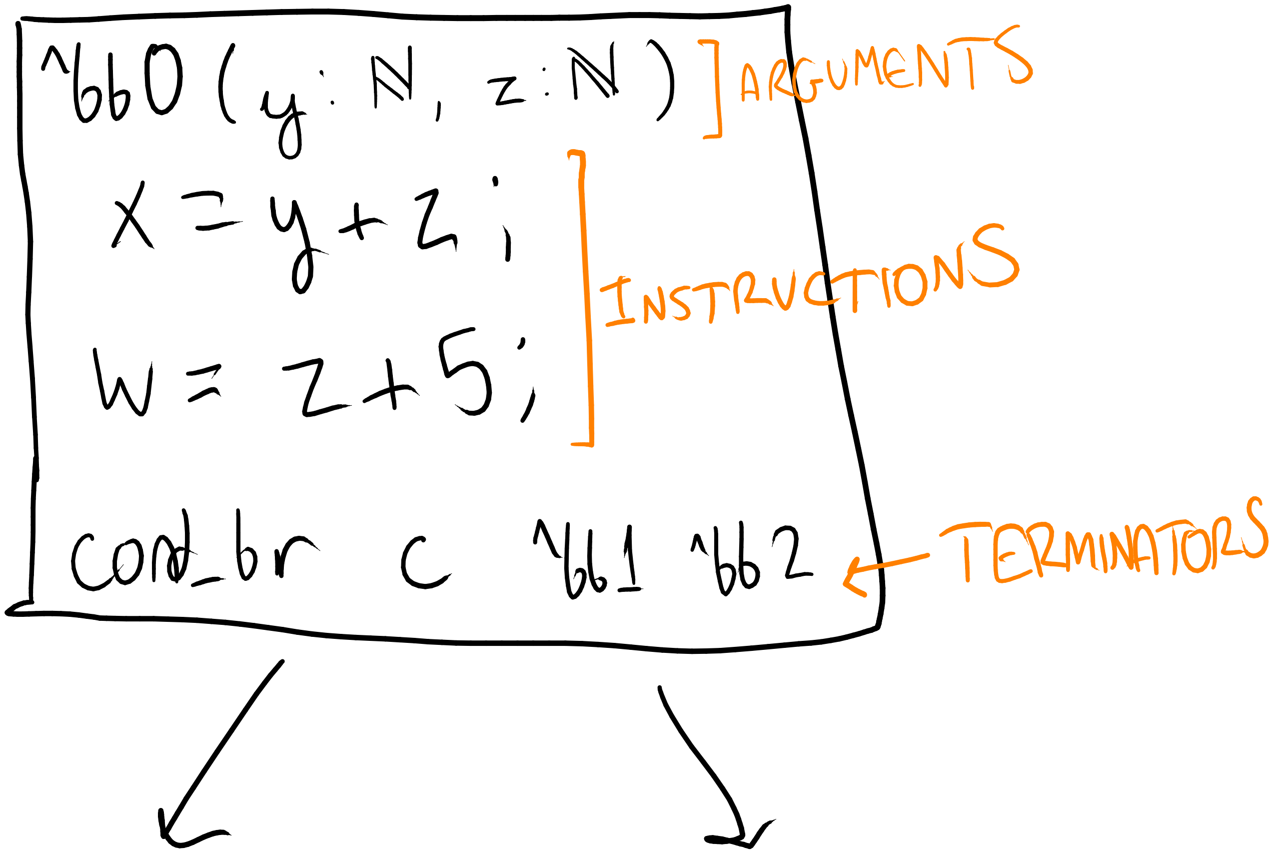
```
x = a + b
y = call f x
```

↳ Terminators:

```
return x
br ^l(y)
```

SSA Recap

Basic Block

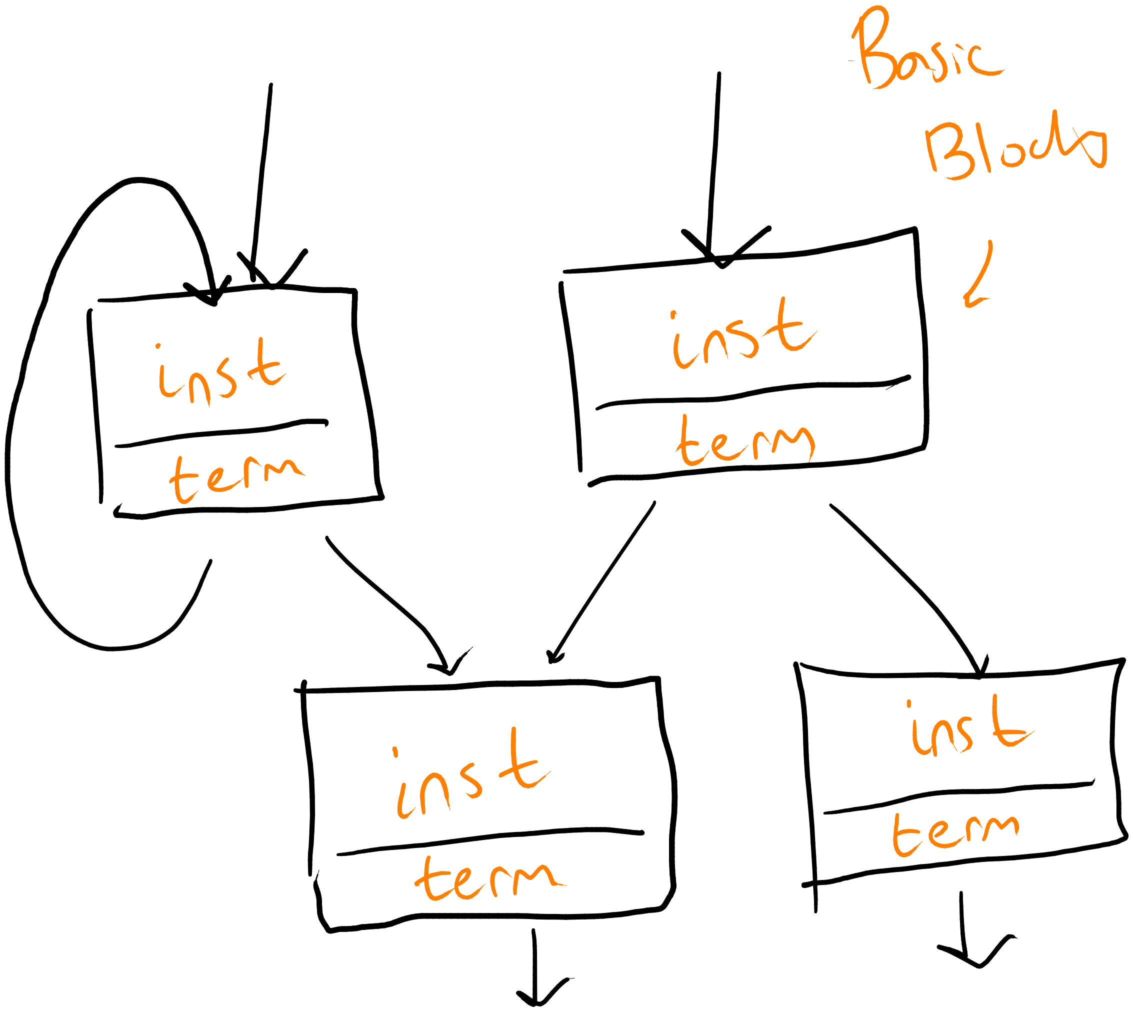


```
^bb0(y: ℕ, z: ℕ)
  x = y + z;
  w = z + 5;

cond_br c ^bb1 ^bb2
```

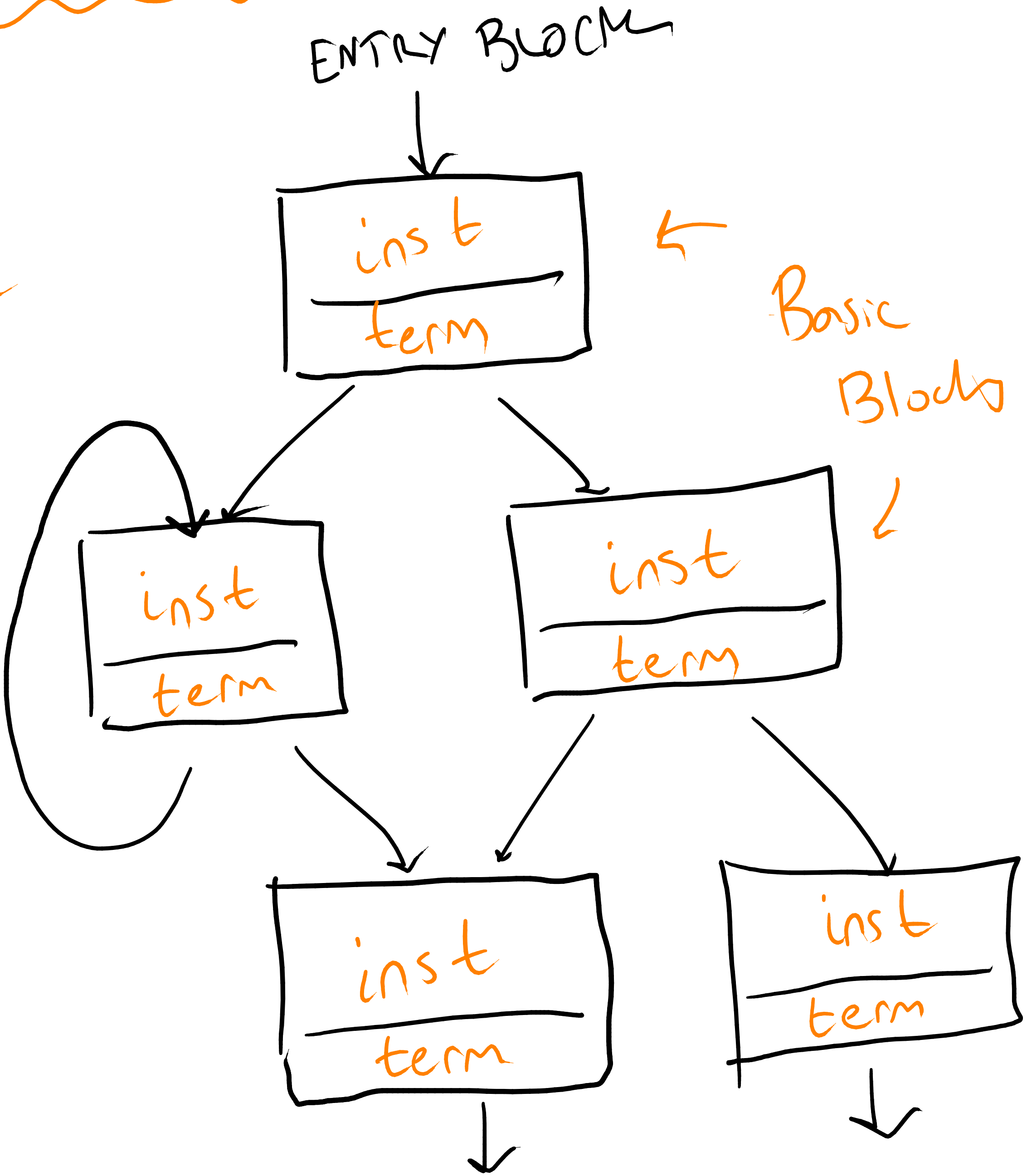
SSA Recap

CFGs

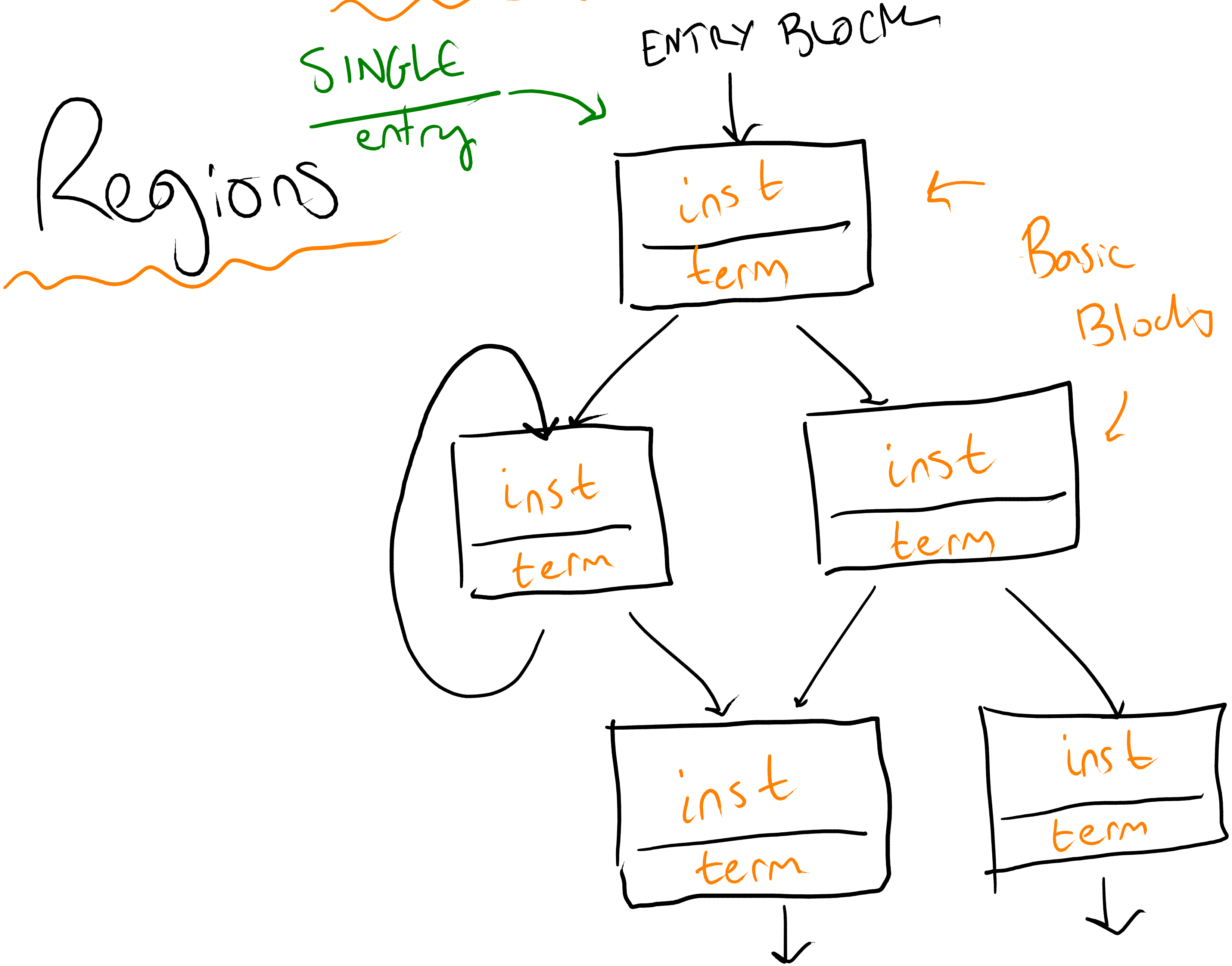


SSA Recap

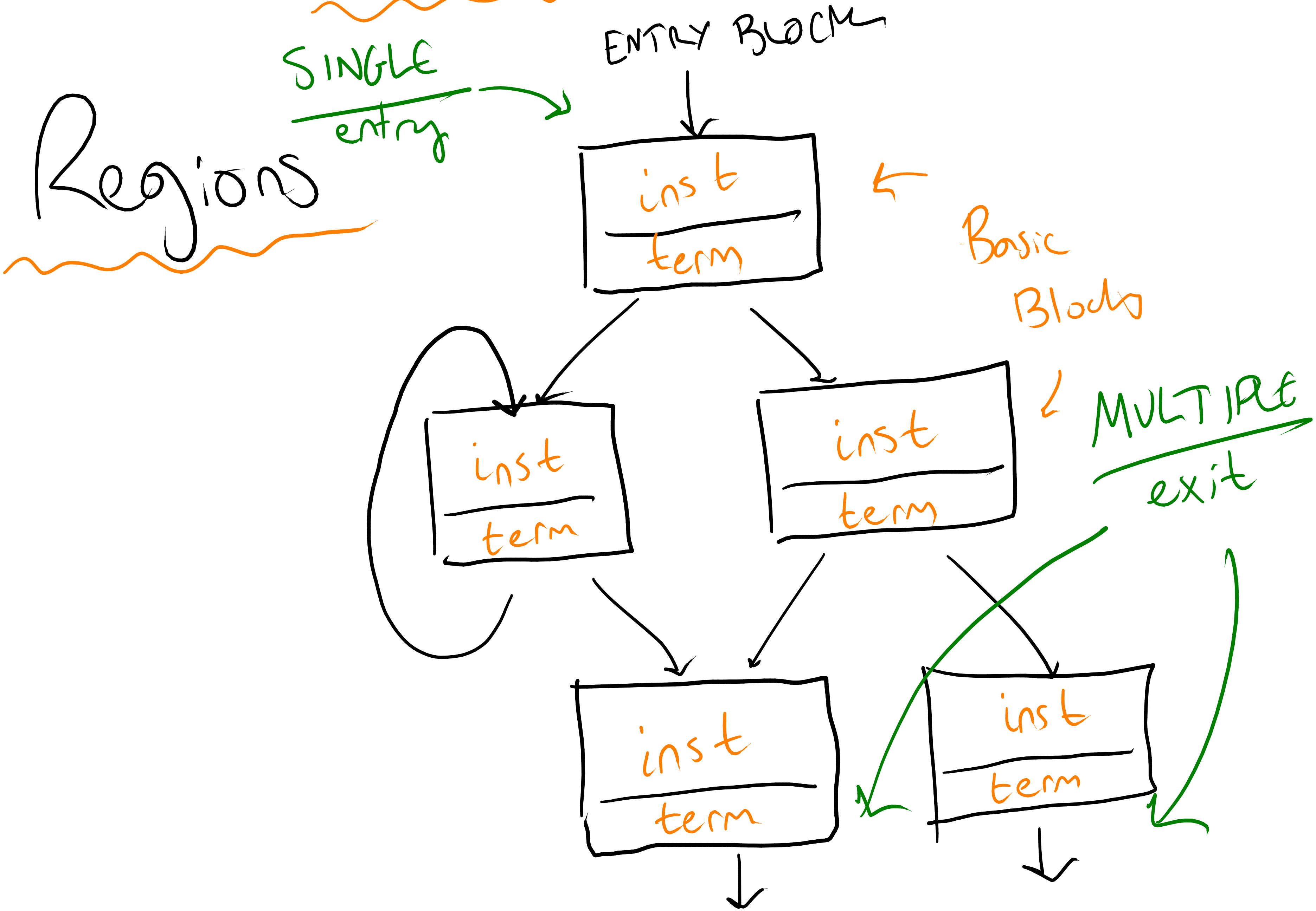
Regions



SSA Recap


ENTRY BLOCK
SINGLE
entry
Regions
Basic
Block
inst
term
inst
term
inst
term
inst
term
inst
term

SSA Recap


Regions
SINGLE entry
ENTRY BLOCK
inst
term
Basic Blocks
MULTIPLE exit
inst
term
inst
term
inst
term
inst
term

Applications of SSA

Applications of SSA

- Classical compilers

Applications of SSA

- Classical compilers
  - x86, ARM, ...

Applications of SSA

- Classical compilers
  - └ x86, ARM, ...
- Accelerators

Applications of SSA

- Classical compilers
  - x86, ARM, ...
- Accelerators
  - GPU, FPGA, systolic array...

Applications of SSA

- Classical compilers
  - x86, ARM, ...

- Accelerators
  - GPU, FPGA, systolic array...

- High-level IRs

Applications of SSA

- Classical compilers
  - x86, ARM, ...
- Accelerators
  - GPU, FPGA, systolic array...
- High-level IRs
  - 'async-await', tensors

Applications of SSA

- Classical compilers
  - x86, ARM, ...
- Accelerators
  - GPU, FPGA, systolic array...
- High-level IRs
  - 'async-await', tensors

- Hardware
- Quantum...

WANT: Semantics

WANT: Semantics

Issues:

- LOTS of models!

WANT: Semantics

Issues:

- LOTS of models!
- Compositionality!

WANT: Semantics

Issues:

- LOTS of models!
- Compositionality!
- Graphical Intuition!

Abstract + Compositional

= Categories

Categorical Semantics

Categorical Semantics

$$\Gamma \vdash a : A$$

Categorical Semantics

$$\Gamma \vdash \boxed{\overset{\text{TERM}}{a}} : A$$

Categorical Semantics

$$\Gamma \vdash \boxed{\overset{\text{TERM}}{a}} : \boxed{\overset{\text{TYPE}}{A}}$$

Categorical Semantics

$$\overset{\text{CTX}}{\boxed{\Gamma}} \vdash \overset{\text{TERM}}{\boxed{a}} : \overset{\text{TYPE}}{\boxed{A}}$$

Categorical Semantics

$$[\![\Gamma \vdash a : A]\!] : \mathcal{C}([\![\Gamma]\!], [\![A]\!])$$

Categorical Semantics

$$[\![\Gamma \vdash a : A]\!]$$

$$: \mathcal{C}([\![\Gamma]\!], [\![A]\!])$$

Contexts + types interpreted as OBJECTS

Categorical Semantics

$$[\![\Gamma \vdash a : A]\!]$$

$$: \mathcal{C}([\![\Gamma]\!], [\![A]\!])$$

Compositionality

Plan: - give inductive grammar for instructions, programs

- give typing rules

- give cat. semantics for well-typed programs

Compositionality

$$[\![\Gamma \vdash f\ a : B]\!]$$

Compositionality

$$[\![\Gamma \vdash f\ a : B]\!]$$

where $f \in \mathrm{inst}(A, B)$

Compositionality

$$[\![\Gamma \vdash f\ a : B]\!] = [\![f]\!] \circ [\![\Gamma \vdash a : A]\!]$$

where $f \in inst(A,B)$

Compositionality

$$[\![\Gamma \vdash f\ a : B]\!] = [\![f]\!] \circ [\![\Gamma \vdash a : A]\!]$$

where $f \in \mathrm{inst}(A, B)$

$\uparrow$

NOTE:

Doesn't depend on context!

Compositionality

$$[\![\Gamma \vdash f\ a : B]\!] = [\![\Gamma \vdash a : A]\!] ; [\![f]\!]$$

where $f \in \mathrm{inst}(A,B)$

Compositionality

$$[\![\Gamma \vdash f\ a : B]\!] = [\![\Gamma \vdash a : A]\!] ; [\![f]\!]$$

where $f \in \mathrm{inst}(A,B)$

↑

Do this

Compositionality

$$[\![\Gamma \vdash f\ a : B]\!] = [\![\Gamma \vdash a : A]\!] ; [\![f]\!]$$

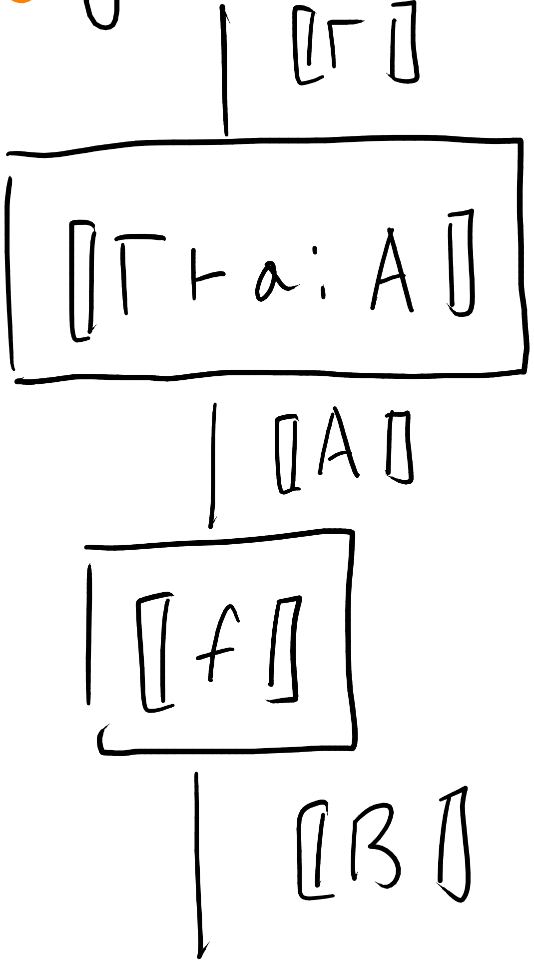
where $f \in \mathrm{inst}(A,B)$

↑ Do this

↑ THEN,
w/ the output,
do this.

Compositionality

$$[\![\Gamma \vdash f\ a : B]\!] =$$



where $f \in \mathrm{inst}(A,B)$

Cartesian Product

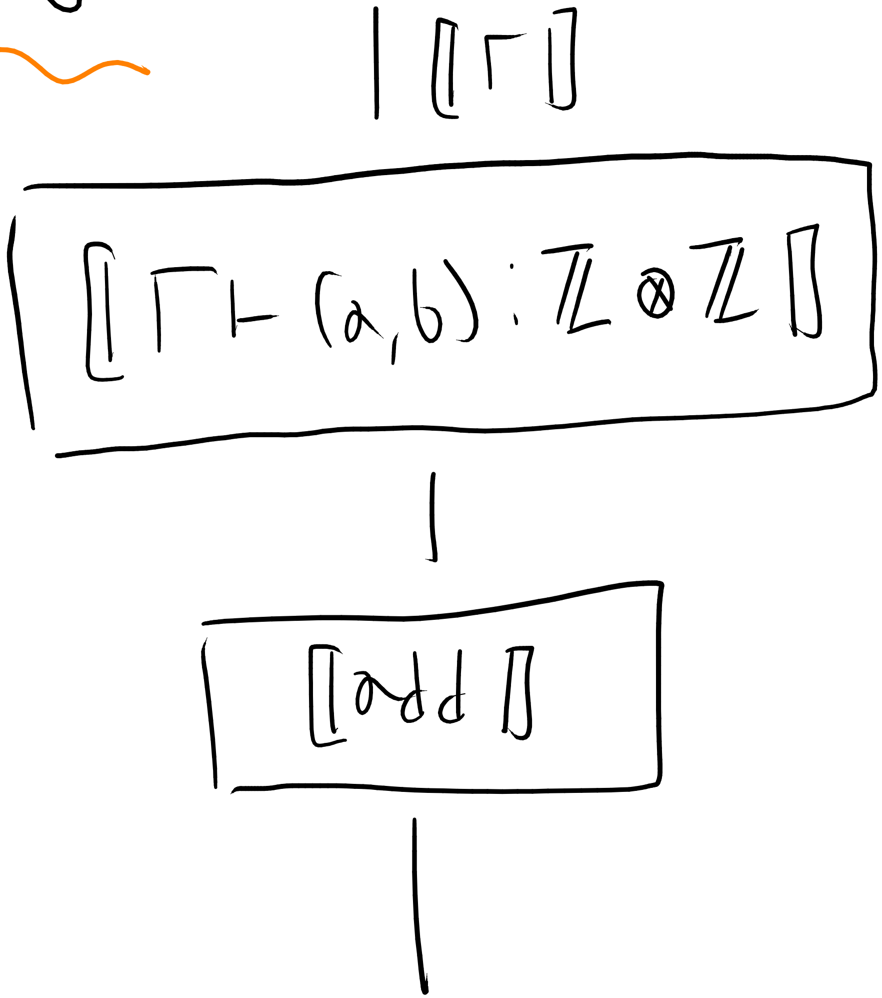
$$[\![\Gamma \vdash \mathrm{add}\ a\ b : \mathbb{Z}]\!]$$

Cartesian Products

$$[\![\Gamma \vdash \mathrm{add}(a, b) : \mathbb{Z}]\!]$$

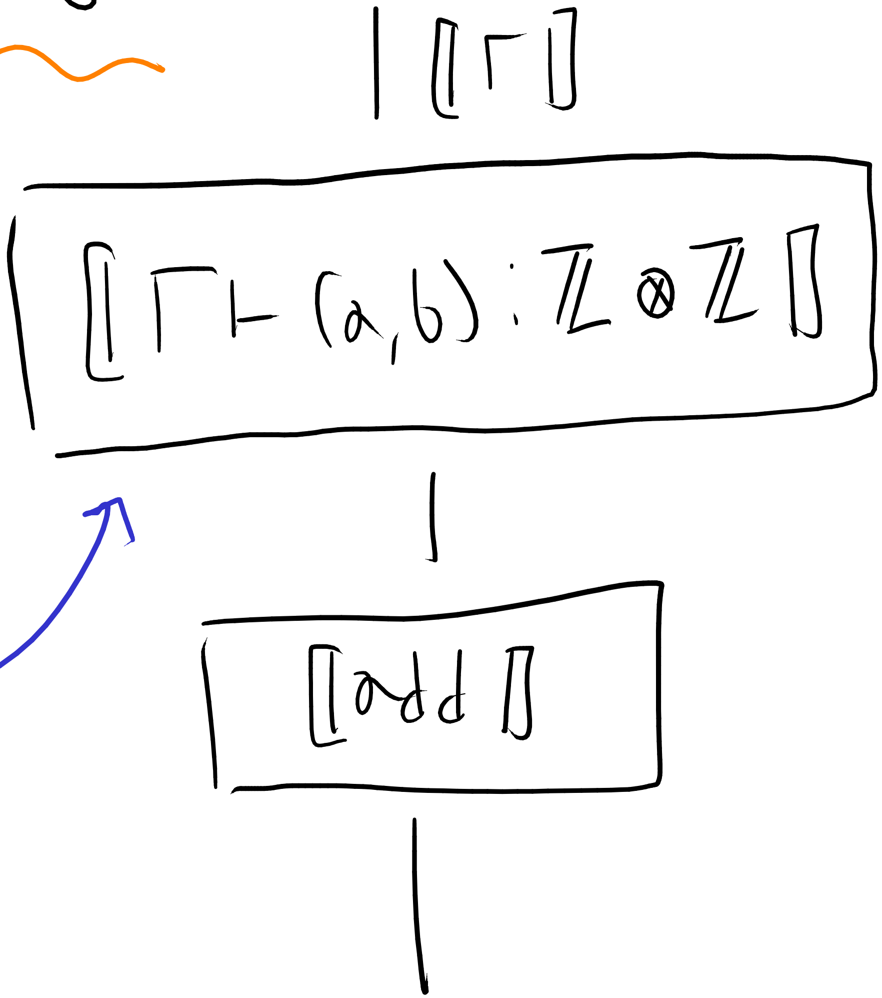
Cartesian Product

$$[\![\Gamma \vdash \mathrm{add}(a,b) : \mathbb{Z}]\!] =$$



Cartesian Products

$$[\![\Gamma \vdash \mathrm{add}\,(a,b) : \mathbb{Z}]\!] =$$



?

Cartesian Products

Given $f : A \to B$, $g : A \to C$,
Want $\langle f, g \rangle$ unique s.t.

$$\begin{tikzcd}
 & A \arrow[ddl, bend right, "f"'] \arrow[d, "{\langle f, g \rangle}"] \arrow[ddr, bend left, "g"] & \\
 & B \otimes C \arrow[dl, "\pi_1"'] \arrow[dr, "\pi_2"] & \\
B & & C
\end{tikzcd}$$

Cartesian Products

Given $f : A \to B$, $g : A \to C$,

Want $\langle f,g\rangle$ unique s.t.

$$\begin{tikzcd}
 & A \arrow[ddl, bend right, "f"'] \arrow[d, "\langle f{,}g\rangle"] \arrow[ddr, bend left, "g"] & \\
 & B \otimes C \arrow[dl, "\pi_1"'] \arrow[dr, "\pi_2"] & \\
B & & C
\end{tikzcd}$$

Note:

$\pi_1 : A \otimes B \to A$

$\pi_2 : A \otimes B \to B$

Cartesian Products

Given $f : A \to B$, $g : C \to D$,

can define $f \times g = \langle \pi_1 ; f , \pi_2 ; g \rangle : A \otimes C \to B \otimes D$

Not $\otimes$, will get to this later...

Cartesian Products

Given $f: A \to B$, $g: C \to D$,

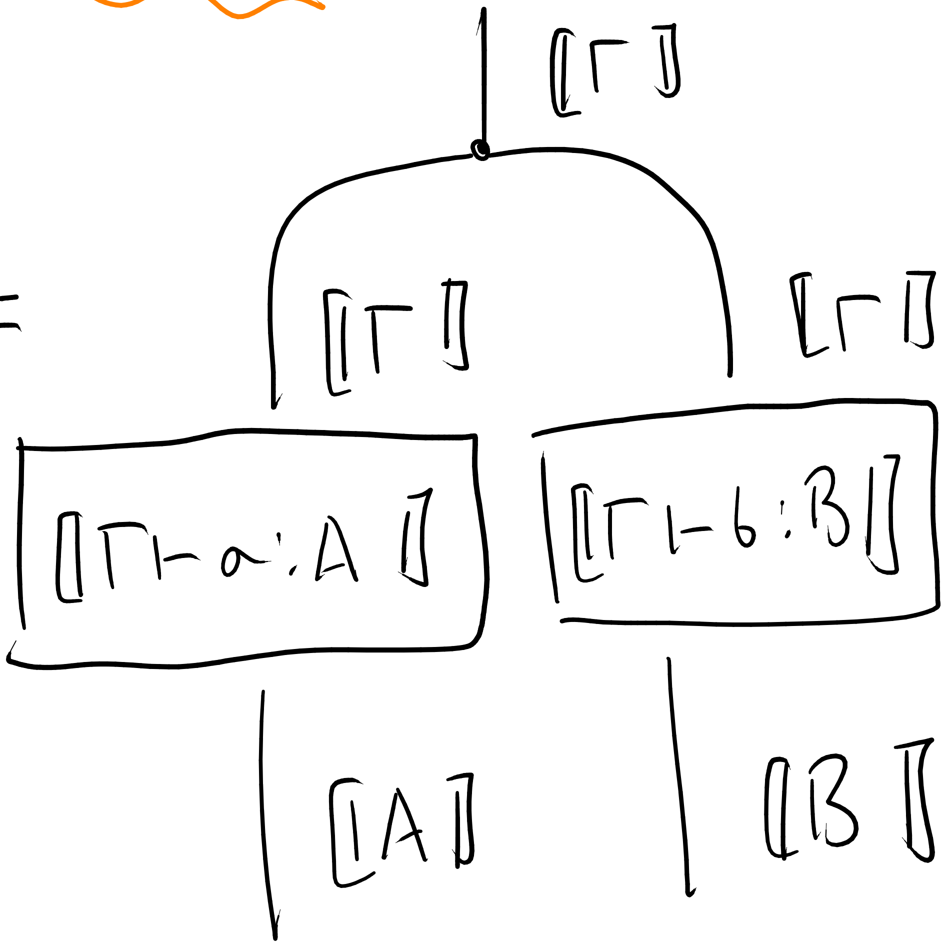
can define $f \times g = \langle \pi_1 ; f, \pi_2 ; g \rangle : A \otimes C \to B \otimes D$

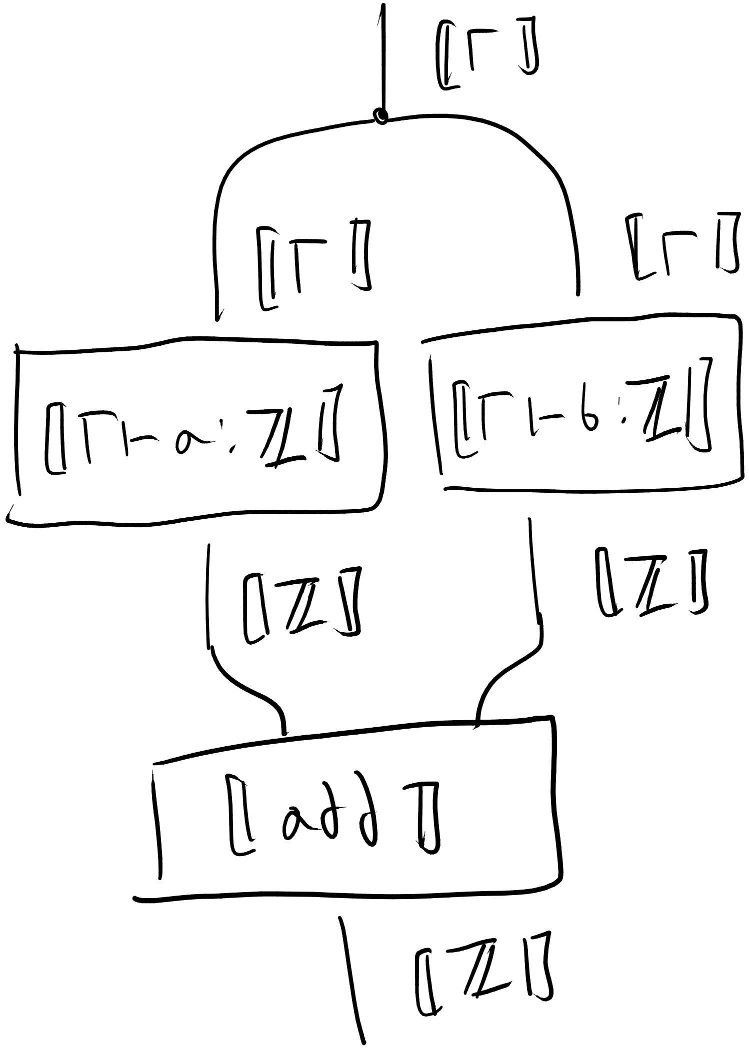
Cartesion Prodcty

$$\llbracket \Gamma \vdash (a, b) : A \otimes B \rrbracket = \langle \llbracket \Gamma \vdash a : A \rrbracket , \llbracket \Gamma \vdash b : B \rrbracket \rangle$$

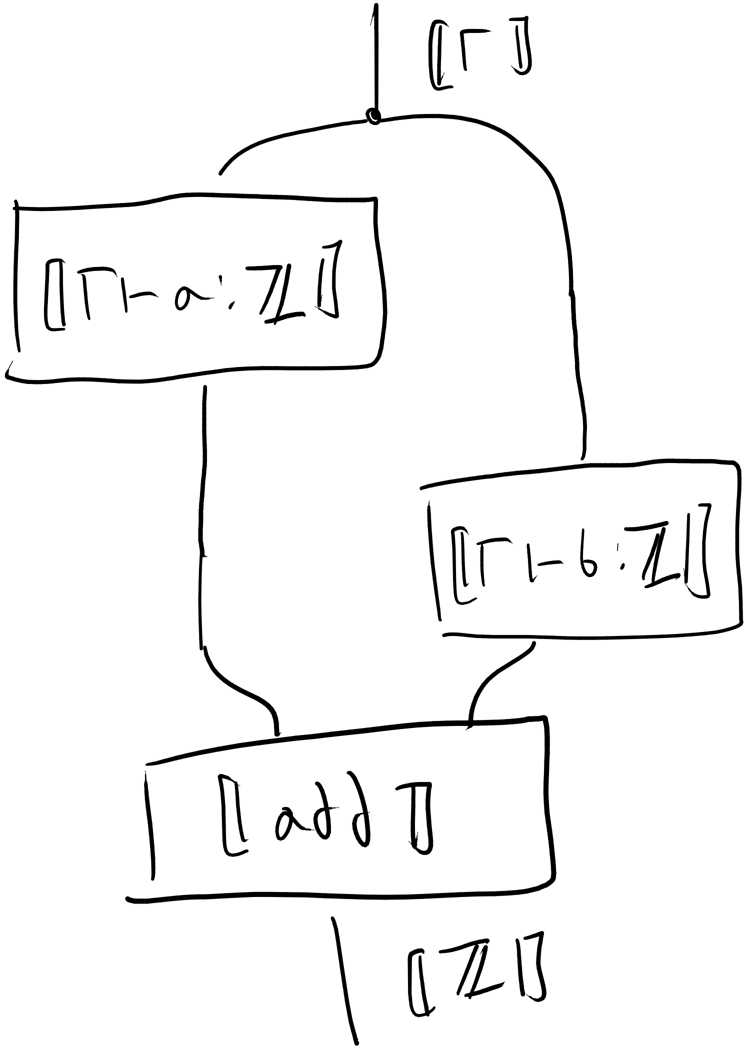
Cartesian Product

$$[\![\Gamma \vdash (a,b) : A \otimes B]\!] =$$

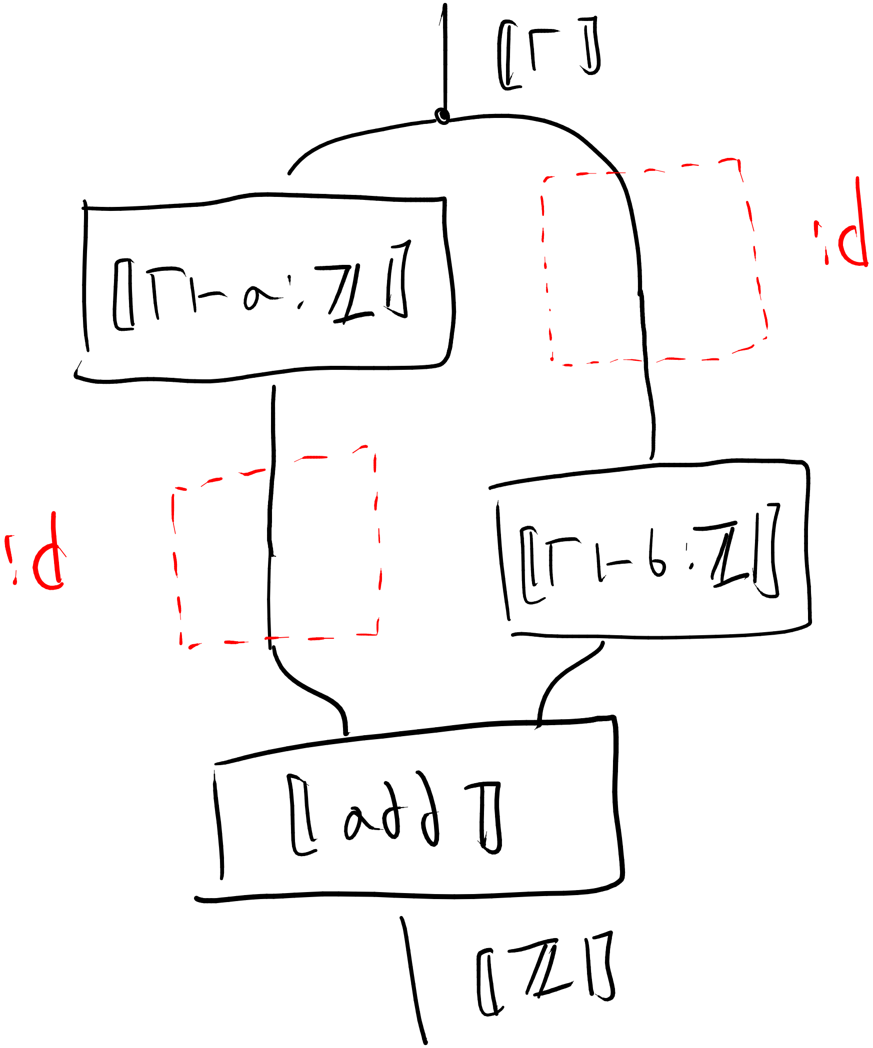


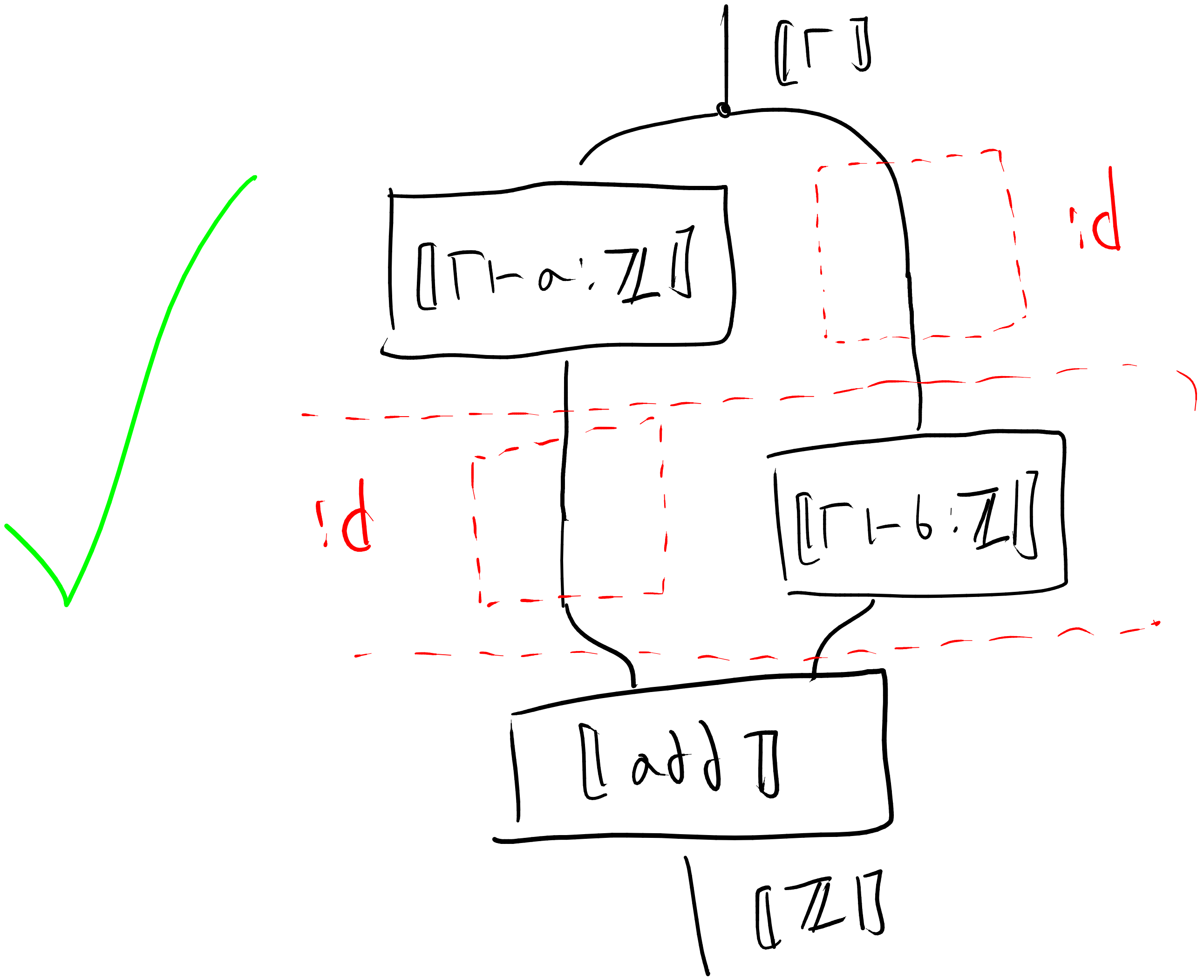

⟦Γ⟧
⟦Γ⟧
⟦Γ⟧
⟦Γ ⊢ a : ℤ⟧
⟦Γ ⊢ b : ℤ⟧
⟦ℤ⟧
⟦ℤ⟧
⟦add⟧
⟦ℤ⟧

✓

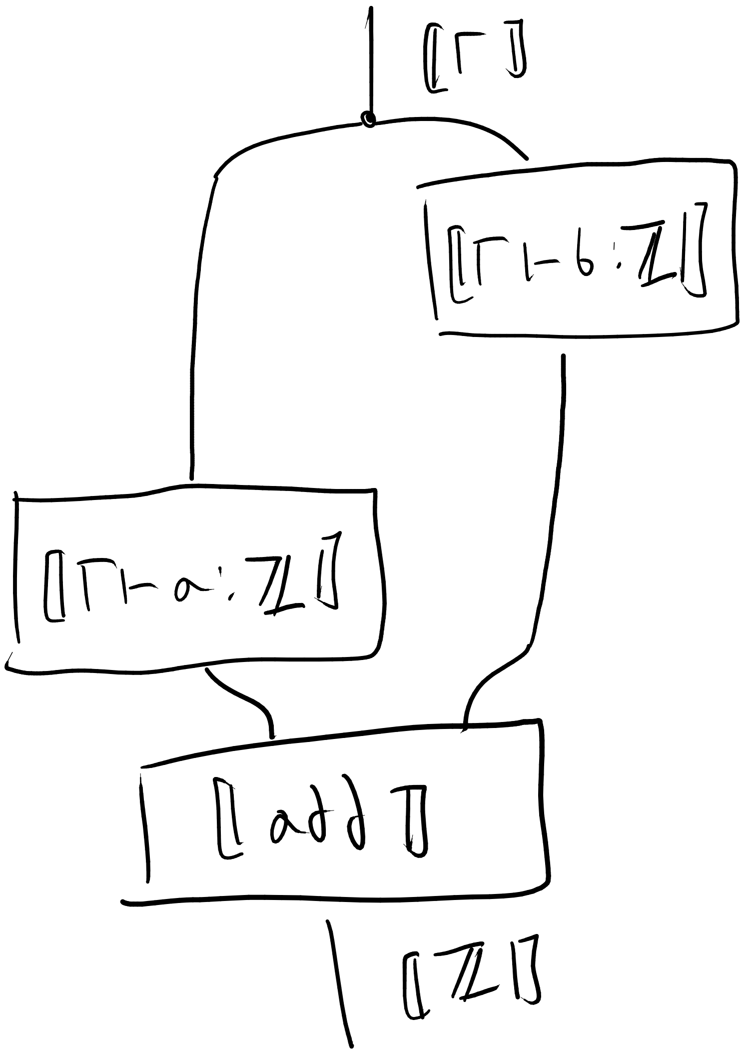

⟦Γ⟧
⟦Γ ⊢ a : ℤ⟧
⟦Γ ⊢ b : ℤ⟧
⟦add⟧
⟦ℤ⟧

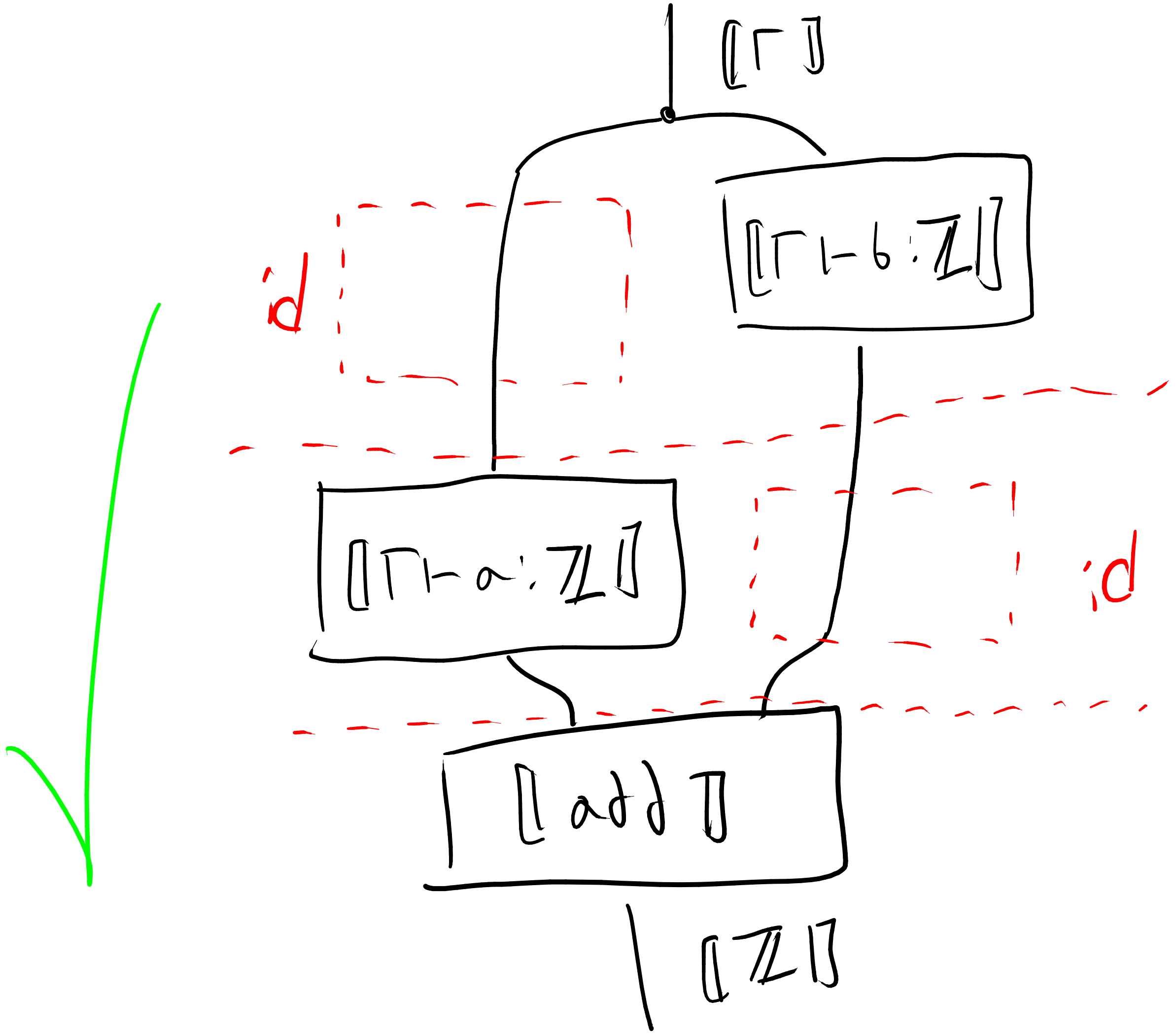
✓


⟦Γ⟧
⟦Γ ⊢ a : ℤ⟧
id
id
⟦Γ ⊢ b : ℤ⟧
⟦add⟧
⟦ℤ⟧

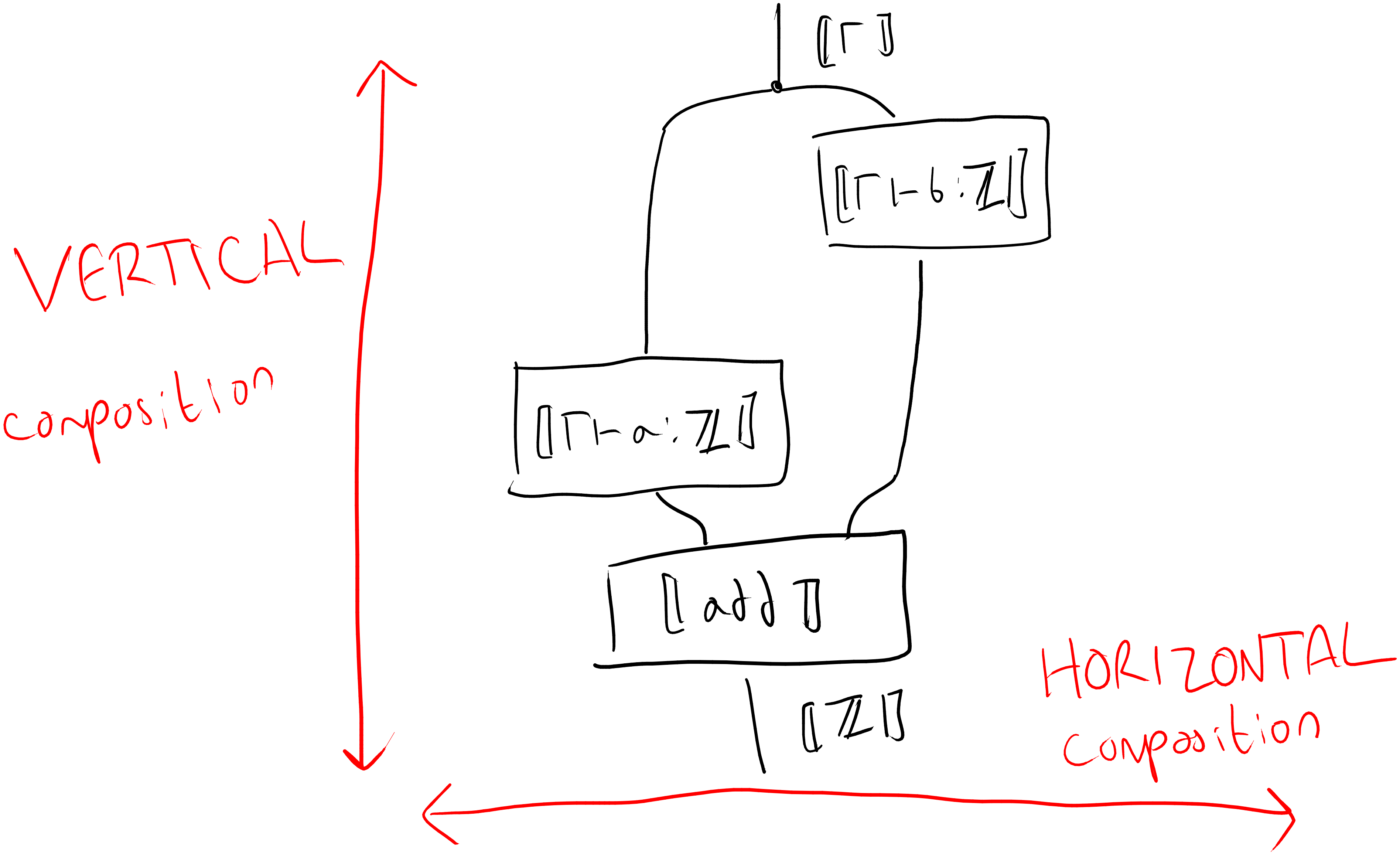


$$= \left\langle [\![\Gamma \vdash a : \mathbb{Z}]\!],\ \mathrm{id}_{[\![\Gamma]\!]} \right\rangle ; \left(\mathrm{id}_{[\![\mathbb{Z}]\!]} \times [\![\Gamma \vdash b : \mathbb{Z}]\!] \right) ; [\![\mathrm{add}]\!]$$


⟦Γ⟧
⟦Γ ⊢ b : ℤ⟧
⟦Γ ⊢ a : ℤ⟧
⟦add⟧
⟦ℤ⟧

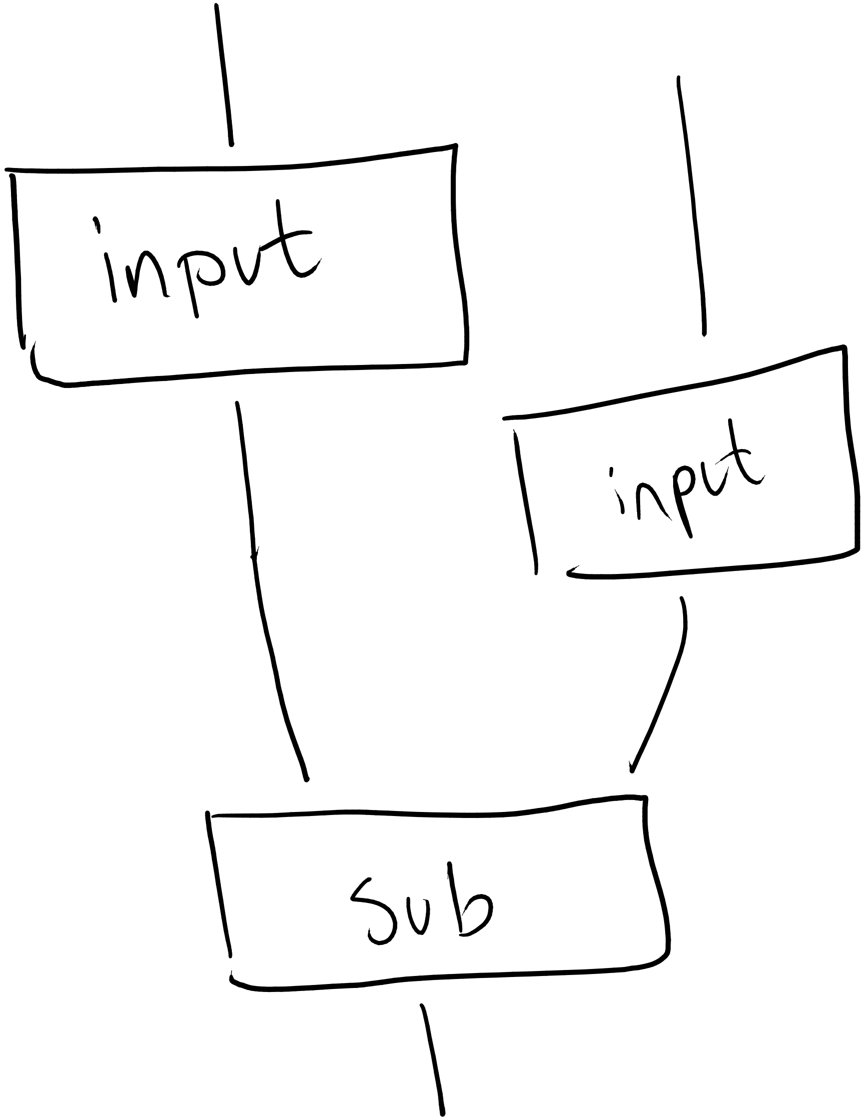


$$= \langle \mathrm{id}_{[\![\Gamma]\!]}, [\![\Gamma \vdash b : \mathbb{Z}]\!] \rangle ; ([\![\Gamma \vdash a : \mathbb{Z}]\!] \times \mathrm{id}_{[\![\mathbb{Z}]\!]}) ; [\![\mathrm{add}]\!]$$

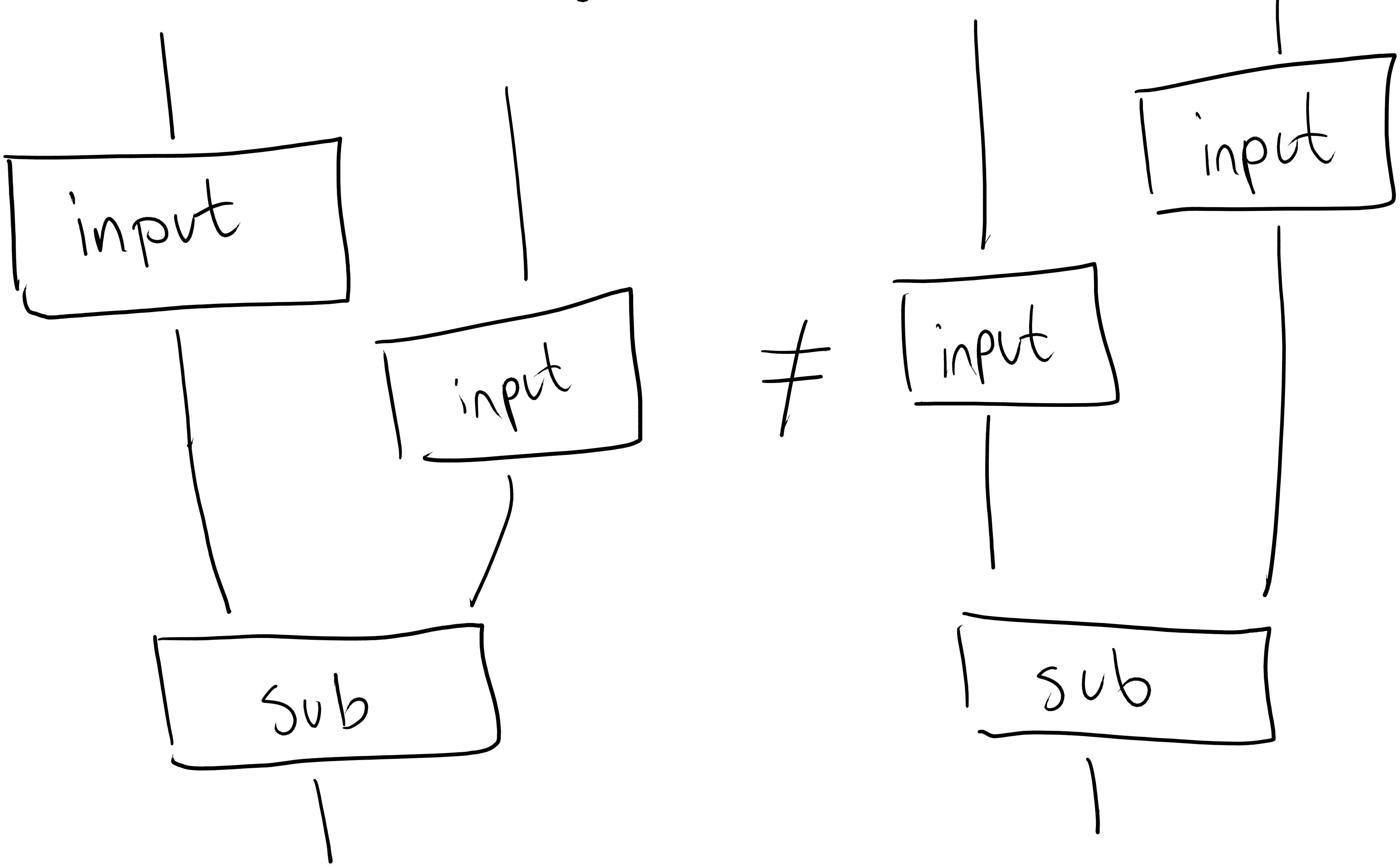

⟦Γ⟧
⟦Γ ⊢ b : ℤ⟧
⟦Γ ⊢ a : ℤ⟧
⟦add⟧
⟦ℤ⟧
VERTICAL
composition
HORIZONTAL
composition

Purity

Purity

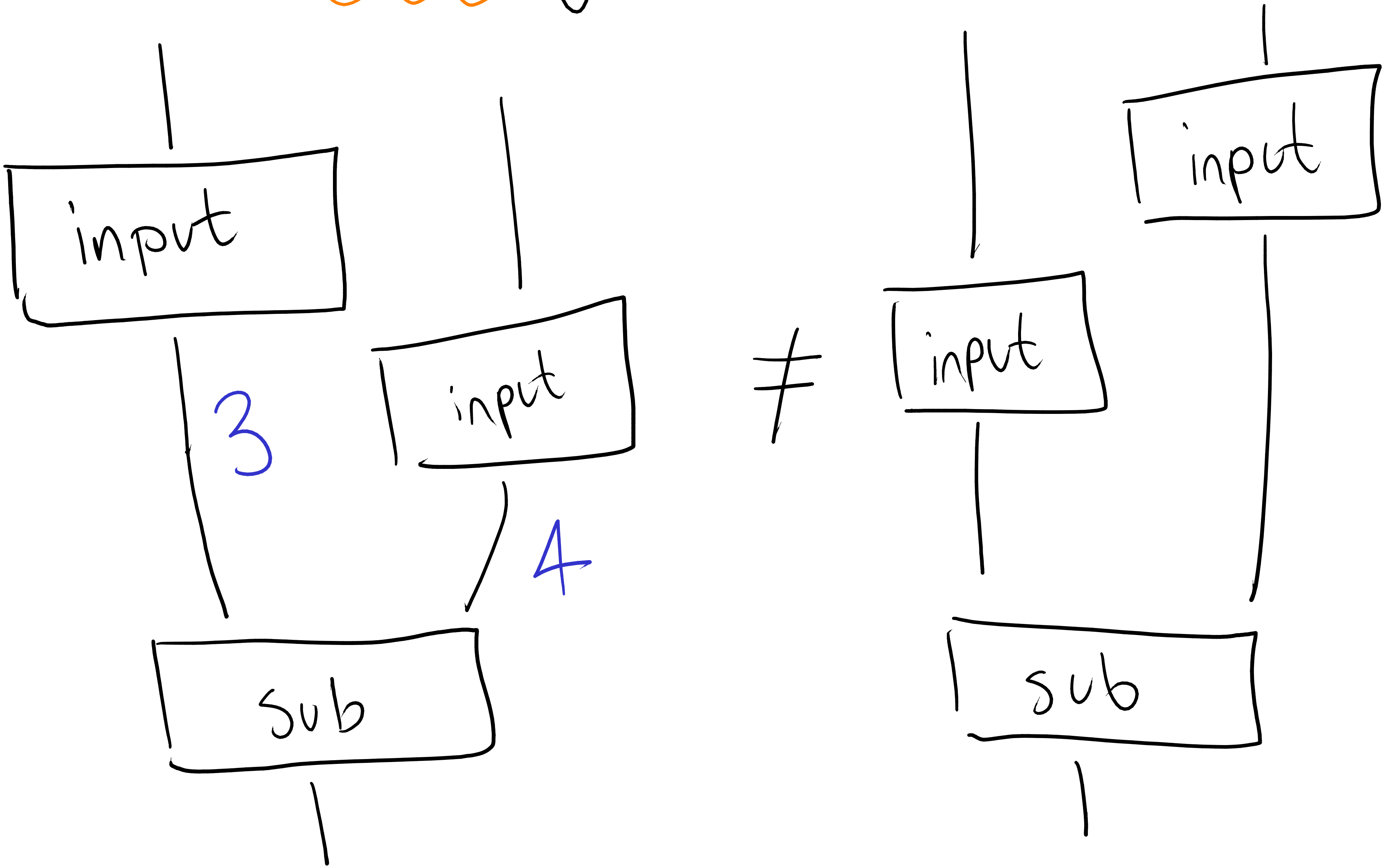


Purity



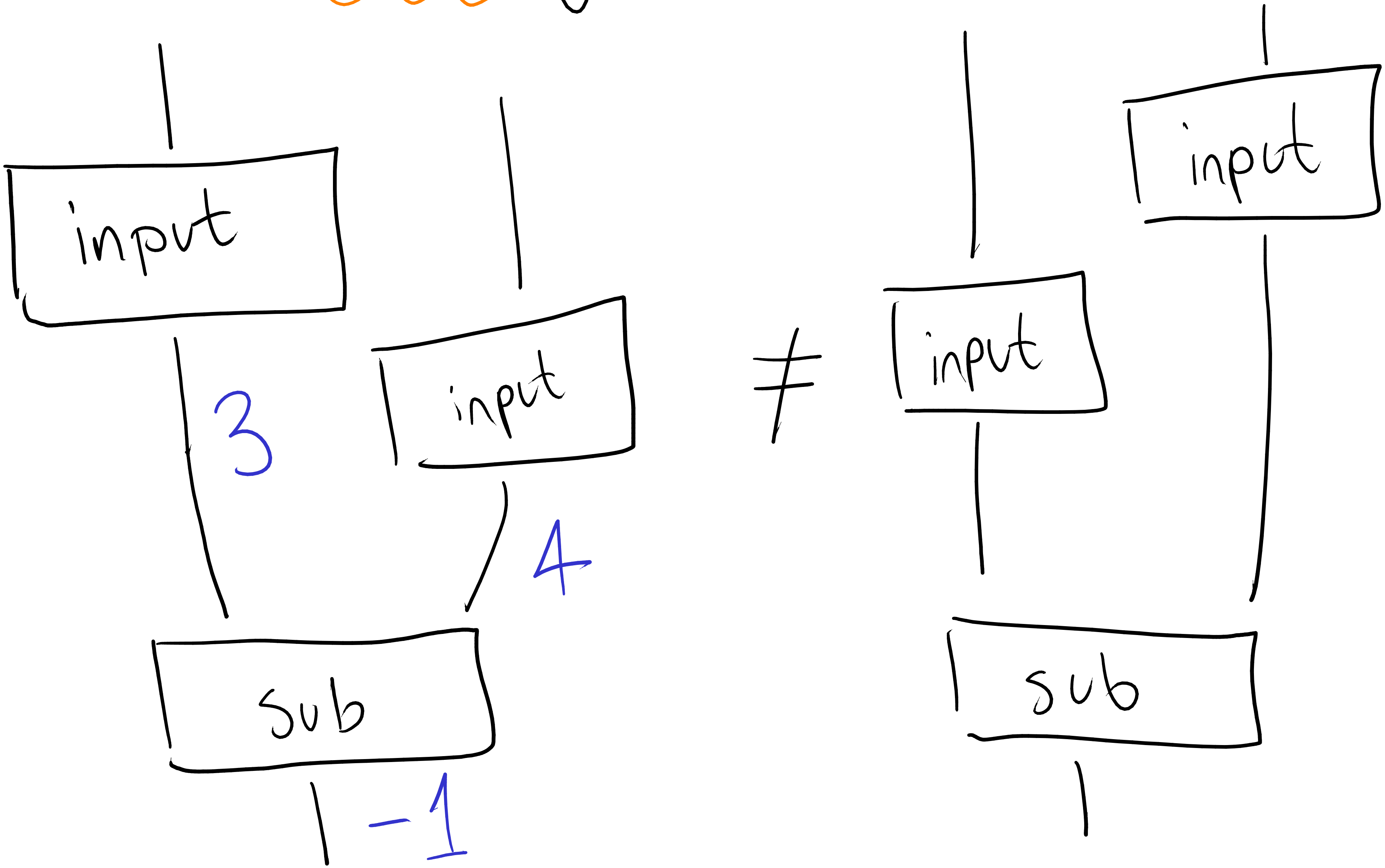
Purity

INPUT: 3, 4



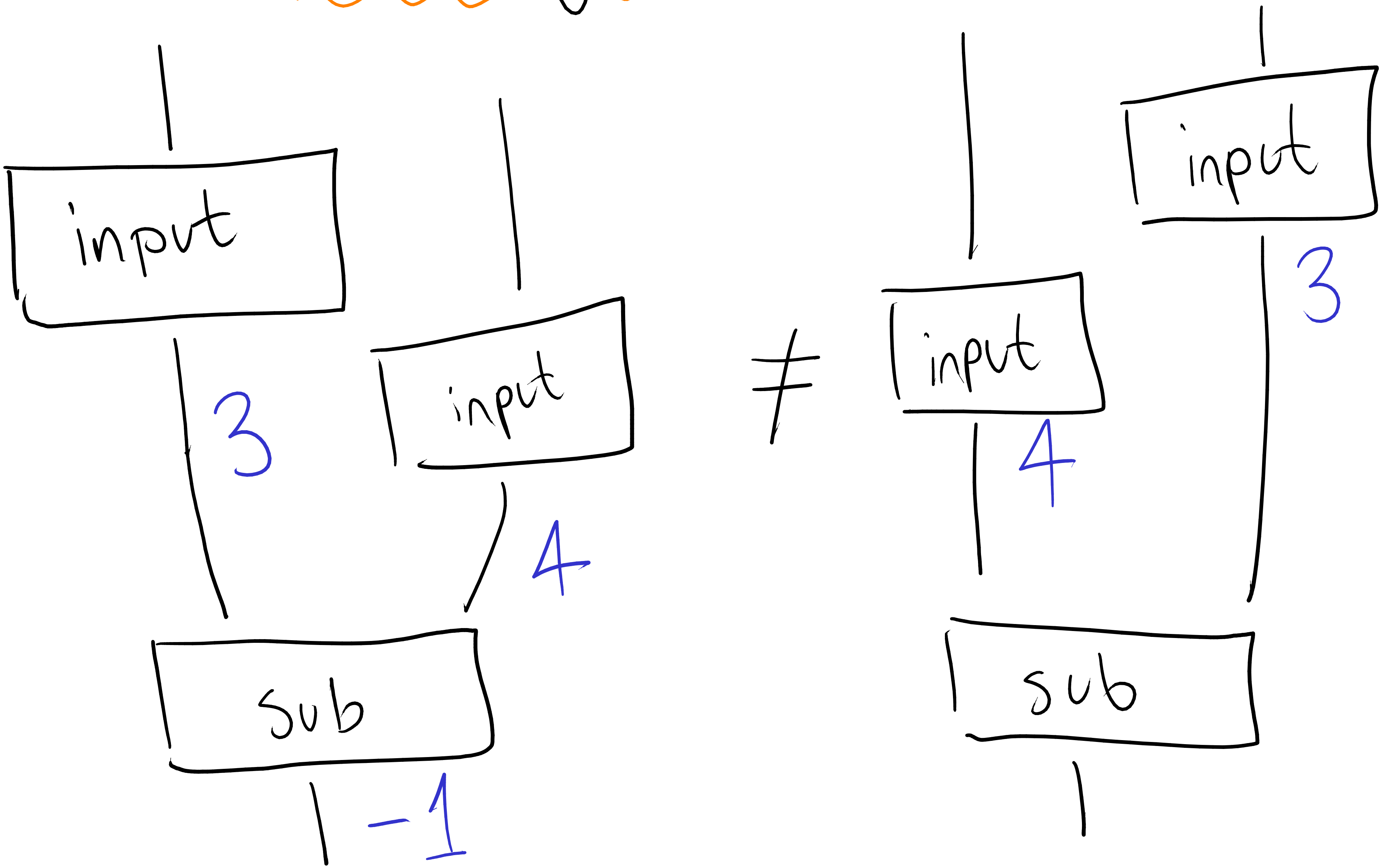
Purity

INPUT: 3, 4



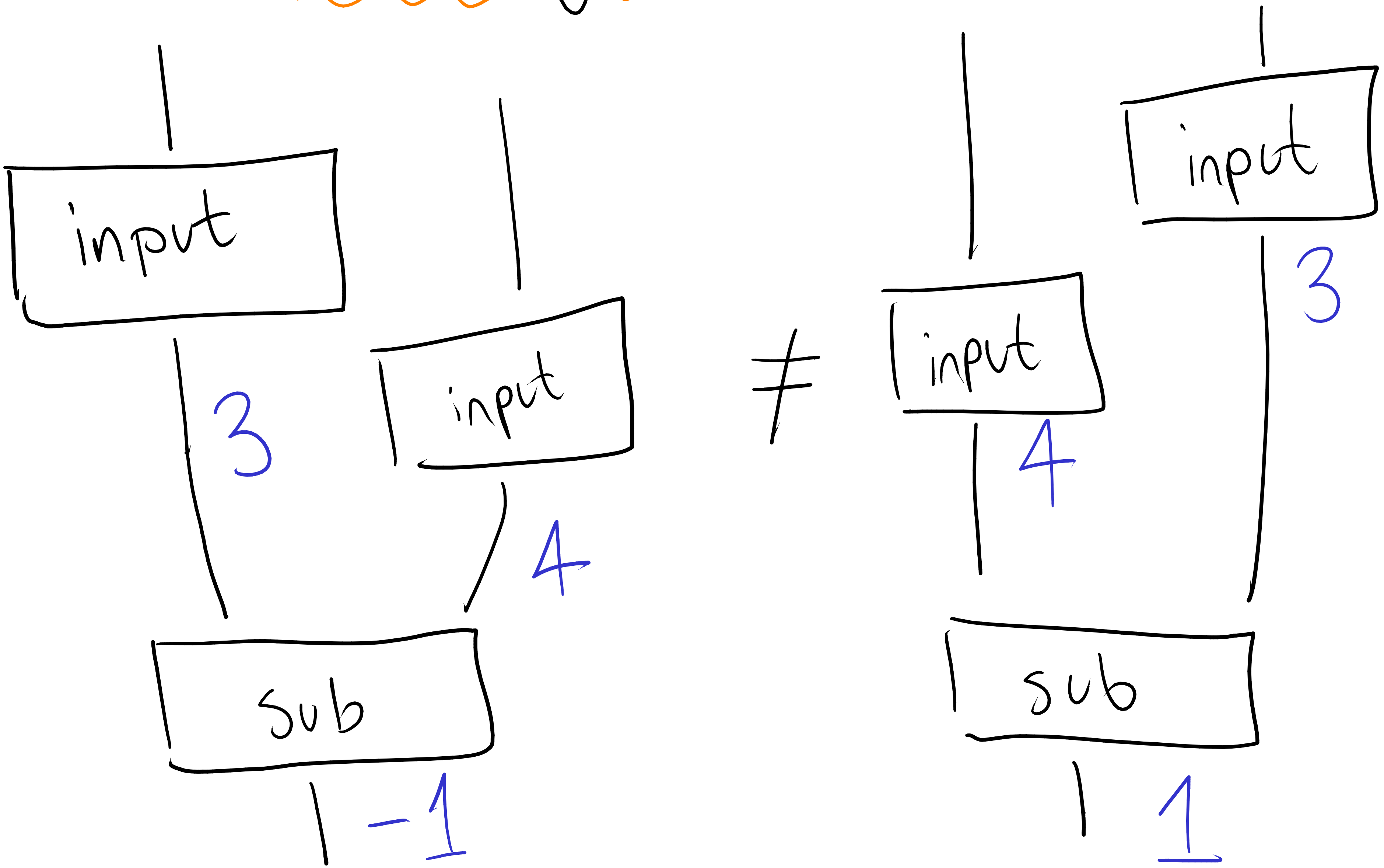
Purity

INPUT: 3, 4



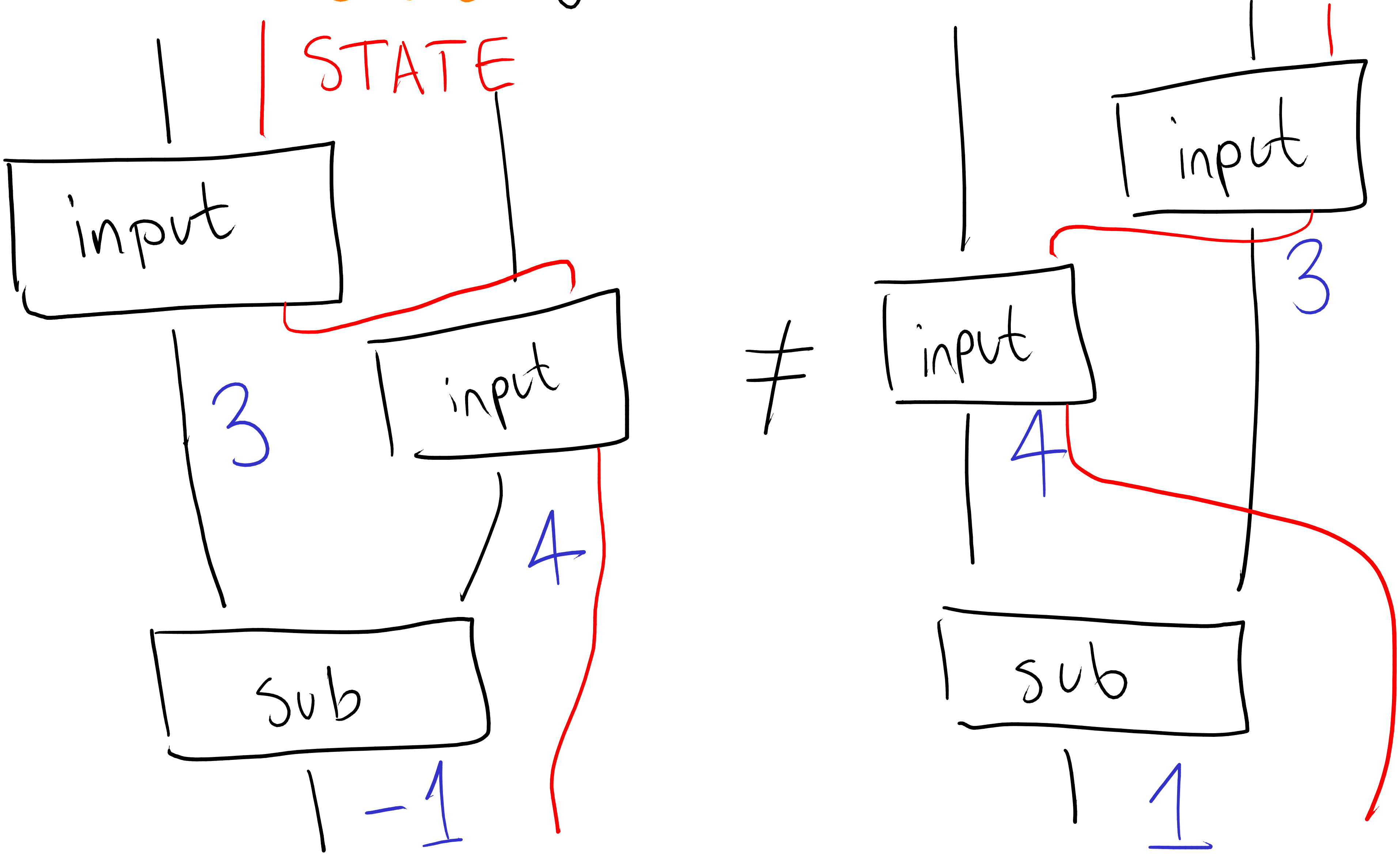
Purity

INPUT: 3, 4



Purity

INPUT: 3, 4



Cartesian vs. Tensor Product

$\langle f, g \rangle$ – cartesian product

$\pi_1 : A \otimes B \to A$ $\pi_2 : A \otimes B \to B$

Cartesian vs. Tensor Product

$\langle f, g \rangle$ – cartesian product

$\pi_1 : A \otimes B \to A \qquad \pi_2 : A \otimes B \to B$

Issue: for f, g impure, $\langle f, g \rangle / f \times g$ is ambiguous!

Notation

$$C_1(A,B) \subseteq C(A,B)$$

Notation

$$C_1(A,B) \subseteq C(A,B)$$

PURE

Notation

$$C_1(A, B) \subseteq C_0(A, B)$$

PURE

Notation

$$C_1(A,B) \subseteq C_0(A,B)$$

PURE

Define, $\forall$ obj. C, $f : C_p(A,B)$,

$f \otimes C : C_p(A \otimes C, B \otimes C)$ $\quad C \otimes f : C_p(C \otimes A, C \otimes B)$

Freyd Structure

For pure f, $c \otimes f = id_c \times f$

$f \otimes c = f \times id_c$

Freyd Structure

For <u>pure</u> f, $c \otimes f = id_c \times f$

$$f \otimes c = f \times id_c$$

$$\Rightarrow \langle f, g \rangle = \langle id, id \rangle ; f \otimes - ; - \otimes g$$

$$= \langle id, id \rangle ; - \otimes g ; f \otimes -$$

Freyd Structure

For pure f, $c \otimes f = id_c \times f$

$$f \otimes c = f \times id_c$$

$$\Rightarrow \langle f, g \rangle = \langle id, id \rangle ; f \otimes - ; - \otimes g$$

$$= \underbrace{\langle id, id \rangle}_{\text{Diagonal map } \Delta} ; - \otimes g ; f \otimes -$$

Functionality $\Leftrightarrow$ Structure

Functionality ⟺ Structure

Multiple inputs ⟺ Tensor Product

Functionality ⟺ Structure

Multiple inputs ⟺ Tensor Product

Pure/ impure ⟺ Freyd category

Semantics of Instructions

Semantics of Instructions

$e ::= \quad x$

Semantics of Instructions

VARIABLES

$$e ::= \; x$$

Semantics of Instructions

VARIABLES

$$e ::= x \mid f\ a$$

APPLICATIONS

Semantics of Instructions

VARIABLES

$$e ::= x \mid f\ a \mid (a, b)$$

APPLICATIONS

TUPLES

Semantics of Instructions

$$e ::= \overset{\text{VARIABLES}}{x} \mid \underset{\text{APPLICATIONS}}{f\ a} \mid \underset{\text{TUPLES}}{(a, b)} \mid \underset{\text{CONSTANTS}}{c}$$

Semantics of Instructions

VARIABLES

$$e ::= x \mid f\ a \mid (a, b) \mid c$$

APPLICATIONS

TUPLES

CONSTANTS

$$A ::= X$$

Base Types

Semantics of Instructions

$$e ::= \overset{\text{VARIABLES}}{\overset{\searrow}{x}} \mid \underset{\underset{\text{APPLICATIONS}}{\nearrow}}{f\ a} \mid \underset{\underset{\text{TUPLES}}{\uparrow}}{(a, b)} \mid \underset{\underset{\text{CONSTANTS}}{\uparrow}}{c}$$

$$A ::= \underset{\underset{\text{Base Types}}{\uparrow}}{X} \mid \underset{\underset{\text{Products}}{\uparrow}}{A \otimes B}$$

Semantics of Instructions

VARIABLES

$$e ::= x \mid f\ a \mid (a, b) \mid c$$

APPLICATIONS TUPLES CONSTANTS

Assume

$$1, 2 \in X$$

$$A ::= X \mid A \otimes B$$

Base Types Products

Semantics of Instructions

$$e ::= \overset{\text{VARIABLES}}{\searrow}\, x \mid \underset{\overset{\nearrow}{\text{APPLICATIONS}}}{f\ a} \mid \underset{\overset{\uparrow}{\text{TUPLES}}}{(a, b)} \mid \underset{\overset{\uparrow}{\text{CONSTANTS}}}{c}$$

Assume

$$A ::= \underset{\overset{\uparrow}{\text{Base Types}}}{X} \mid \underset{\overset{\uparrow}{\text{Products}}}{A \otimes B} \qquad \underset{\overset{\uparrow}{\text{Unit Type}}}{\underline{1}}, 2 \in X$$

Semantics of Instructions

VARIABLES

$$e ::= x \mid f\ a \mid (a, b) \mid c$$

APPLICATIONS

TUPLES

CONSTANTS

Assume

$$A ::= X \mid A \otimes B \qquad \underline{1}, 2 \in X$$

Base Types

Products

Unit Type

Booleans

Semantics of Instructions

$$e ::= x \mid f\ a \mid (a, b) \mid c$$

$$A ::= X \mid A \otimes B \quad \text{where } 1, 2 \in X$$

$$\Gamma ::= \cdot \mid \Gamma, x : A$$

Semantics of Instructions

$$e ::= x \mid f\ a \mid (a, b) \mid c$$

$$A ::= X \mid A \otimes B \quad \text{where } 1, 2 \in X$$

$$\Gamma ::= \cdot \mid \Gamma, x : A$$

variable type

variable name

Semantics of Instructions

$$\Gamma \vdash e : A$$

Context Instruction Type

Semantics of Instructions

$$\Gamma \vdash_{p} e : A$$

Context Instruction Type

PURITY $p \in \{0, 1\}$

Semantics of Instructions

$$\Gamma \vdash_p e : A$$

Context Instruction Type

PURITY $p \in \{0,1\}$

$p=0 \Rightarrow$ IMPURE

Semantics of Instructions

$$\Gamma \vdash_p e : A$$

Context Instruction Type

PURITY $p \in \{0, 1\}$

$p = 0 \Rightarrow$ IMPURE

$p = 1 \Rightarrow$ PURE

Semantics of Instructions

$$[\![\Gamma \vdash_p e : A]\!] : C_p([\![\Gamma]\!], [\![A]\!])$$

Semantics of Instructions

$$[\![\Gamma \vdash_p e : A]\!] : C_p([\![\Gamma]\!], [\![A]\!])$$

$[\![X]\!] \in |C|$ where $[\![1]\!] = I$ $\quad [\![2]\!] = I + I$

Semantics of Instructions

$$[\![\Gamma \vdash_p e : A]\!] : C_p([\![\Gamma]\!], [\![A]\!])$$

$[\![X]\!] \in |C|$ where $[\![1]\!] = I$ $[\![2]\!] = I + I$

$[\![A \otimes B]\!] = [\![A]\!] \otimes [\![B]\!]$

Semantics of Instructions

$$[\![\Gamma \vdash_p e : A]\!] : C_p([\![\Gamma]\!], [\![A]\!])$$

$[\![X]\!] \in |C|$ where $[\![1]\!] = I$ $\quad [\![2]\!] = I + I$

$[\![A \otimes B]\!] = [\![A]\!] \otimes [\![B]\!]$

$[\![\cdot]\!] = I$ $\qquad [\![\Gamma, x : A]\!] = [\![\Gamma]\!] \otimes [\![A]\!]$

$$\frac{\quad}{\Gamma \vdash_p f\ a : B}$$

$$\frac{f \in \mathrm{inst}_p(A, B)}{\Gamma \vdash_p f\ a : B}$$

Instruction Purity

$$\frac{f \in \mathrm{inst}_{\boxed{p}}(A, B)}{\Gamma \vdash_p f\ a : B}$$

Instruction Purity

$inst_1(A,B) \subseteq inst_0(A,B)$

$$\frac{f \in inst_{\enclose{circle}{p}}(A,B)}{\Gamma \vdash_p f\ a : B}$$

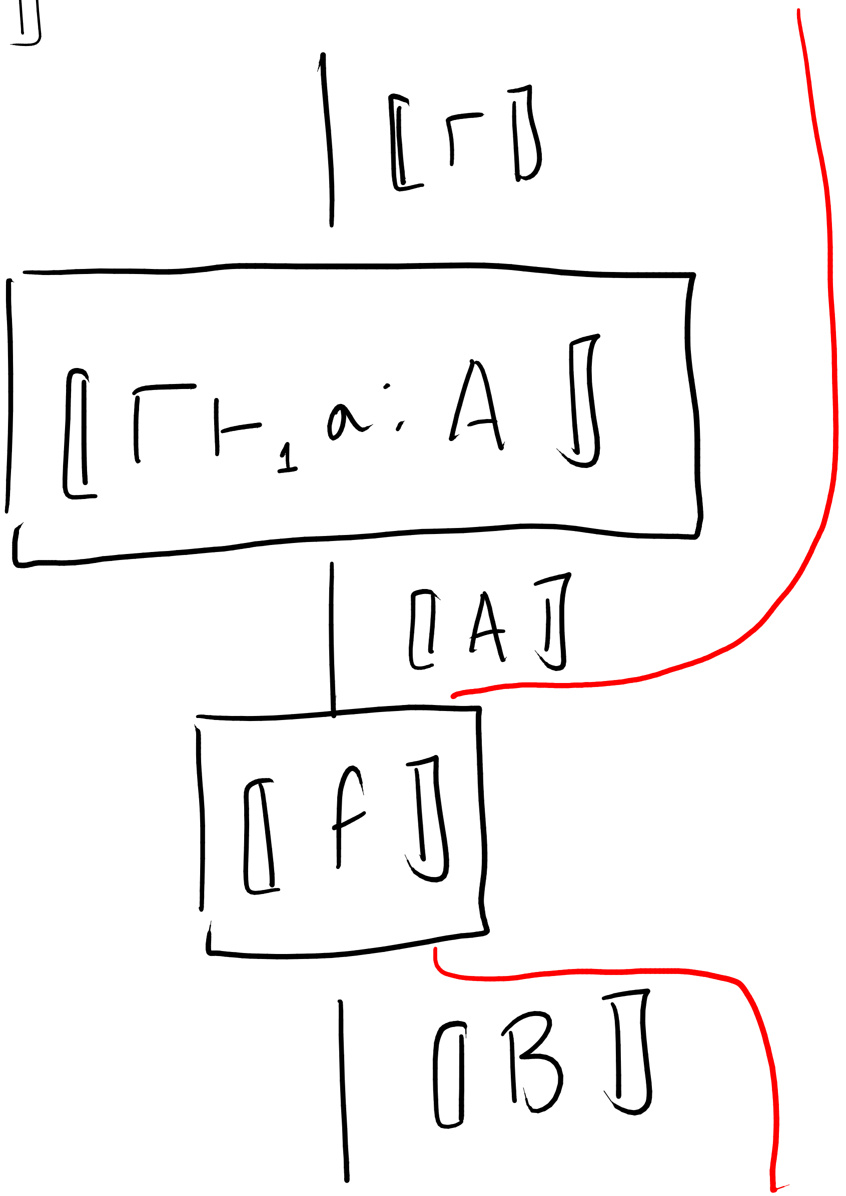
$$\frac{f \in \mathrm{inst}_p(A,B) \qquad \Gamma \vdash_1 a : A}{\Gamma \vdash_p f\ a : B}$$

Argument is <u>always</u> pure!

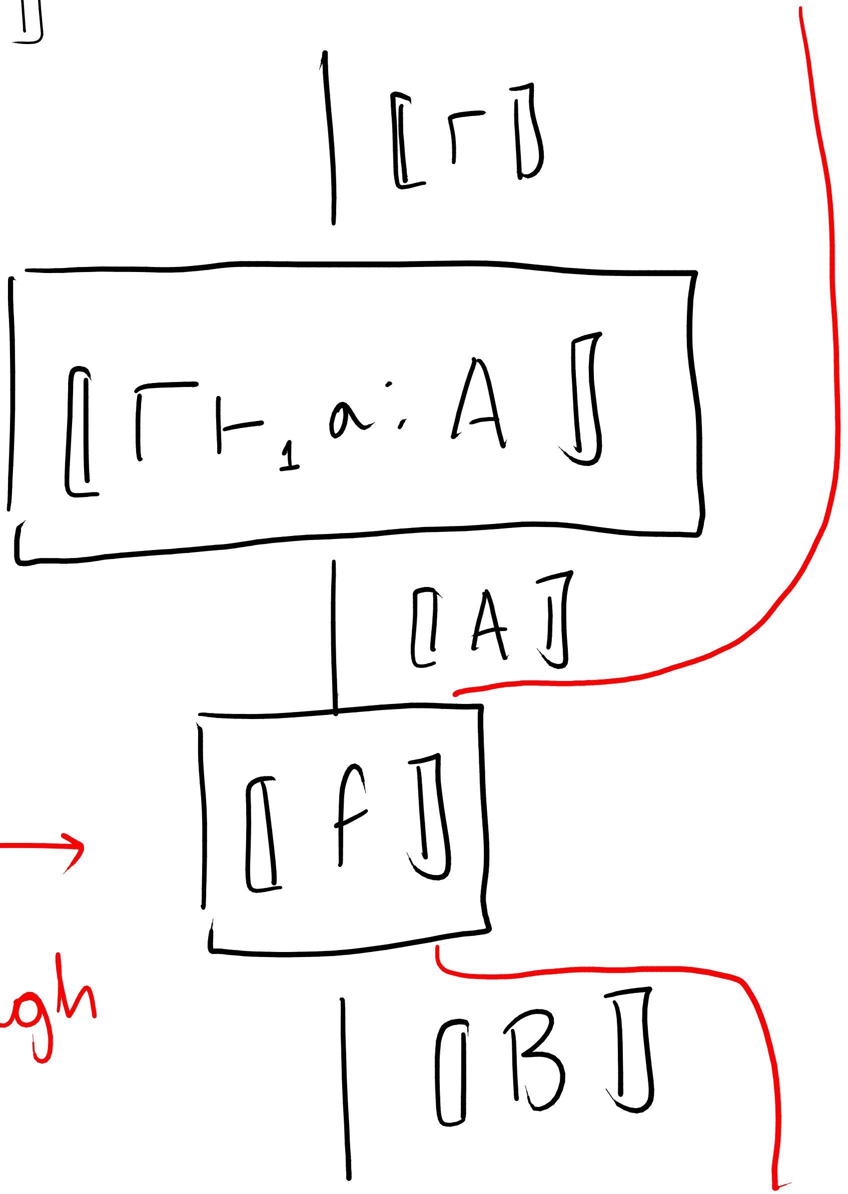
$$\frac{f \in \mathrm{inst}_p(A, B) \qquad \Gamma \vdash_{①} a : A}{\Gamma \vdash_p f\ a : B}$$

$$\left[\!\!\left[\frac{f \in \mathrm{inst}_p(A,B) \qquad \Gamma \vdash_1 a : A}{\Gamma \vdash_p f\; a : B} \right]\!\!\right]$$

$$\left[\!\left[\frac{f \in \mathrm{inst}_p(A,B) \qquad \Gamma \vdash_1 a : A}{\Gamma \vdash_p f\ a : B} \right]\!\right] =$$



$$\left[\!\!\left[\frac{f \in \mathrm{inst}_p(A,B) \qquad \Gamma \vdash_1 a : A}{\Gamma \vdash_p f\ a : B} \right]\!\!\right] =$$



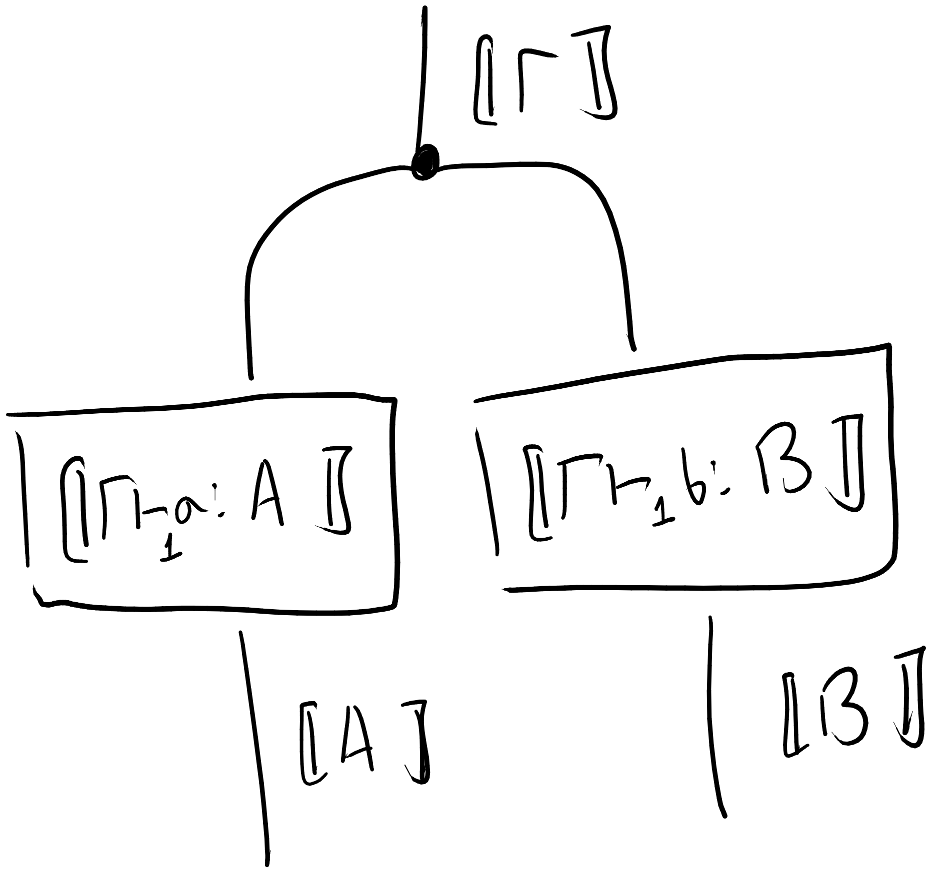
If something is POTENTIALLY impure, we thread the state wire through it! →

$$\left[\frac{\Gamma \vdash_1 a : A \qquad \Gamma \vdash_1 b : B}{\Gamma \vdash_p (a,b) : A \otimes B} \right]$$

$$\left[\frac{\Gamma \vdash_{\textcircled{1}} a : A \qquad \Gamma \vdash_{\textcircled{1}} b : B}{\Gamma \vdash_{p} (a,b) : A \otimes B}\right]$$

Note: <u>both</u> components of a pair <u>must</u> be pure!

$$\left[\!\left[\frac{\Gamma \vdash_1 a : A \qquad \Gamma \vdash_1 b : B}{\Gamma \vdash_p (a,b) : A \otimes B} \right]\!\right] =$$



Constants

Constants

$$\frac{c \in \mathrm{consts}(A)}{\Gamma \vdash_1 c : A}$$

Constants

$$\left[\!\left[\frac{c \in \mathrm{consts}(A)}{\Gamma \vdash_1 c : A} \right]\!\right] =$$

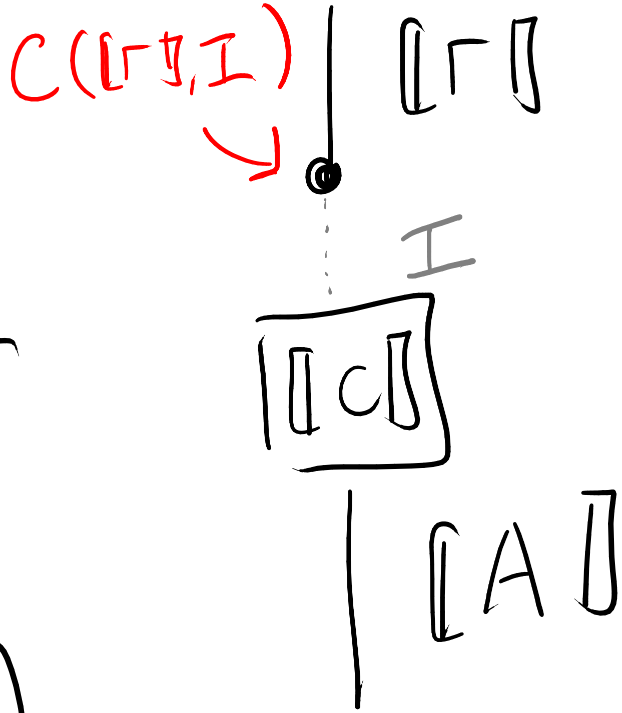
Constants

$$\left\llbracket \frac{c \in \mathrm{consts}(A)}{\Gamma \vdash_1 c : A} \right\rrbracket =$$

Here $\llbracket c \rrbracket : \mathcal{C}(I, \llbracket A \rrbracket)$

Constants

$$\left[\!\left[\frac{c \in \mathrm{consts}(A)}{\Gamma \vdash_1 c : A} \right]\!\right] =$$



Here $\llbracket c\rrbracket : C(I, \llbracket A\rrbracket)$

Variables

$$\frac{\Gamma \mapsto x : A}{\Gamma \vdash x : A}$$

Variables

$$\frac{\Gamma \mapsto x : A}{\Gamma \vdash x : A}$$

"$x : A$ is a WEAKENING of Γ"

Variables

$$\frac{\Gamma \mapsto x : A}{\Gamma \vdash x : A}$$

"$x : A$ is a WEAKENING of Γ"

"Γ has more variables than $x : A$"

Variables

$$\left[\!\left[\frac{\Gamma \mapsto x : A}{\Gamma \vdash x : A} \right]\!\right] = [\![\Gamma \mapsto x : A]\!]$$

Weakening

$$\Gamma \vdash \longrightarrow \Delta$$

Weakening

$$\Gamma \mapsto \Delta$$

" Γ weakens Δ "

Weakening

$$\Gamma \mapsto \Delta$$

"Γ weakens Δ"

"Γ has more vars than Δ"

Weakening

$$[\![\Gamma \mapsto \Delta]\!] : \mathcal{C}_{\underline{1}}([\![\Gamma]\!], [\![\Delta]\!])$$

Weakening

$$[\![\Gamma \mapsto \Delta]\!] : \mathcal{C}_1([\![\Gamma]\!], [\![\Delta]\!])$$

"Drop all variables from $[\![\Gamma]\!]$ which do not appear in Δ"

Weakening

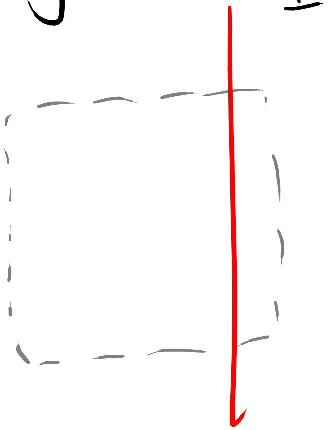
$$\llbracket \Gamma \mapsto \Delta \rrbracket : \mathcal{C}_1(\llbracket \Gamma \rrbracket, \llbracket \Delta \rrbracket)$$

$$\left\llbracket \frac{}{\cdot \mapsto \cdot} \right\rrbracket = id_I$$

Weakening

$$[\![\Gamma \vdash \Delta]\!] : \mathcal{C}_1([\![\Gamma]\!], [\![\Delta]\!])$$

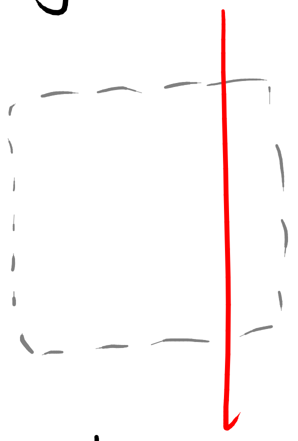
$$\left[\!\!\left[\frac{}{\cdot \vdash \cdot} \right]\!\!\right] = id_I =$$



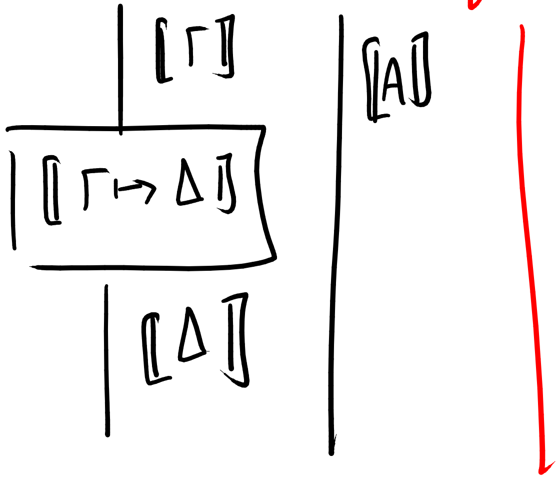
Weakening

$$[\![\Gamma \mapsto \Delta]\!] : \mathcal{C}_1([\![\Gamma]\!], [\![\Delta]\!])$$

$$\left[\!\!\left[\frac{}{\cdot \mapsto \cdot}\right]\!\!\right] = id_I =$$



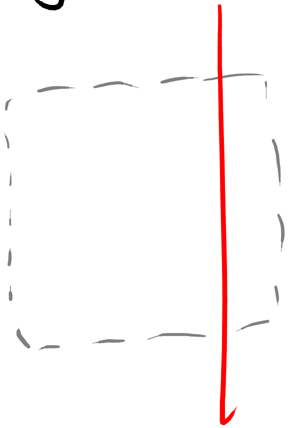
$$\left[\!\!\left[\frac{\Gamma \mapsto \Delta}{\Gamma, x:A \mapsto \Delta, x:A}\right]\!\!\right] =$$



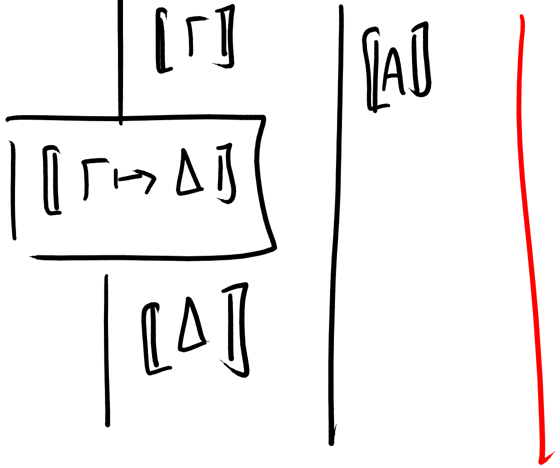
Weakening

$$[\![\Gamma \mapsto \Delta]\!] : \mathcal{C}_1([\![\Gamma]\!], [\![\Delta]\!])$$

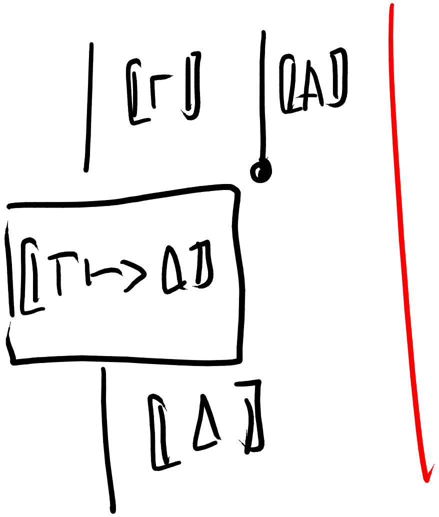
$$\left[\!\!\left[\frac{}{\circ \mapsto \circ} \right]\!\!\right] = id_I =$$



$$\left[\!\!\left[\frac{\Gamma \mapsto \Delta}{\Gamma, x:A \mapsto \Delta, x:A} \right]\!\!\right] =$$



$$\left[\!\!\left[\frac{\Gamma \mapsto \Delta}{\Gamma, x:A \mapsto \Delta} \right]\!\!\right] =$$



Thm : Weakening

Thm: Weakening

$\Gamma \vdash \Delta$ and $\Delta \vdash_p a : A$

$\Rightarrow \Gamma \vdash_p a : A$

Thm: Semantic Weakening

$$\Gamma \mapsto \Delta \quad \text{and} \quad \Delta \vdash_p a : A$$

$$\Rightarrow [\![\Gamma \vdash_p a : A]\!]$$

$$= [\![\Gamma \mapsto \Delta]\!] ; [\![\Delta \vdash_p a : A]\!]$$

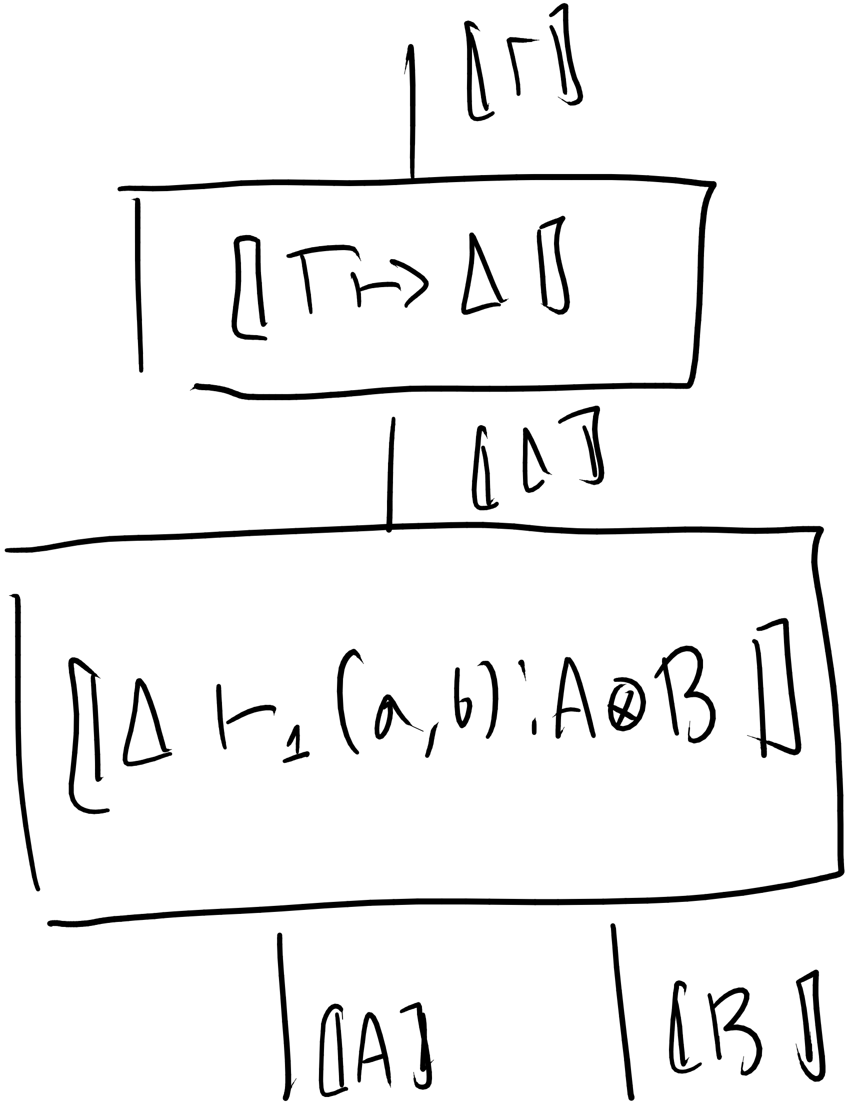
A Picture is not

a Proof

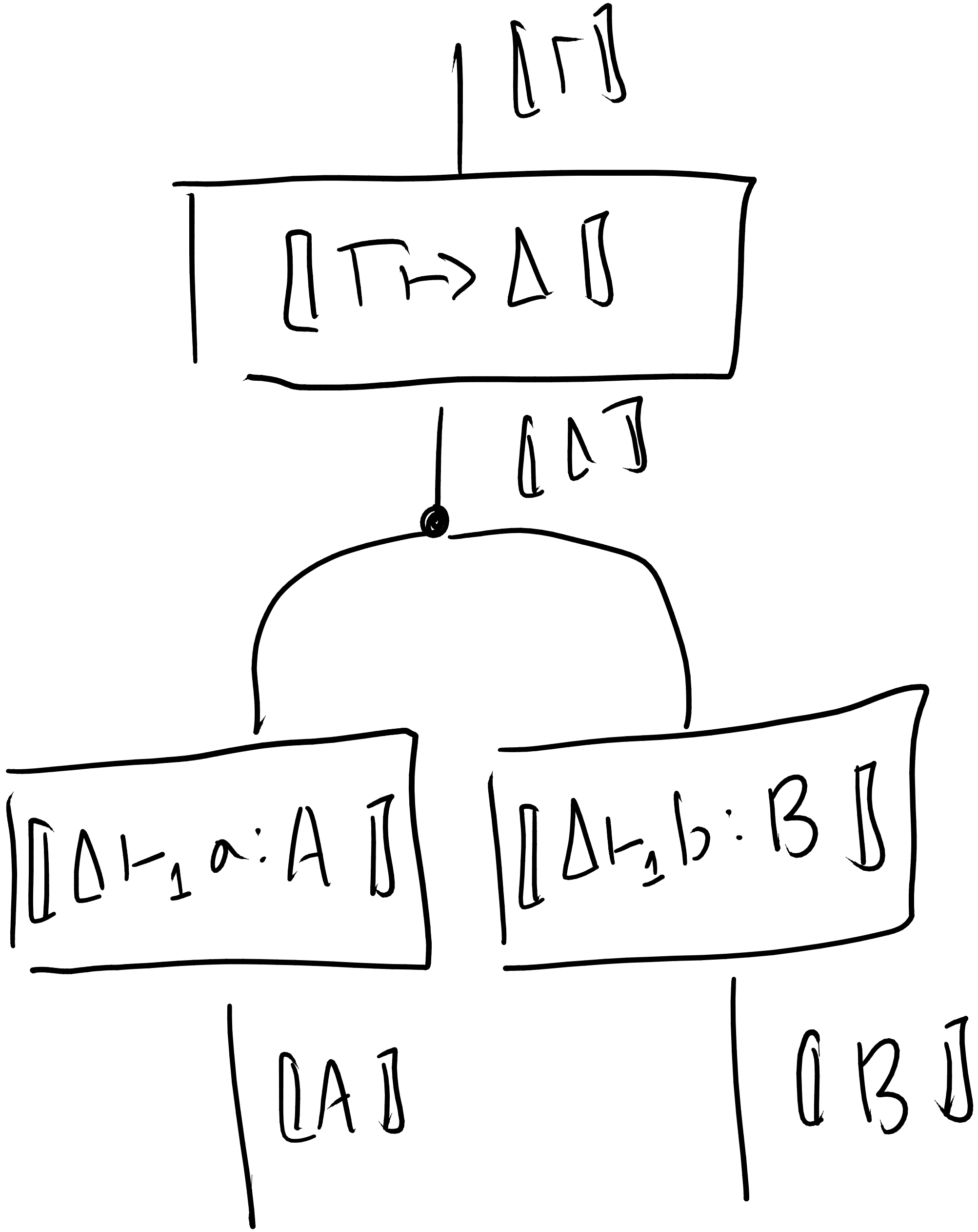
A Picture is ~~not~~
a Proof

A Picture is ~~not~~ a Proof

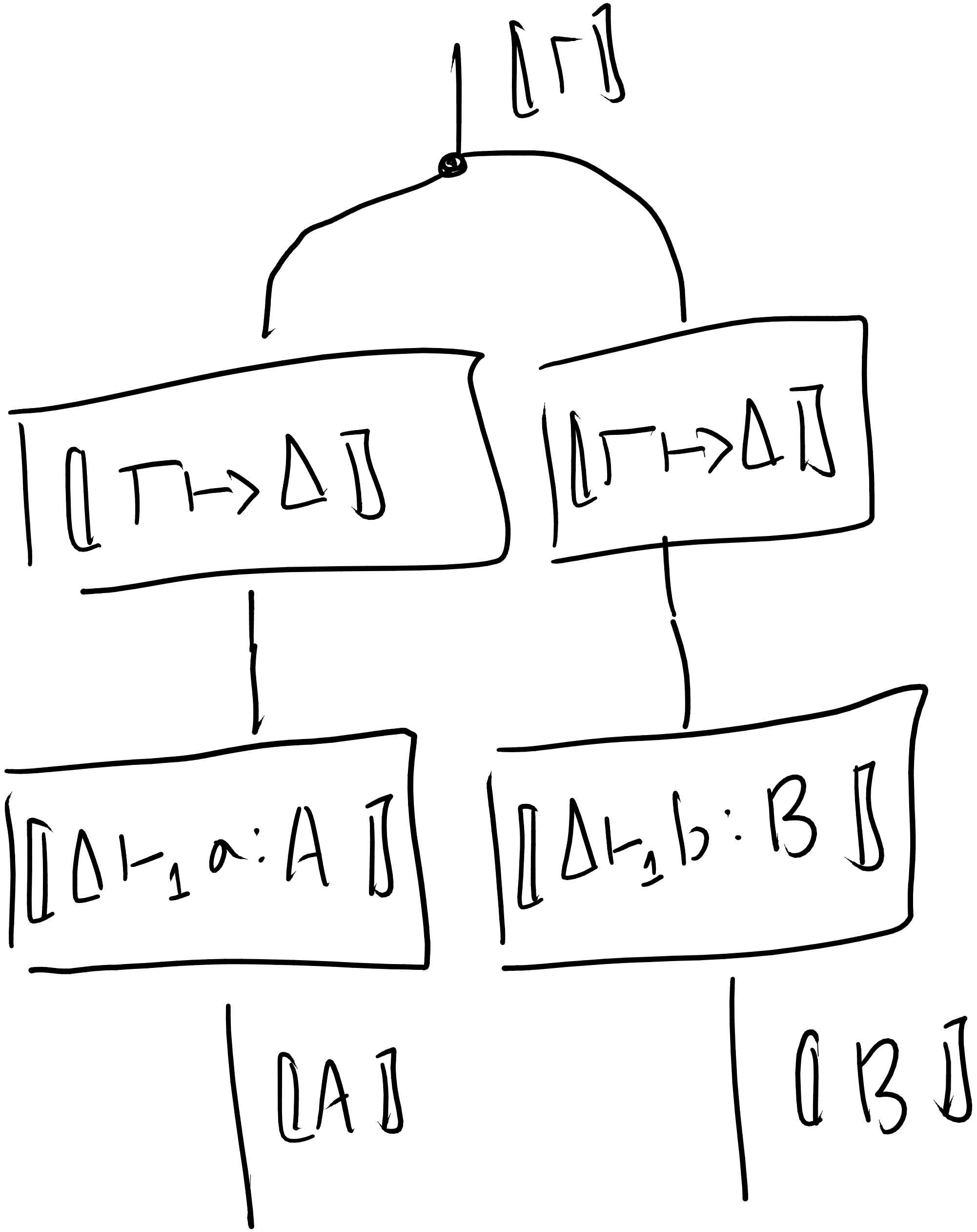
$\neg\neg A \Rightarrow A$ mmkay


⟦Γ⟧
⟦Γ ↦ Δ⟧
⟦Δ⟧
⟦Δ ⊢₁ (a, b) : A⊗B⟧
⟦A⟧
⟦B⟧

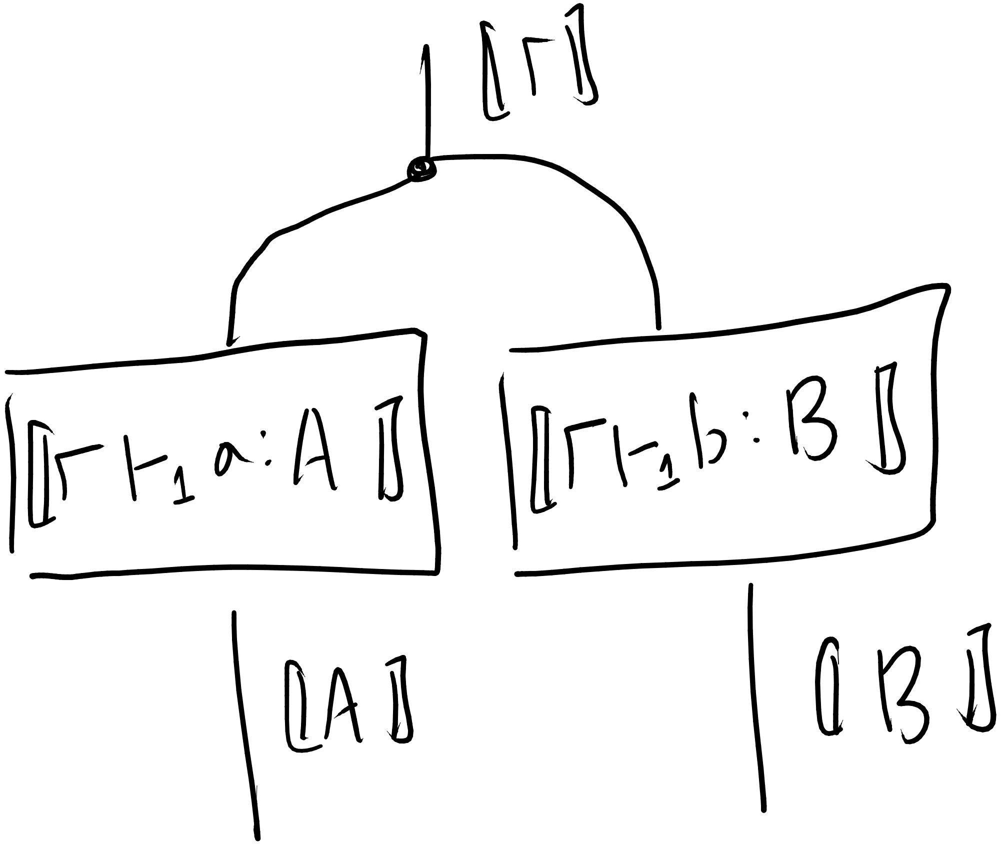
By Def'n



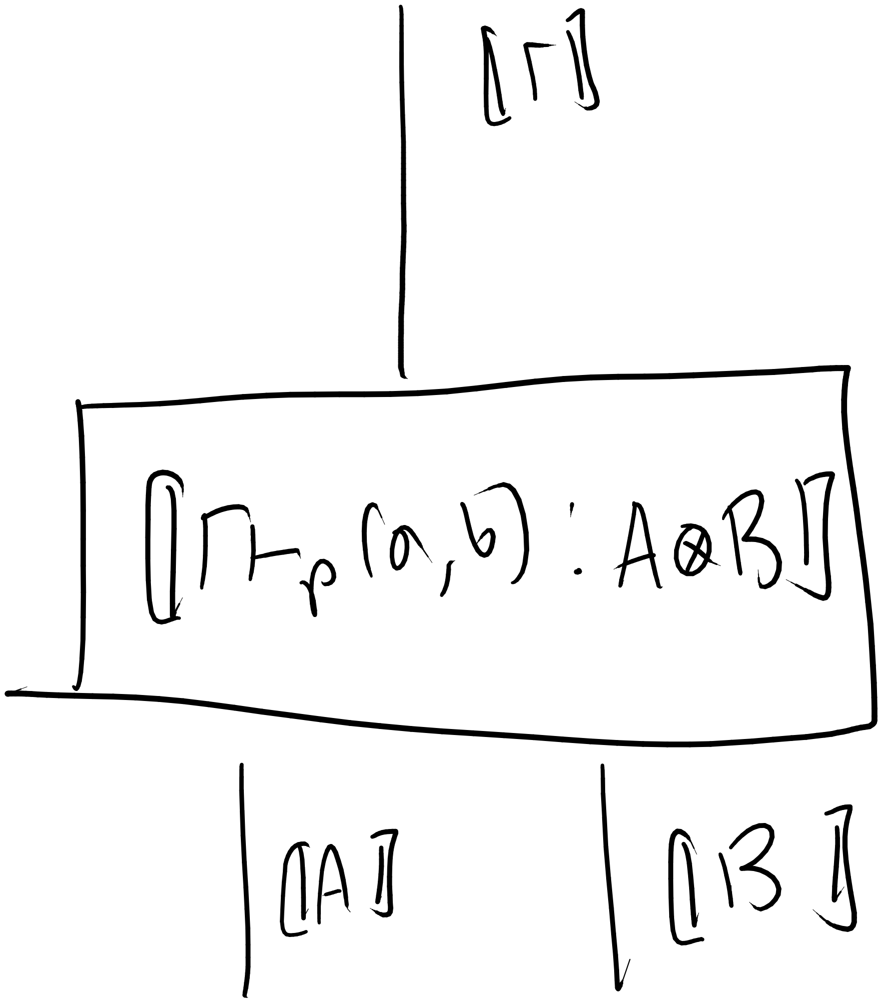
By Purity



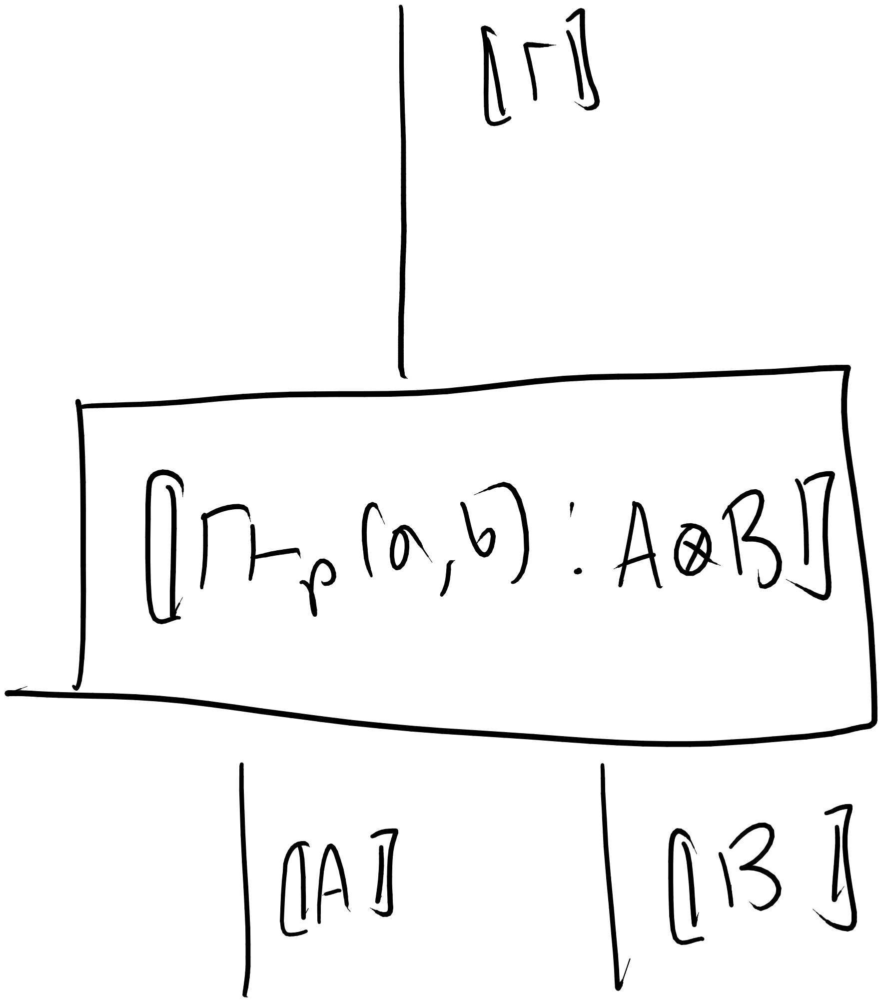
By Induction



By Def'n

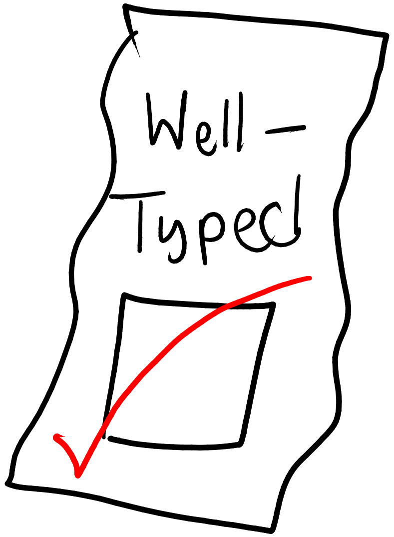


By Def'n

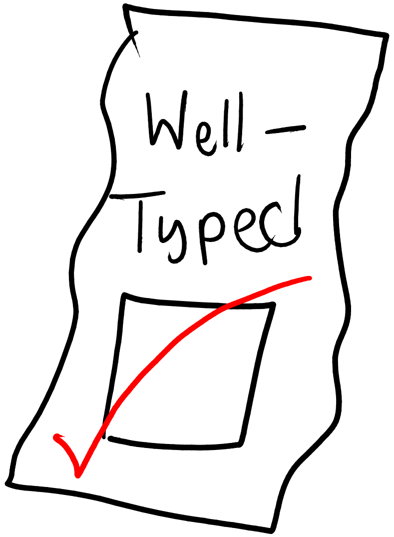


As Desired!

Type Theory Checklist

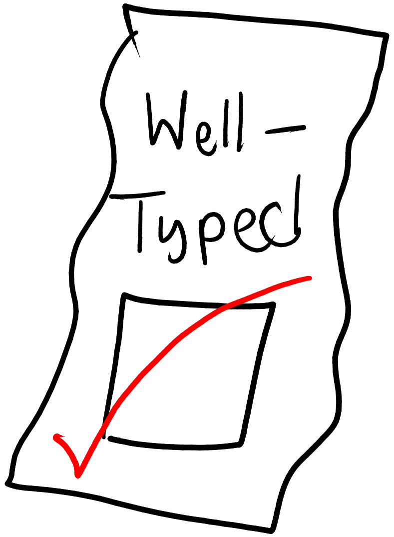


Type Theory Checklist



- Weakening
- Substitution
- Semantics
- Semantic Weakening
- Semantic Substitution

Type Theory Checklist



- ~~Weakening~~
- Substitution
- ~~Semantics~~
- ~~Semantic Weakening~~
- Semantic Substitution

Instructions

~~Instructions~~

~~Instructions~~

Blocks ?

Regions ?

~~Instructions~~

Blocks

Regions

~~Instructions~~

Blocks ←

Regions

~~Instructions~~

Blocks ←

Regions

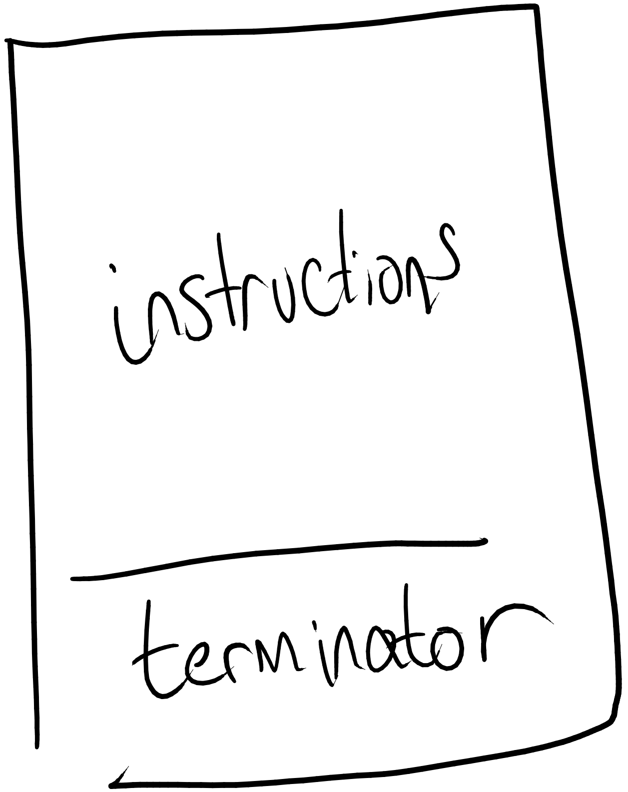


Fig. A Basic Block

~~Instructions~~

Blocks ←

Regions

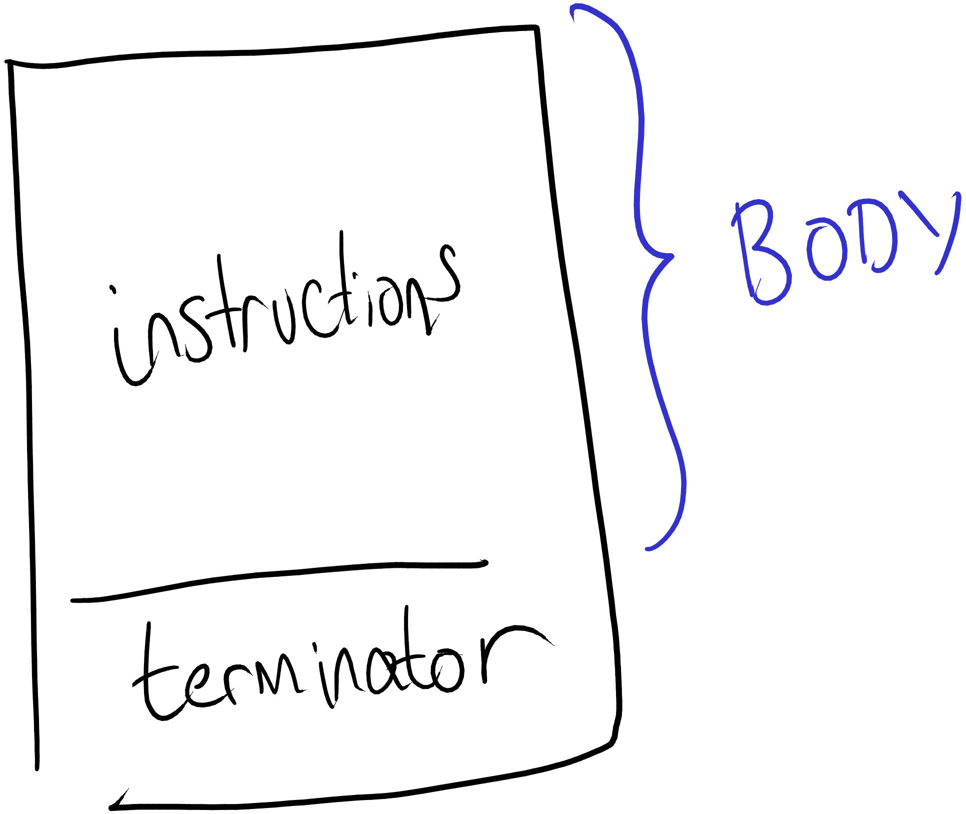


Fig. A Basic Block

Grammar for Blocks

$\beta ::= b \, ; \, t$

Grammar for Blocks

$$\overset{\text{Basic Block}}{\overset{\downarrow}{\beta}} ::= b \, ; t$$

Grammar for Blocks

$$\overset{\text{Basic Block}}{\beta} ::= \overset{\text{Body}}{b} ; t$$

Grammar for Blocks

$$\overset{\text{Basic Block}}{\overset{\downarrow}{\beta}} ::= \overset{\text{Body}}{b} \,; t \leftarrow \text{Terminator}$$

Grammar for Blocks

$$\beta ::= b ; t$$

$$b ::= \cdot \mid \text{let } x = e ; b \mid \text{let } (x, y) = e ; b$$

Grammar for Blocks

$$\beta ::= b\ ;\ t$$

$$b ::= \cdot \ \mid\ \mathrm{let}\ x = e\ ;\ b\ \mid\ \mathrm{let}\ (x, y) = e\ ;\ b$$

$$t ::= \mathrm{br}\ {}^\wedge\ell\ e$$

Grammar for Blocks

$$\beta ::= b;\ t$$

$$b ::= \cdot \mid \text{let } x = e;\ b \mid \text{let } (x, y) = e;\ b$$

Label to branch to

$$t ::= \text{br } {}^{\wedge}\ell \ e$$

Grammar for Blocks

$$\beta ::= b ; t$$

$$b ::= \cdot \mid \text{let } x = e ; b \mid \text{let } (x, y) = e ; b$$

Label to branch to

Argument to target block

$$t ::= \text{br } {}^\wedge\ell \ e$$

Grammar for Blocks

$$\beta ::= b ; t$$

$$b ::= \cdot \mid \text{let } x = e ; b \mid \text{let } (x, y) = e ; b$$

$$t ::= \text{br } {}^{\wedge}\ell \; e \mid \text{if } e \; \{ s \} \text{ else } \{ t \}$$

Grammar for Blocks

$$\beta ::= b ; t$$

$$b ::= \cdot \mid \text{let } x = e ; b \mid \text{let } (x,y) = e ; b$$

$$t ::= \text{br } {}^{\wedge}\ell \; e \mid \text{if } e \, \{ s \} \text{ else } \{ t \}$$

$$\Gamma \vdash_p b : \Delta$$

Grammar for Blocks

$$\beta ::= b ; t$$

$$b ::= \cdot \mid \text{let } x = e ; b \mid \text{let } (x, y) = e ; b$$

$$t ::= \text{br } {}^{\wedge}\ell \; e \mid \text{if } e \; \{ s \} \text{ else } \{ t \}$$

$$\Gamma \vdash_p b : \Delta$$

$\uparrow$

Variables live on entry to b

Grammar for Blocks

$$\beta ::= b;\ t$$

$$b ::= \cdot \mid \text{let } x = e;\ b \mid \text{let } (x,y) = e;\ b$$

$$t ::= \text{br}\ {}^{\wedge}\ell\ e \mid \text{if } e\ \{S\} \text{ else } \{t\}$$

Variables live on exit from b

$$\Gamma \vdash_p b : \Delta$$

Variables live on entry to b

Grammar for Blocks

$$\beta ::= b ; t$$

$$b ::= \cdot \mid \text{let } x = e ; b \mid \text{let } (x, y) = e ; b$$

$$t ::= \text{br } {}^{\wedge}\ell \ e \mid \text{if } e \, \{ s \} \text{ else } \{ t \}$$

Purity of inst. in b

Variables live on exit from b

$$\Gamma \vdash_p b : \Delta$$

Variables live on entry to b

Grammar for Blocks

$$\beta ::= b ; t$$

$$b ::= \cdot \mid \text{let } x = e ; b \mid \text{let } (x, y) = e ; b$$

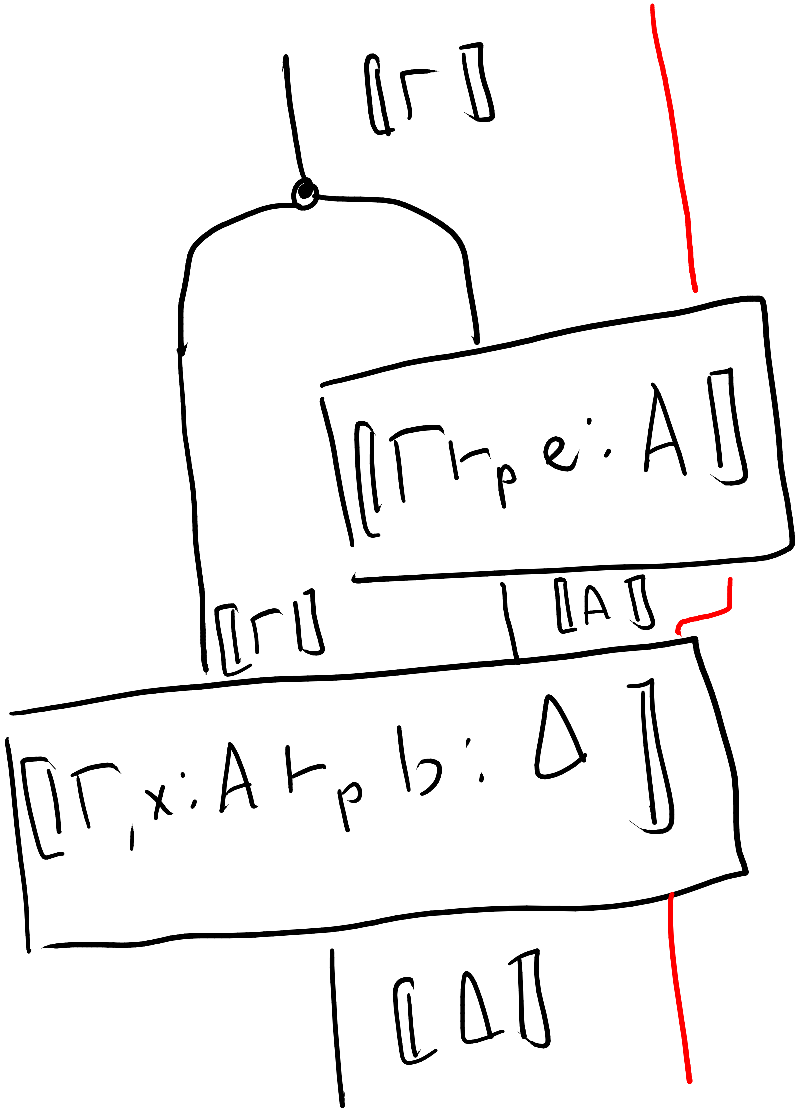
$$t ::= \text{br}^{\ell}\ e \mid \text{if } e \ \{ s \} \text{ else } \{ t \}$$

$$[\![\Gamma \vdash_p b : \Delta]\!] : \mathcal{C}_p([\![\Gamma]\!], [\![\Delta]\!])$$

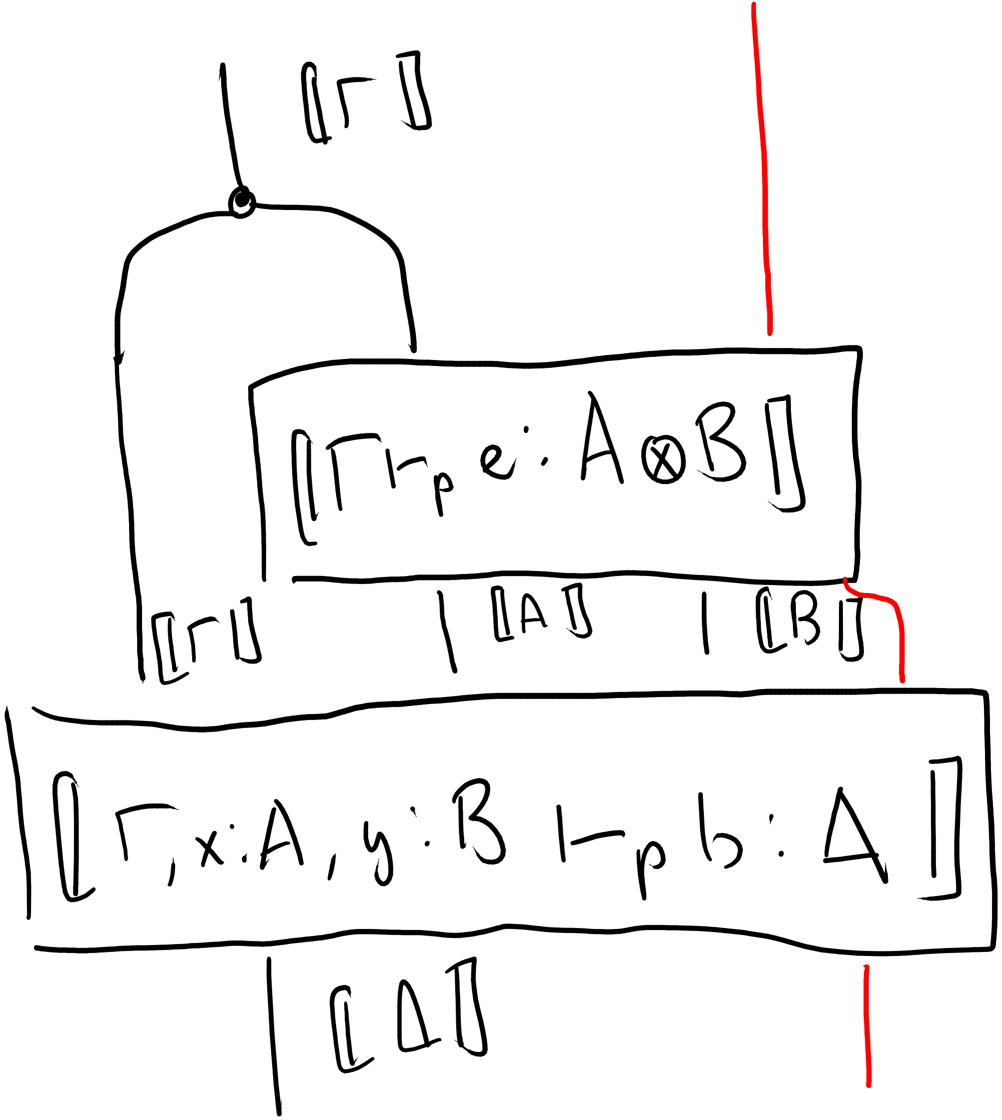
$$\left[\!\!\left[\frac{\Gamma \mapsto \Delta}{\Gamma \vdash_{p} \circ : \Delta} \right]\!\!\right] = [\![\Gamma \mapsto \Delta]\!]$$

$$\left[\!\left[\frac{\Gamma, x : A \vdash_p b : \Delta \qquad \Gamma \vdash_p e : A}{\Gamma \vdash_p \text{let } x = e;\ b : \Delta} \right]\!\right] =$$

$$\left[\!\!\left[\frac{\Gamma, x : A \vdash_p b : \Delta \qquad \Gamma \vdash_p e : A}{\Gamma \vdash_p \mathrm{let}\ x = e ; b : \Delta} \right]\!\!\right] =$$



$$\left[\!\!\left[\frac{\Gamma, x:A, y:B \vdash_p b:\Delta \qquad \Gamma \vdash_p e : A \otimes B}{\Gamma \vdash_p \mathrm{let}(x,y)=e; b : \Delta} \right]\!\!\right] =$$



Terminator Typing

$\beta ::= b; t$

$b ::= \cdot \mid \text{let } x = e; b \mid \text{let } (x, y) = e; b$

$t ::= \text{br } {'}\ell\ e \mid \text{if } e\ \{ s \} \text{ else } \{ t \}$

Terminator Typing

$$t ::= \mathrm{br}\ {}^{\wedge}\ell\ e \mid \mathrm{if}\ e\ \{s\}\ \mathrm{else}\ \{t\}$$

Terminator Typing

$$t ::= \mathrm{br}\ {}^{\wedge}\ell\ e \mid \mathrm{if}\ e\ \{s\}\ \mathrm{else}\ \{t\}$$

$$\Gamma \vdash t \triangleright L$$

Terminator Typing

$$t ::= \text{br}\ {}^\wedge\ell\ e \mid \text{if}\ e\ \{ s \}\ \text{else}\ \{ t \}$$

$$\Gamma \vdash t \rhd L$$

Variables
live on entry

Terminator Typing

$$t ::= \text{br}\ {}^\wedge\ell\ e \mid \text{if}\ e\ \{s\}\ \text{else}\ \{t\}$$

$$\Gamma \vdash t \rhd L$$

Variables live on entry

Targets branched to

Terminator Typing

$$t ::= \mathrm{br}\ {}^{\wedge}\ell\ e \mid \mathrm{if}\ e\ \{s\}\ \mathrm{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

$$\Gamma \vdash t \rhd L$$

Terminator Typing

$$t ::= \mathrm{br}\ {}^\wedge\ell\ e \mid \mathrm{if}\ e\ \{s\}\ \mathrm{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^\wedge\ell[\Delta](A)$$

↑ Label branched to

$$\Gamma \vdash t \rhd L$$

Terminator Typing

$$t ::= \text{br}\ {}^\wedge\ell\ e \mid \text{if}\ e\ \{s\}\ \text{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^\wedge\ell[\Delta](A)$$

Label branched to

Variables live on branch

$$\Gamma \vdash t \rhd L$$

Terminator Typing

$$t ::= \mathrm{br}\ {}^\wedge\ell\ e \mid \mathrm{if}\ e\ \{s\}\ \mathrm{else}\ \{t\}$$

Block argument

$$L ::= \cdot \mid L, {}^\wedge\ell[\Delta](A)$$

Label branched to

Variables live on branch

$$\Gamma \vdash t \rhd L$$

Terminator Typing

$$t ::= \mathsf{br}\ {}^{\wedge}\ell\ e \mid \mathsf{if}\ e\ \{s\}\ \mathsf{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

Block argument (multiple args w/ $A \otimes B$)

Label branched to

Variables live on branch

$$\Gamma \vdash t \rhd L$$

Terminator Typing

$$t ::= \mathrm{br}\ {}^{\wedge}\ell\ e \mid \mathrm{ife}\ \{s\}\ \mathrm{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

$$[\![\Gamma \vdash t \rhd L]\!] : C_1([\![\Gamma]\!], [\![L]\!])$$

Terminator Typing

$$t ::= \mathsf{br}\ {}^{\wedge}\ell\ e \mid \mathsf{if}\ e\ \{s\}\ \mathsf{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

$$[\![\Gamma \vdash t \rhd L]\!] : C_{\textcircled{1}}([\![\Gamma]\!], [\![L]\!])$$

Always consider terminators PURE

Terminator Typing

$$t ::= \mathrm{br}\ {}^{\wedge}\ell\ e \mid \mathrm{if}\ e\ \{s\}\ \mathrm{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

$$[\![\Gamma \vdash t \rhd L]\!] : C_1([\![\Gamma]\!], \overset{?}{[\![L]\!]})$$

Terminator Typing

$$t ::= \mathrm{br}\ {}^{\wedge}\ell\ e \mid \mathrm{if}\ e\ \{s\}\ \mathrm{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

$$[\![\Gamma \vdash t \rhd L]\!] : C_1([\![\Gamma]\!], [\![L]\!])$$

$$[\![\cdot]\!] = 0$$

Terminator Typing

$$t ::= \mathrm{br}\ {}^{\wedge}\ell\ e \mid \mathrm{ife}\ \{s\}\ \mathrm{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

$$[\![\Gamma \vdash t \rhd L]\!] : C_{\underline{1}}([\![\Gamma]\!], [\![L]\!])$$

$$[\![\cdot]\!] = 0 \qquad [\![L, {}^{\wedge}\ell[\Delta](A)]\!] = [\![L]\!] + ([\![\Delta]\!] \otimes [\![A]\!])$$

Terminator Typing

$$t ::= \mathrm{br}\ {}^{\wedge}\ell\ e \mid \mathrm{ife}\ \{s\}\ \mathrm{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

$$[\![\Gamma \vdash t \triangleright L]\!] : C_1([\![\Gamma]\!], [\![L]\!])$$

$$[\![\cdot]\!] = 0 \qquad [\![L, {}^{\wedge}\ell[\Delta](A)]\!] = \overset{\text{This}\searrow}{[\![L]\!]} + ([\![\Delta]\!] \otimes [\![A]\!])$$

Terminator Typing

$$t ::= \mathsf{br}\ {}^{\wedge}\ell\ e \mid \mathsf{if}\ e\ \{s\}\ \mathsf{else}\ \{t\}$$

$$L ::= \cdot \mid L, {}^{\wedge}\ell[\Delta](A)$$

$$[\![\Gamma \vdash t \rhd L]\!] : C_1([\![\Gamma]\!], [\![L]\!])$$

This   OR this

$$[\![\cdot]\!] = 0 \qquad [\![L, {}^{\wedge}\ell[\Delta](A)]\!] = [\![L]\!] + ([\![\Delta]\!] \otimes [\![A]\!])$$

Coproducts

Coproducts

Recall $f : C(A,B),\ g : C(A,C)$

$\Rightarrow \langle f, g \rangle : C(A, B \otimes C)$

Coproducts

Recall $f : C(A,B),\ g : C(A,C)$

$\Rightarrow \langle f, g \rangle : C(A, B \otimes C)$

Try: $f : C(B,A),\ g\,(C,A)$

$\Rightarrow [f,g] : C(B+C, A)$

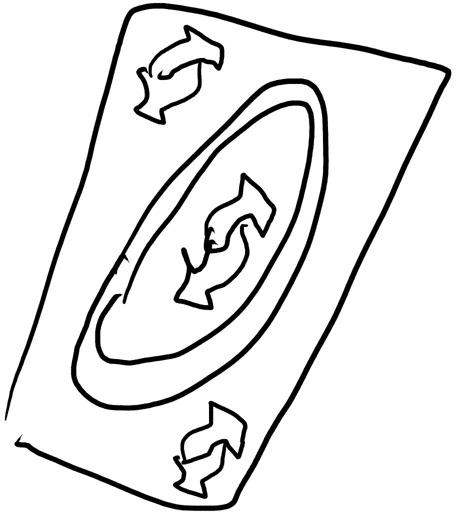
Coproducts

Recall $f : C(A,B),\ g : C(A,C)$

$\Rightarrow \langle f,g \rangle : C(A, B \otimes C)$

Try: $f : C(B,A),\ g\,(C,A)$

$\Rightarrow [f,g] : C(B+C,\ A)$



Coproducts

Need: $[f, g]$ $\qquad 0$ (w/ $0_A : C(0, A)$)

Coproducts

Need: $[f, g]$ $\quad 0$ (w/ $0_A : C(0, A)$)

$inl : C(A, A+B)$ $\quad inr : C(B, A+B)$

Coproducts

Need: $[f, g]$ $\quad 0$ (w/ $0_A : C(0, A)$)

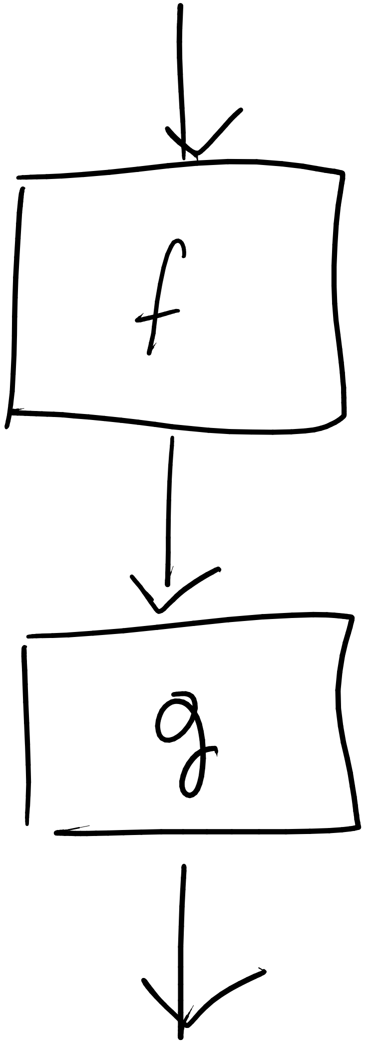
$\mathrm{inl} : C(A, A+B)$ $\quad \mathrm{inr} : C(B, A+B)$

Can def'n, e.g. $f : C(A, B)$ $\quad g : C(A', B')$

$$f + g = [\mathrm{inl} \circ f, \mathrm{inr} \circ g] : C(A+A', B+B')$$

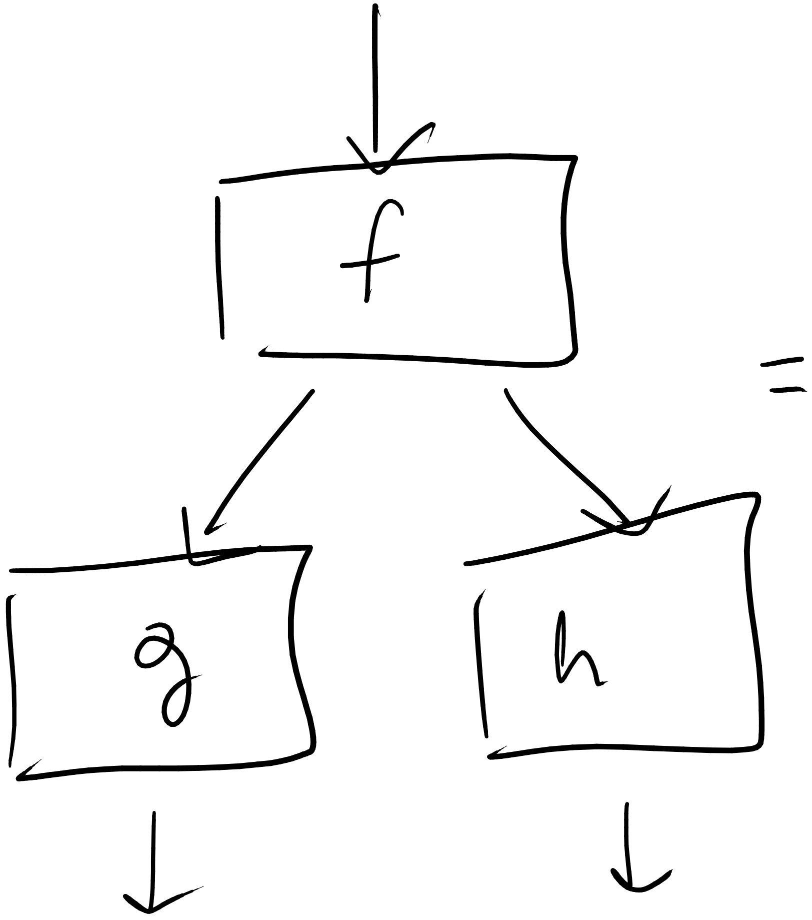
$\dots$

Drawing CFGs



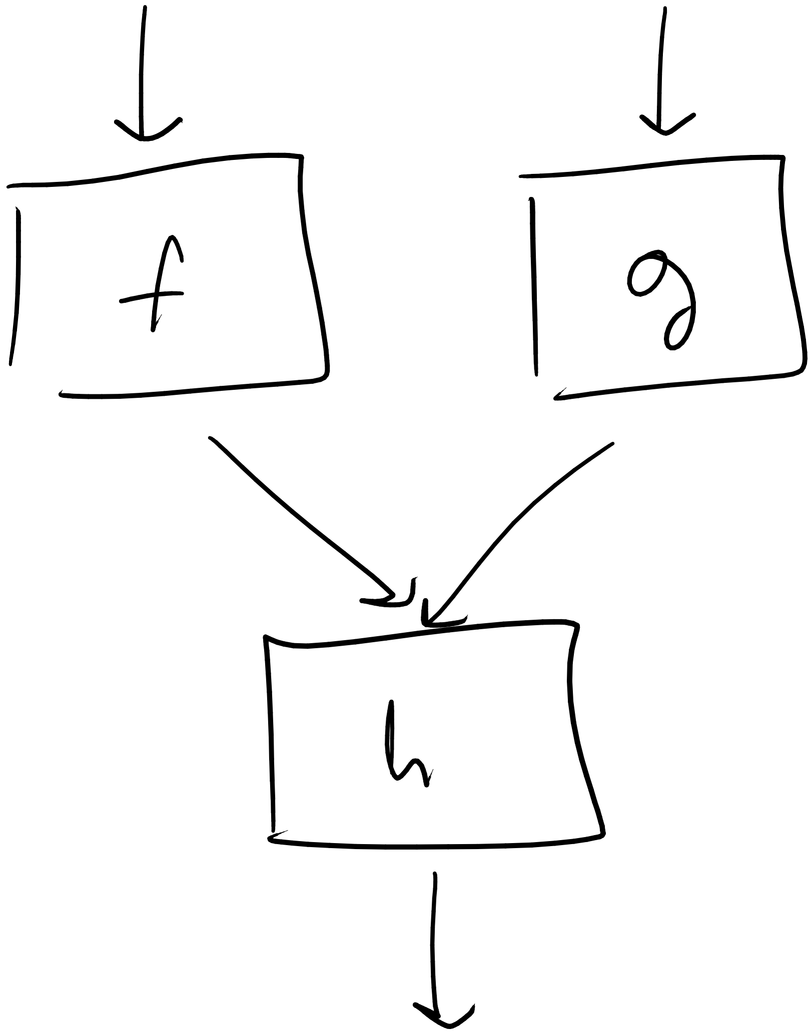
= f ; g

Drawing CFGs



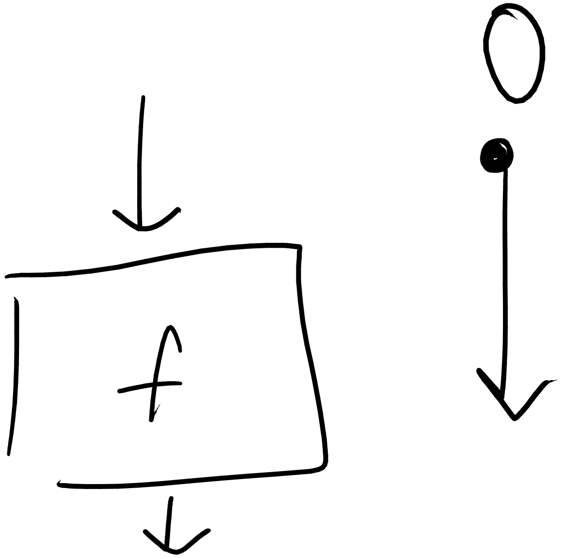
= f ; g+h

Drawing CFGs



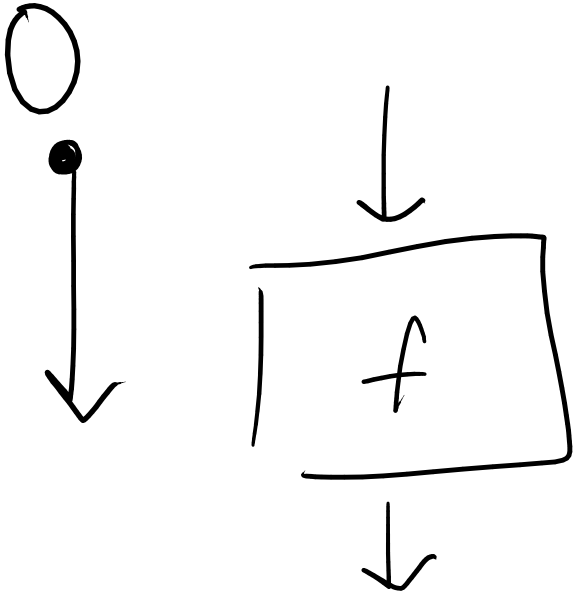
$= [f, g] ; h$

Drawing CFGs



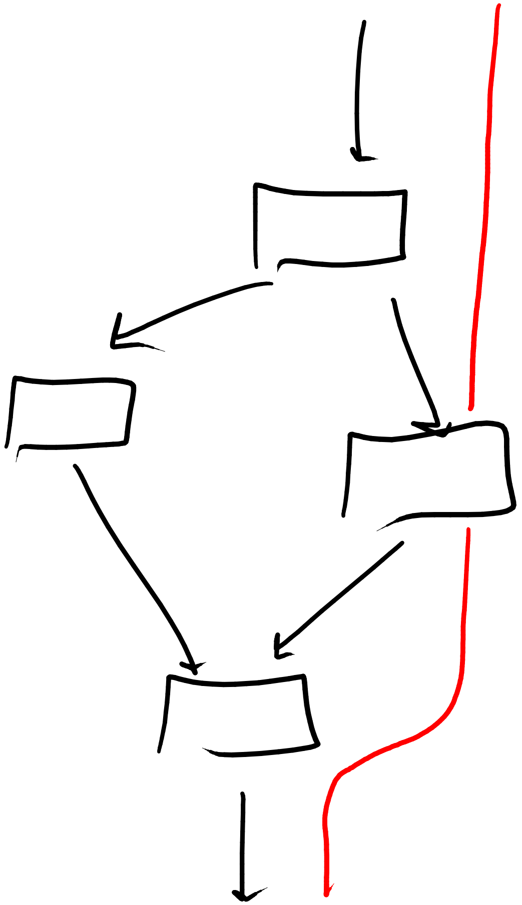
= inl f

Drawing CFGs

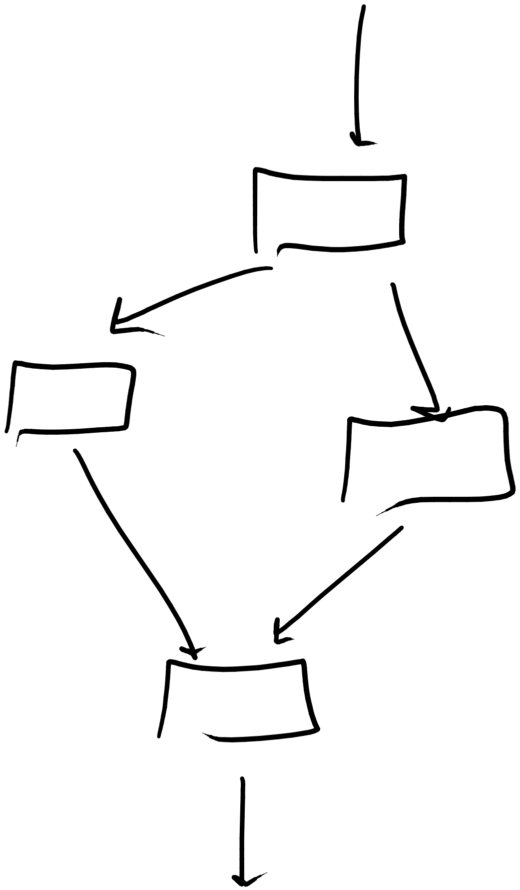


$= \mathrm{inr}\ f$

Drawing CFGs



DATAFLOW (⊗)



CONTROL FLOW (+)

Unconditional Branch Semantics

$$\left\llbracket \frac{\Gamma \vdash_1 e : A \qquad {}^{\wedge}\ell\,[\Gamma](A) \rightsquigarrow L}{\Gamma \vdash \mathrm{br}\ {}^{\wedge}\ell\ e \triangleright L} \right\rrbracket$$

Unconditional Branch Semantics

Can only pass pure expr (e.g var, const, arith)

$$\left\llbracket \frac{\Gamma \vdash_{\textcircled{1}} e : A \qquad {}^{\ell}[\Gamma](A) \rightsquigarrow L}{\Gamma \vdash \mathrm{br}\ {}^{\ell}\ e \rhd L} \right\rrbracket$$

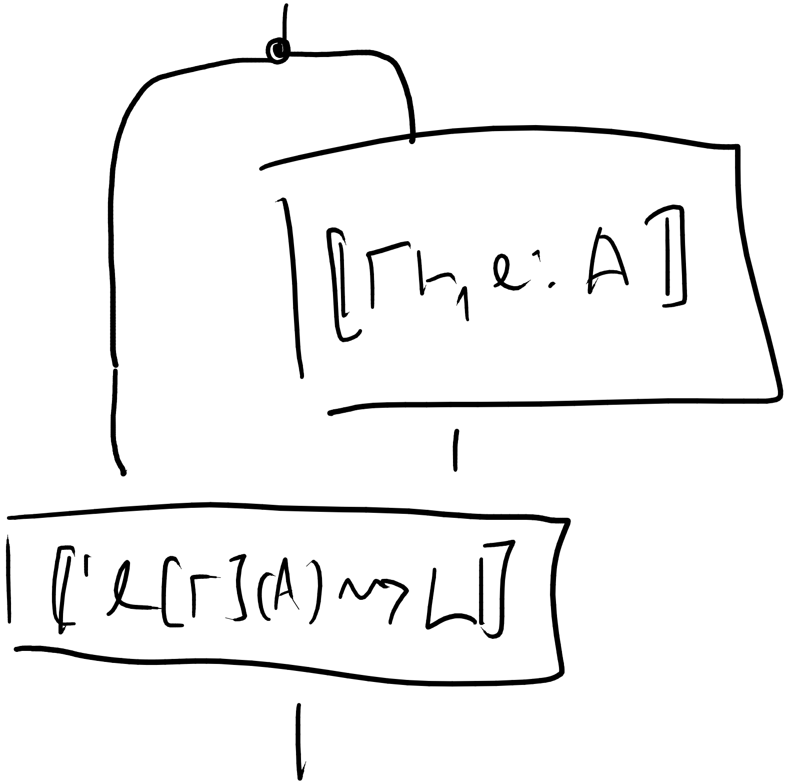
Unconditional Branch Semantics

$$\left[\!\left[\frac{\Gamma \vdash_1 e : A \qquad \boxed{{}^\ell[\Gamma](A) \rightsquigarrow L}}{\Gamma \vdash \mathrm{br}\ {}^\ell\ e \triangleright L} \right]\!\right]$$

Like $\Gamma \mapsto \Delta$ but backwards, for contexts

Unconditional Branch Semantics

$$\left[\!\left[\frac{\Gamma \vdash_1 e : A \qquad {}^{\prime}\ell[\Gamma](A) \rightsquigarrow L}{\Gamma \vdash \mathrm{br}\ {}^{\prime}\ell\ e \rhd L} \right]\!\right] =$$



Label Weakening

$$[\![L \rightsquigarrow K]\!] : C_1([\![L]\!], [\![K]\!])$$

Label Weakening

$$\llbracket L \rightsquigarrow K \rrbracket : C_1(\llbracket L \rrbracket, \llbracket K \rrbracket)$$

K has more labels than L

Label Weakening

$$[\![L \rightsquigarrow K]\!] : \mathcal{C}_1([\![L]\!], [\![K]\!])$$

K has more labels than L

e.g. $\,^{\ell_1}[\Gamma](A), \,^{\ell_2}[\Delta](B) \rightsquigarrow$

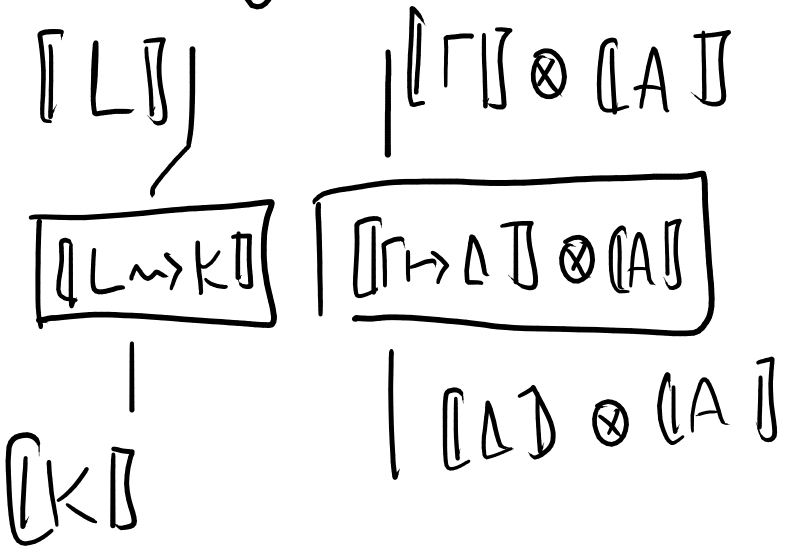
$\,^{\ell_1}[\Gamma](A), \,^{\ell_2}[\Delta](B), \,^{\ell_3}[\Xi](C)$

Label Weakening

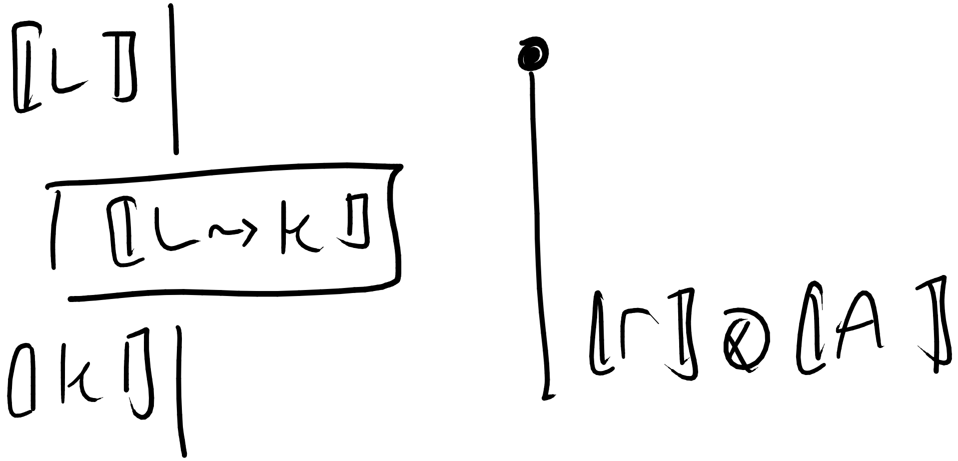
$$\left[\!\left[\frac{}{\cdot \rightsquigarrow \cdot} \right]\!\right] = \mathrm{id}_0$$

Label Weakening

$$\left[\!\left[\frac{L \rightsquigarrow K \qquad \Gamma \mapsto \Delta}{L, {}^{\wedge}\ell[\Gamma](A) \rightsquigarrow K, {}^{\wedge}\ell[\Delta](A)} \right]\!\right] =$$

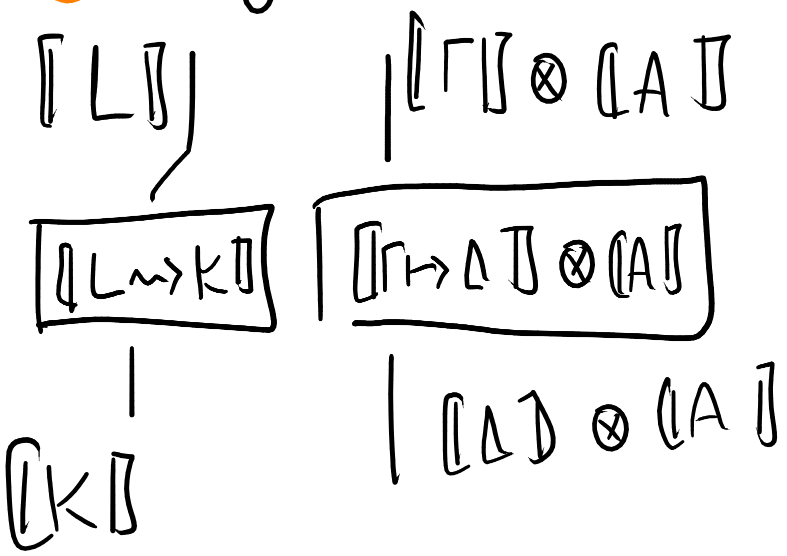


$$\left[\!\left[\frac{L \rightsquigarrow K}{L \rightsquigarrow K, {}^{\wedge}\ell[\Gamma](A)} \right]\!\right] =$$



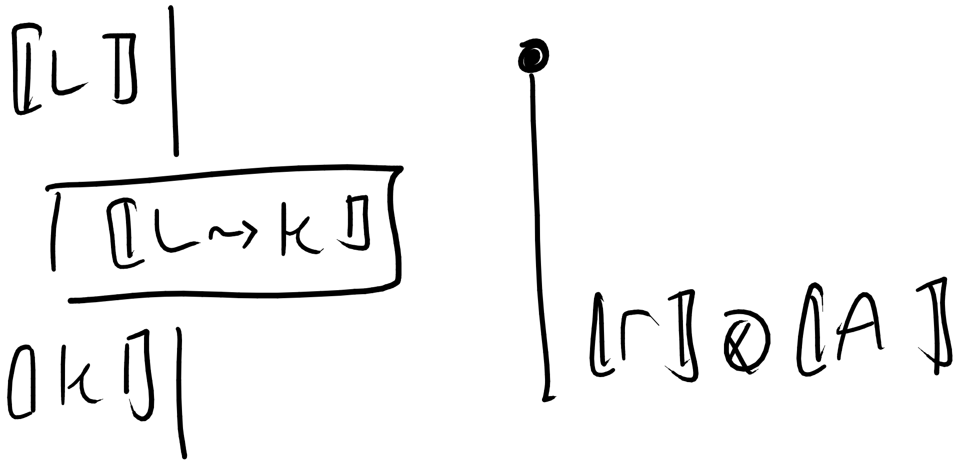
Label Weakening

$$\left[\!\left[\frac{L \rightsquigarrow K \qquad \Gamma \mapsto \Delta}{L, {}^{\wedge}\ell[\Gamma](A) \rightsquigarrow K, {}^{\wedge}\ell[\Delta](A)}\right]\!\right] =$$



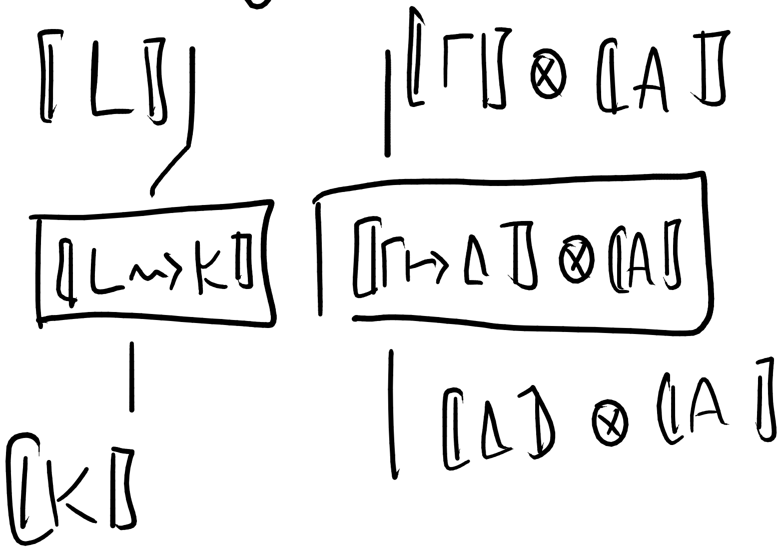
$$\left[\!\left[\frac{L \rightsquigarrow K}{L \rightsquigarrow K, {}^{\wedge}\ell[\Gamma](A)}\right]\!\right] =$$

↑ MORE labels



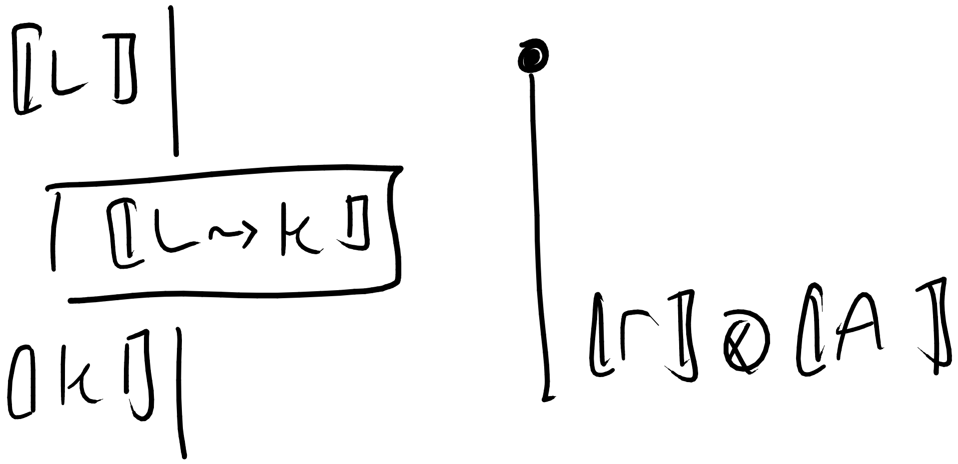
Label Weakening

$$\left\llbracket \frac{L \rightsquigarrow K \qquad \Gamma \mapsto \Delta}{L, {}^{\wedge}\ell\llbracket\Gamma\rrbracket(A) \rightsquigarrow K, {}^{\wedge}\ell\llbracket\Delta\rrbracket(A)} \right\rrbracket =$$



Less variables

$$\left\llbracket \frac{L \rightsquigarrow K}{L \rightsquigarrow K, {}^{\wedge}\ell\llbracket\Gamma\rrbracket(A)} \right\rrbracket =$$



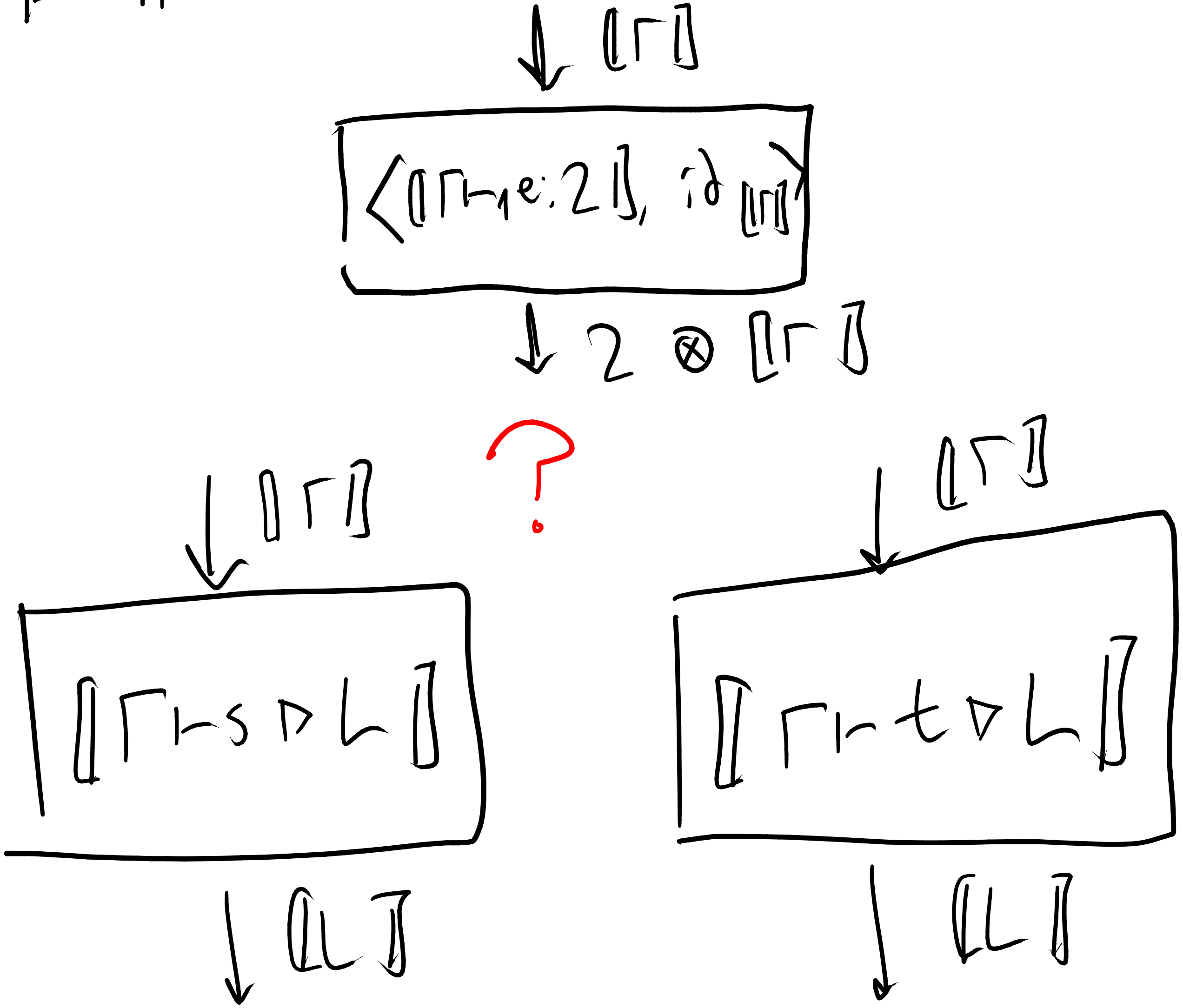
^ MORE labels

Conditional Branch Semantics

Conditional Branch Semantics

$$\frac{\Gamma \vdash_1 e : 2 \qquad \Gamma \vdash s \triangleright L \qquad \Gamma \vdash t \triangleright L}{\Gamma \vdash \text{if } e \ \{s\} \text{ else } \{t\} \triangleright L}$$

$$\left[\!\left[\frac{\Gamma \vdash_1 e : 2 \qquad \Gamma \vdash s \triangleright L \qquad \Gamma \vdash t \triangleright L}{\Gamma \vdash \text{if } e \; \{s\} \text{ else } \{t\} \triangleright L} \right]\!\right] =$$



$$\delta : A \otimes (B + C) \simeq A \otimes B + A \otimes C$$

$$\delta : A \otimes (B + C) \simeq A \otimes B + A \otimes C$$

Given $[\![2]\!] = I + I$,

$$[\![2]\!] \otimes [\![\Gamma]\!] \simeq I \otimes [\![\Gamma]\!] + I \otimes [\![\Gamma]\!]$$

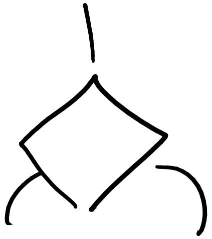
$$\simeq [\![\Gamma]\!] + [\![\Gamma]\!] .$$

$$\delta : A \otimes (B+C) \simeq A \otimes B + A \otimes C$$

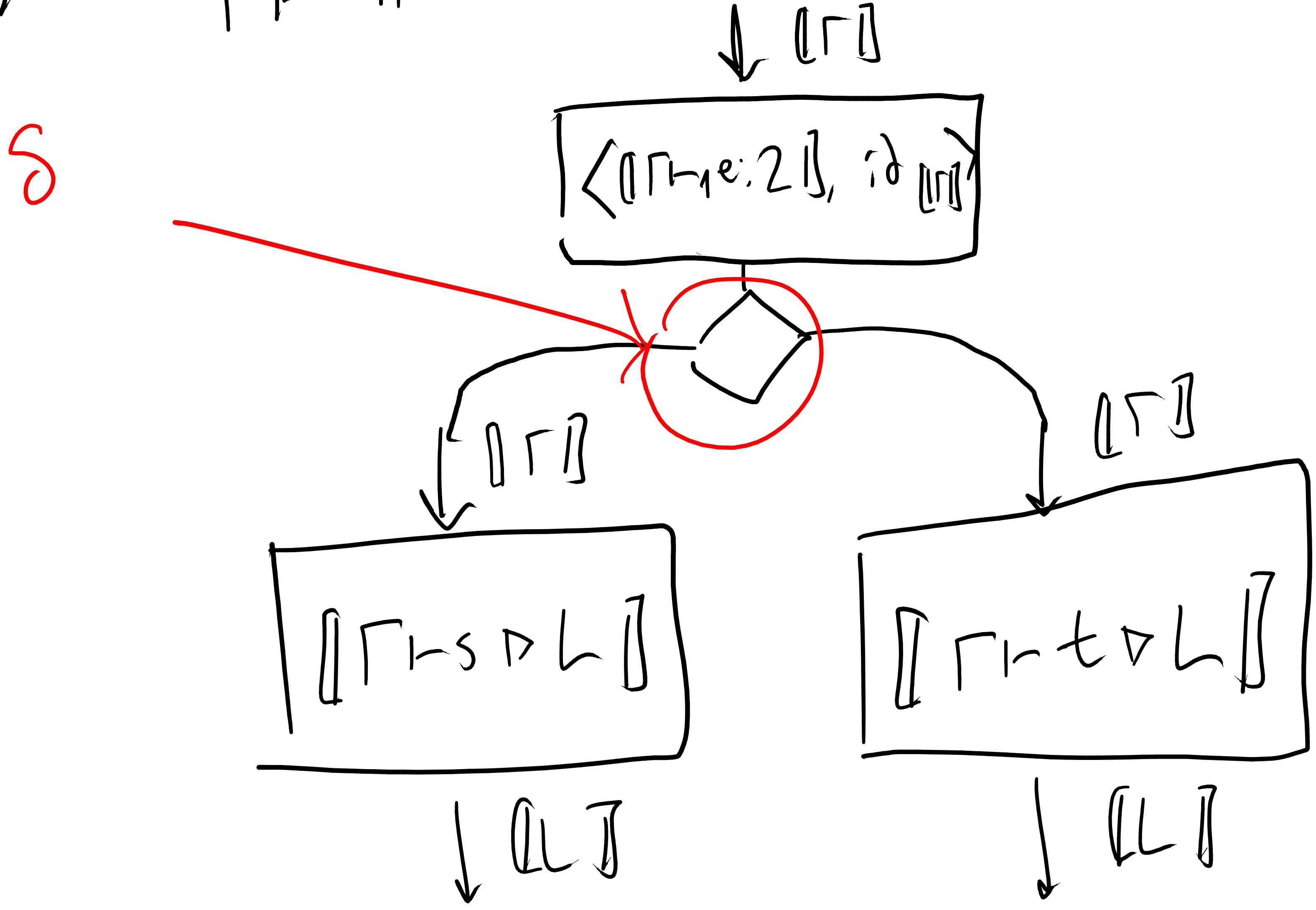
Given $[\![2]\!] = I + I$,

$$[\![2]\!] \otimes [\![\Gamma]\!] \simeq I \otimes [\![\Gamma]\!] + I \otimes [\![\Gamma]\!]$$

$$\simeq [\![\Gamma]\!] + [\![\Gamma]\!] .$$

Draw as 

$$\left[\!\left[\frac{\Gamma \vdash_1 e : 2 \qquad \Gamma \vdash s \triangleright L \qquad \Gamma \vdash t \triangleright L}{\Gamma \vdash \text{if } e \ \{s\} \text{ else } \{t\} \triangleright L} \right]\!\right] =$$



Function ⟺ Structure

Multiple args / Tuples ⟺ Tensor Product

Purity ⟺ Freyd Category

Branching Control Flow ⟺ Coproducts

Conditional Branches ⟺ Distributivity

Basic Block Semantics

$$[\![\Gamma \vdash \beta \triangleright L]\!] : C_0([\![\Gamma]\!], [\![L]\!])$$

Basic Block Semantics

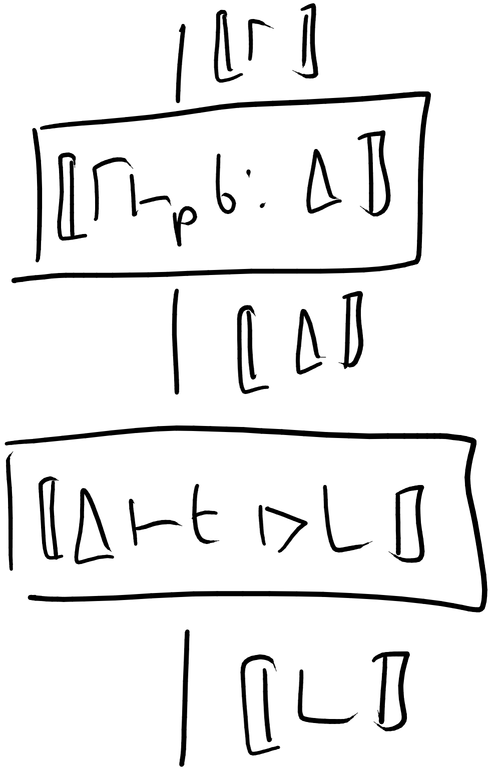
$$[\![\Gamma \vdash \beta \rhd L]\!] : \mathcal{C}_{\partial}([\![\Gamma]\!], [\![L]\!])$$

↑ Always treat as IMPURE

Basic Block Semantics

$$[\![\Gamma \vdash \beta \triangleright L]\!] : \mathcal{C}_0([\![\Gamma]\!], [\![L]\!])$$

$$\left[\!\!\left[\frac{\Gamma \vdash_p b : \Delta \qquad \Delta \vdash t \triangleright L}{\Gamma \vdash b; t \triangleright L}\right]\!\!\right] =$$



~~Instructions~~

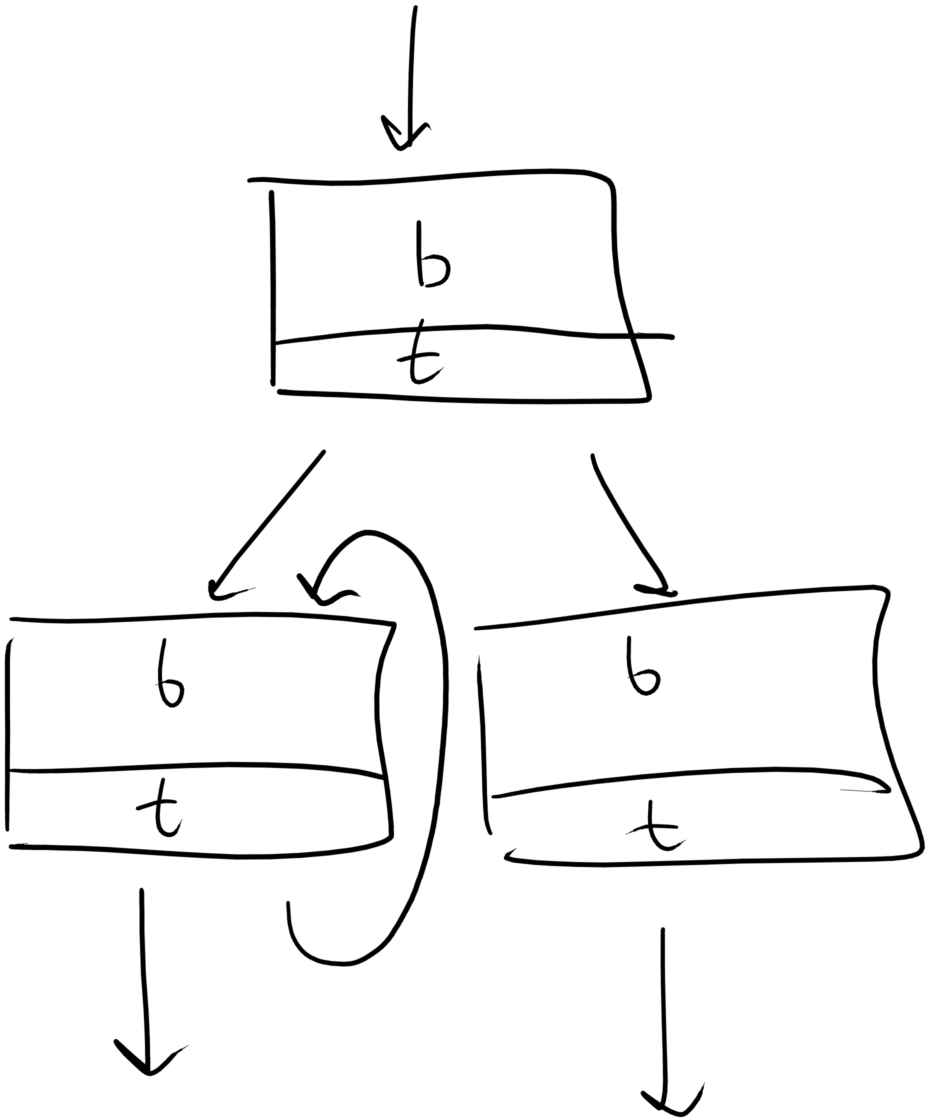
~~Blocks~~

Regions

~~Instructions~~

~~Blocks~~

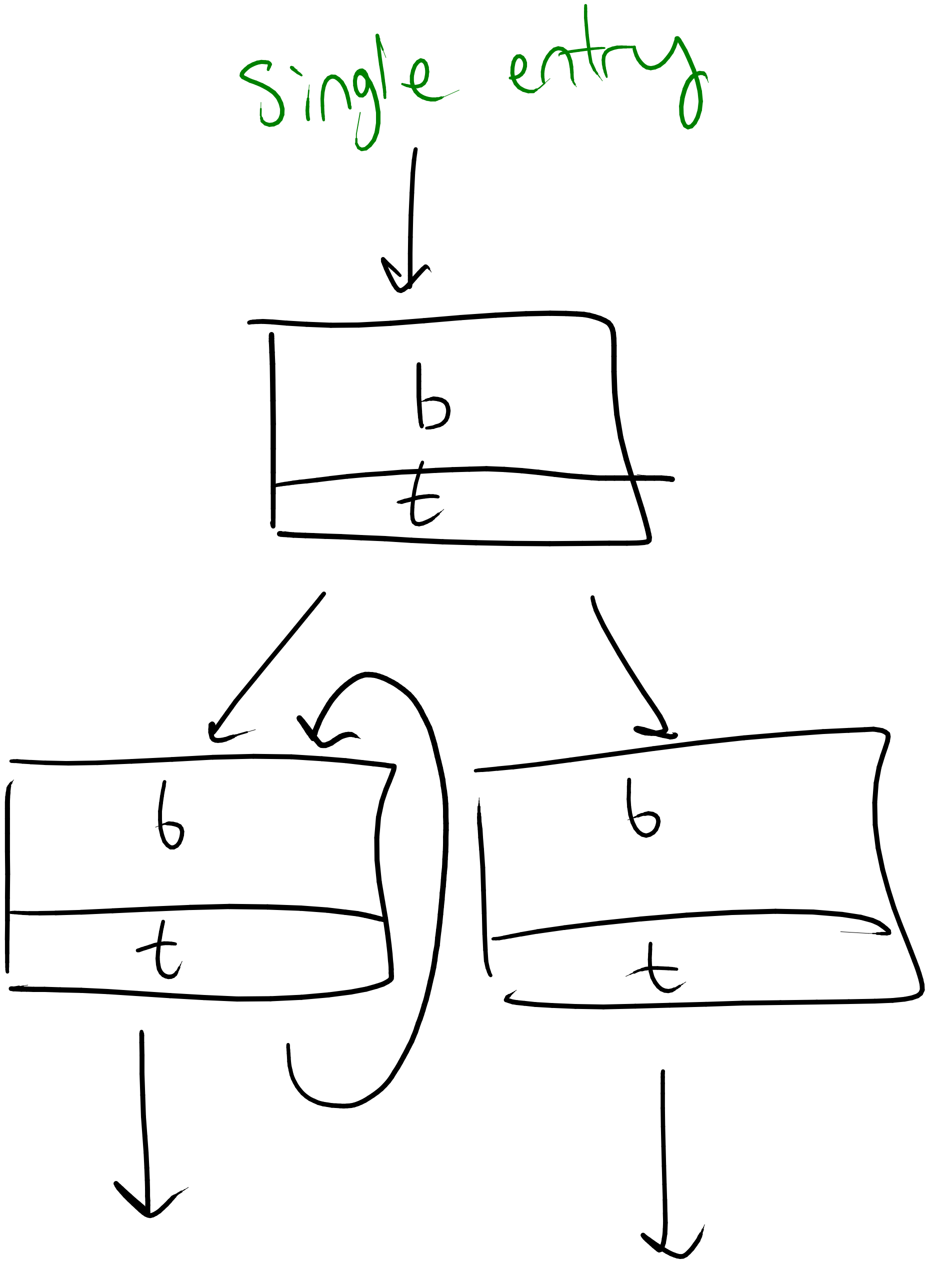
Regions



~~Instructions~~

~~Blocks~~

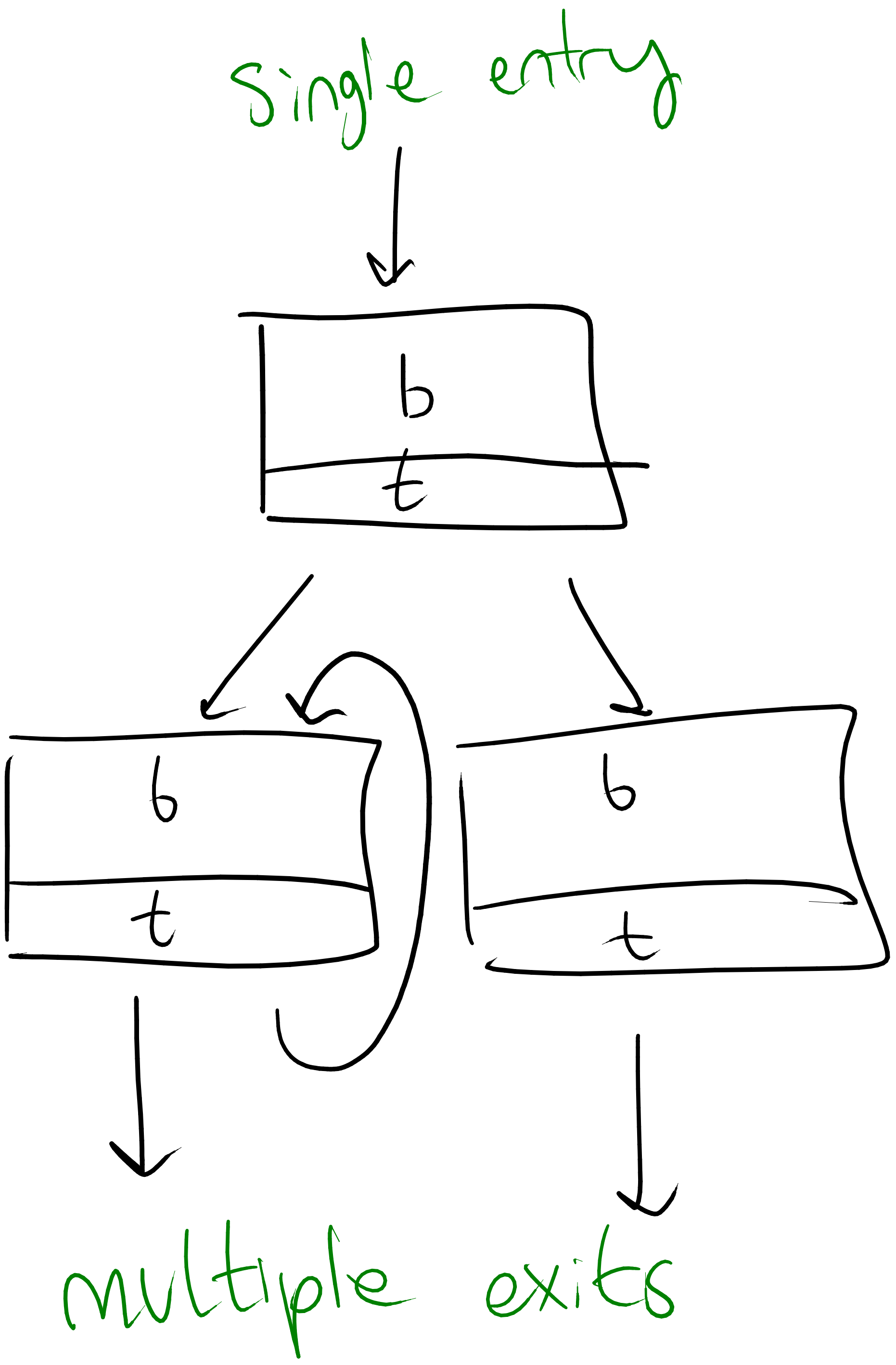
Regions



~~Instructions~~

~~Blocks~~

Regions



Grammar for Regions

$R ::= \beta$ where $\mathcal{L}$

Grammar for Regions

$$R ::= \underset{\uparrow}{\beta} \text{ where } \mathcal{L}$$

entry block

(cannot be
branched to
from inside region)

Grammar for Regions

$$R ::= \beta \text{ where } \mathcal{L}$$

entry block

(cannot be branched to from inside region)

control-flow graph

Grammar for Regions

$$\mathcal{R} ::= \beta \text{ where } \mathcal{L}$$

$$\mathcal{L} ::= \cdot \mid \mathcal{L}, \hat{\ } \ell(x : A) : \beta$$

Grammar for Regions

$$\mathcal{R} ::= \beta \text{ where } \mathcal{L}$$

$$\mathcal{L} ::= \cdot \mid \mathcal{L},\ {}^{\wedge}\underset{\substack{\uparrow \\ \text{Label}}}{\ell}(x : A) : \beta$$

Grammar for Regions

$$\mathcal{R} ::= \beta \text{ where } \mathcal{L}$$

$$\mathcal{L} ::= \cdot \mid \mathcal{L},\ {}^{\wedge}\underset{\substack{\uparrow\\ \text{Label}}}{\ell}(x : \underset{\substack{\uparrow\\ \text{Argument}}}{A}) : \beta$$

Grammar for Regions

$$\mathcal{R} ::= \beta \text{ where } \mathcal{L}$$

$$\mathcal{L} ::= \cdot \mid \mathcal{L},\ \underset{\underset{\text{Label}}{\uparrow}}{{}^\wedge\ell}\ \underset{\underset{\text{Argument}}{\uparrow}}{(x : A)} : \underset{\underset{\text{Block}}{\uparrow}}{\beta}$$

Grammar for Regions

$$\mathcal{R} ::= \beta \text{ where } \mathcal{L}$$

$$\mathcal{L} ::= \cdot \mid \mathcal{L}, {}^{\wedge}\ell\,(x : A) : \beta$$

$$[\![\Gamma \vdash \mathcal{R} \rhd L]\!]$$

Grammar for Regions

$$\mathcal{R} ::= \beta \ \text{where} \ \mathcal{L}$$

$$\mathcal{L} ::= \cdot \mid \mathcal{L}, {}^\wedge\ell(x : A) : \beta$$

$$[\![\Gamma \vdash \mathcal{R} \rhd L]\!] \longrightarrow \text{Targets}$$

Grammar for Regions

$$\mathcal{R} ::= \beta \text{ where } \mathcal{L}$$

$$\mathcal{L} ::= \cdot \mid \mathcal{L}, {}^{\wedge}\ell(x : A) : \beta$$

$$\text{Params} \rightarrow [\![\Gamma \vdash \mathcal{R} \rhd L]\!] \rightarrow \text{Targets}$$

Grammar for Regions

$$\mathcal{R} ::= \beta \quad \text{where} \quad \mathcal{L}$$

$$\mathcal{L} ::= \cdot \mid \mathcal{L}, {}^{\wedge}\ell\,(x : A) : \beta$$

$$\text{Params} \rightarrow \llbracket \Gamma \vdash \mathcal{R} \rhd L \rrbracket \rightarrow \text{Targets}$$

$$: C_0(\llbracket \Gamma \rrbracket, \llbracket L \rrbracket)$$

CFG semantics

$$L \vdash \mathcal{L} \triangleright K$$

CFG Semantics

$$\underset{\text{entry points}}{\longrightarrow} L \vdash \mathcal{L} \rhd K \underset{\text{exit points}}{\longrightarrow}$$

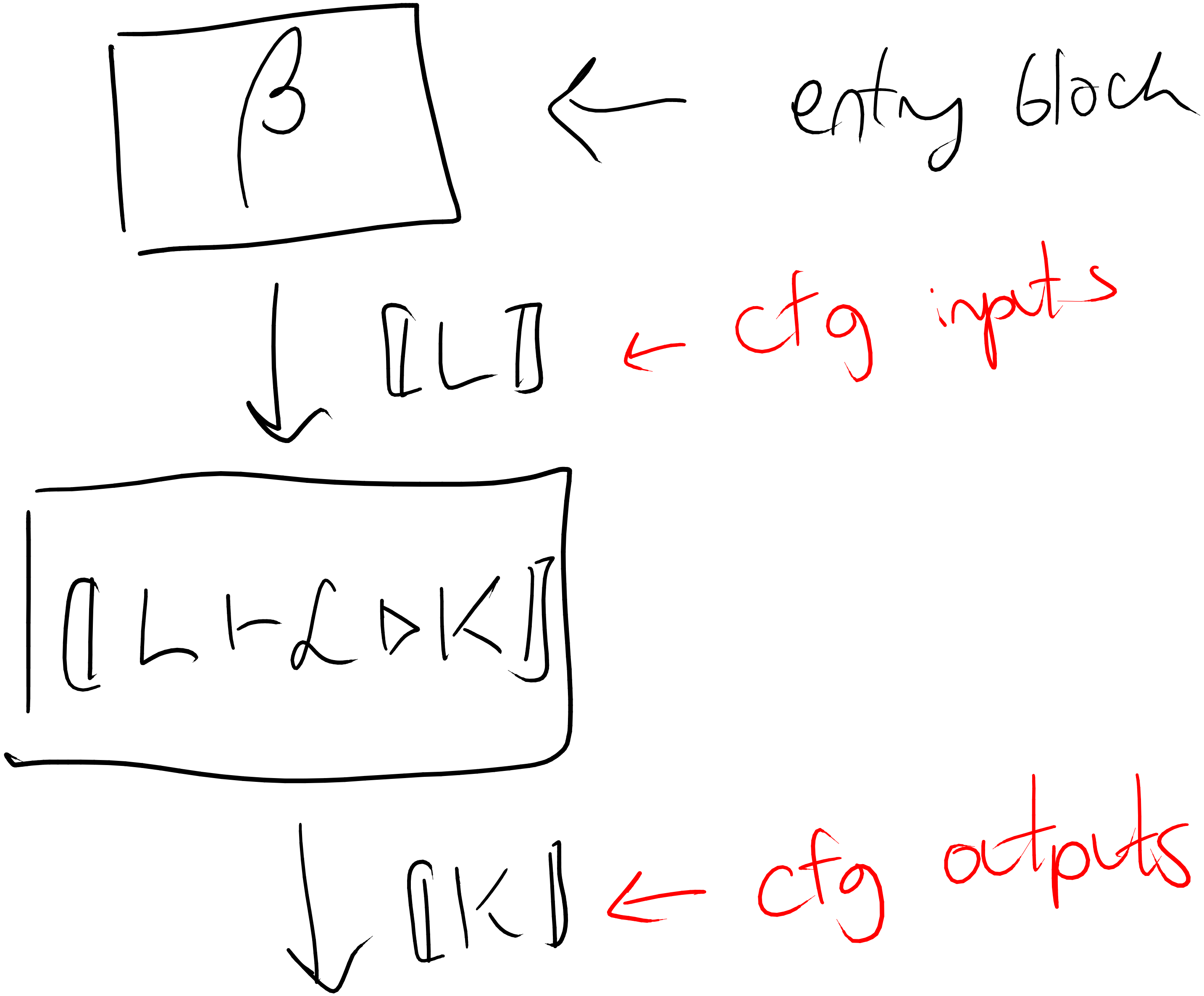
CFG Semantics : Take I

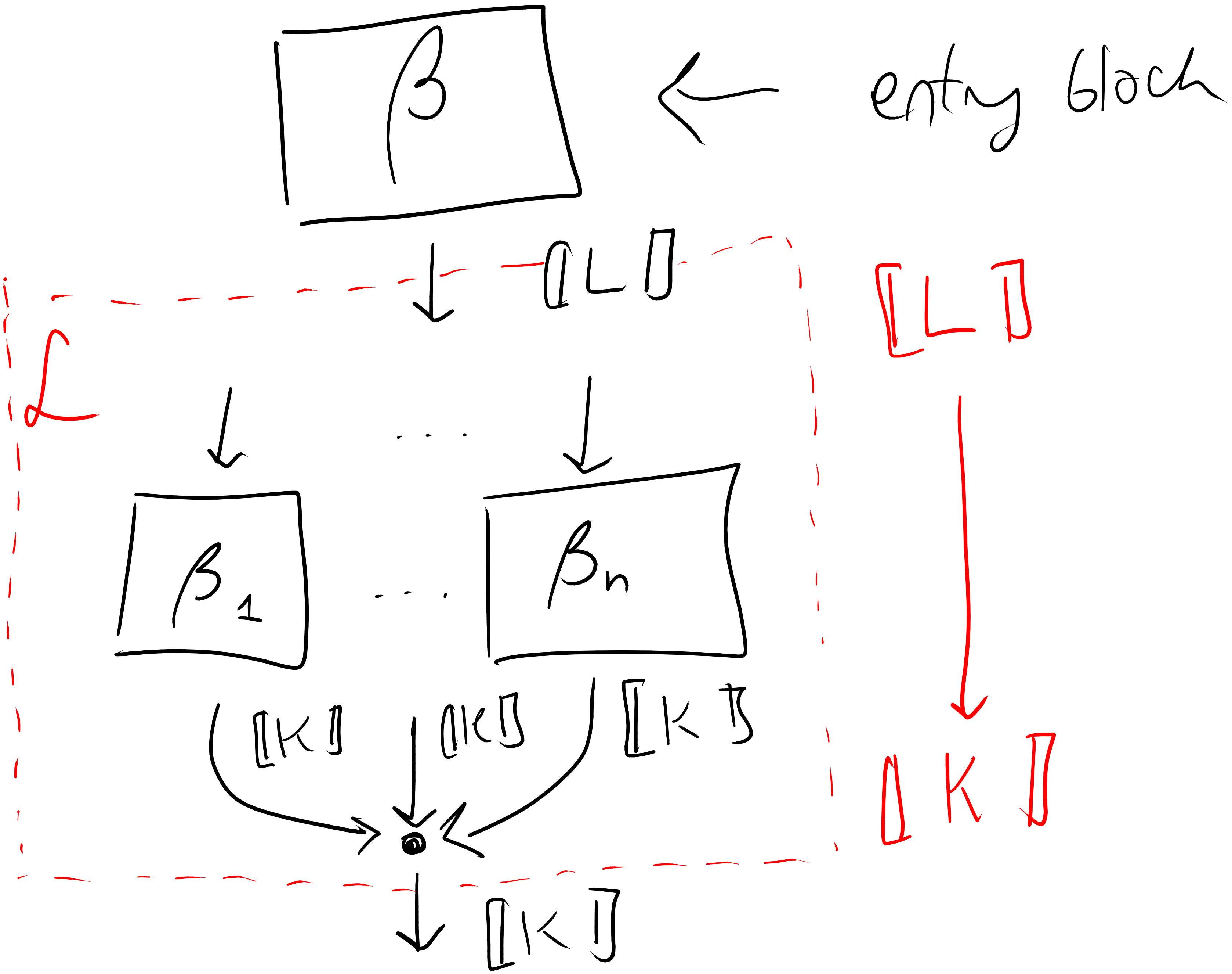
$$\rightarrow [\![L \vdash \mathcal{L} \rhd K]\!] \rightarrow$$

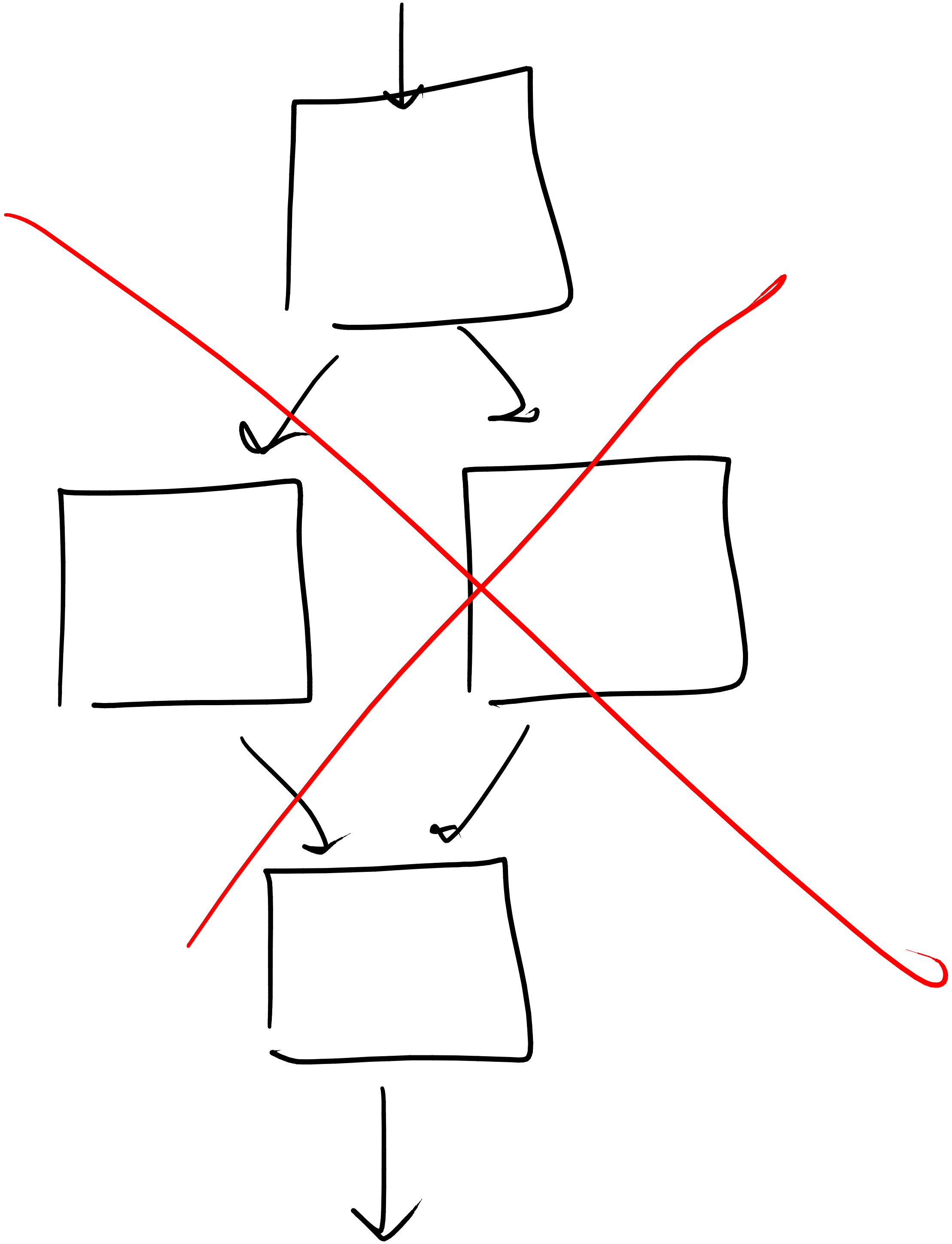
entry points

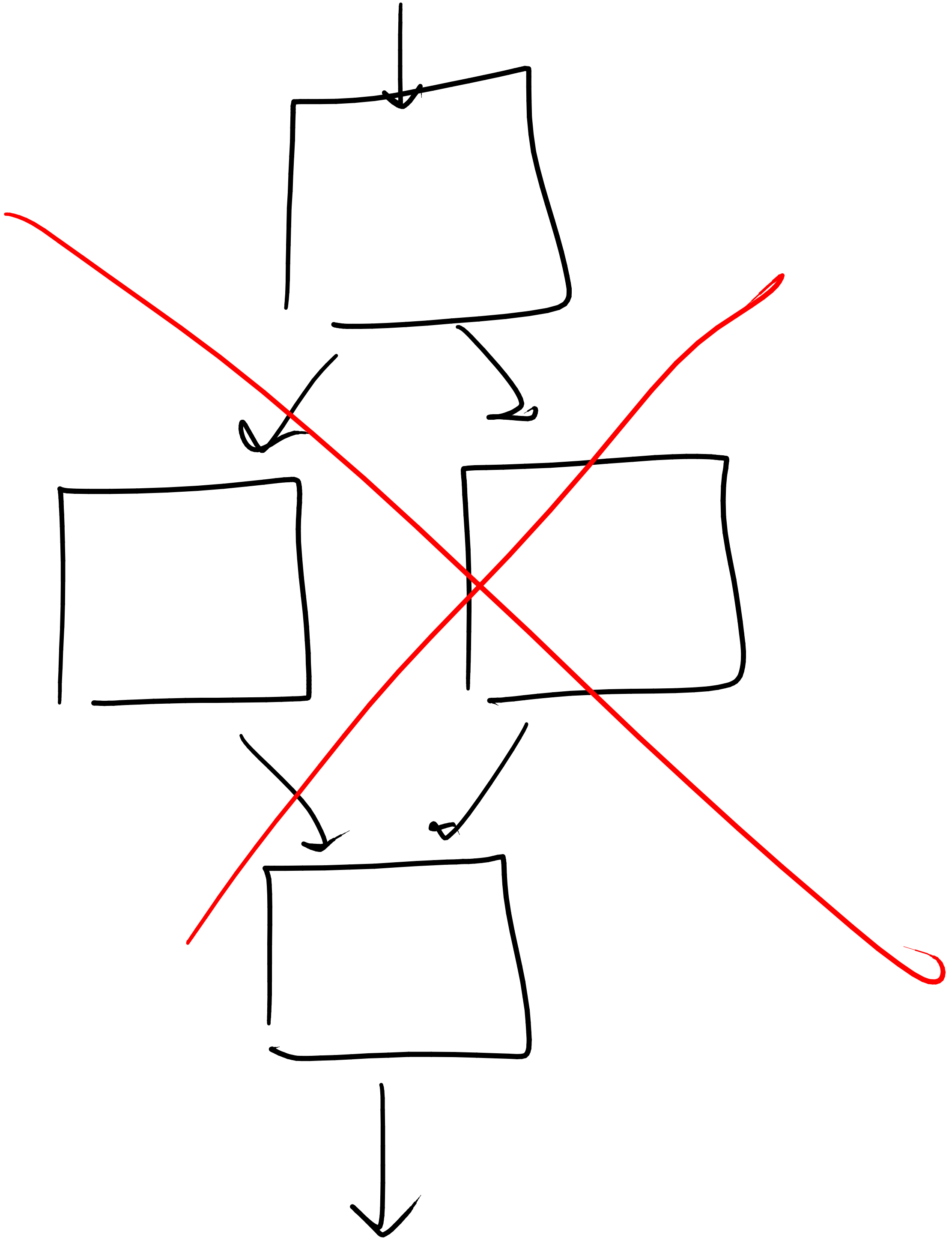
exit points

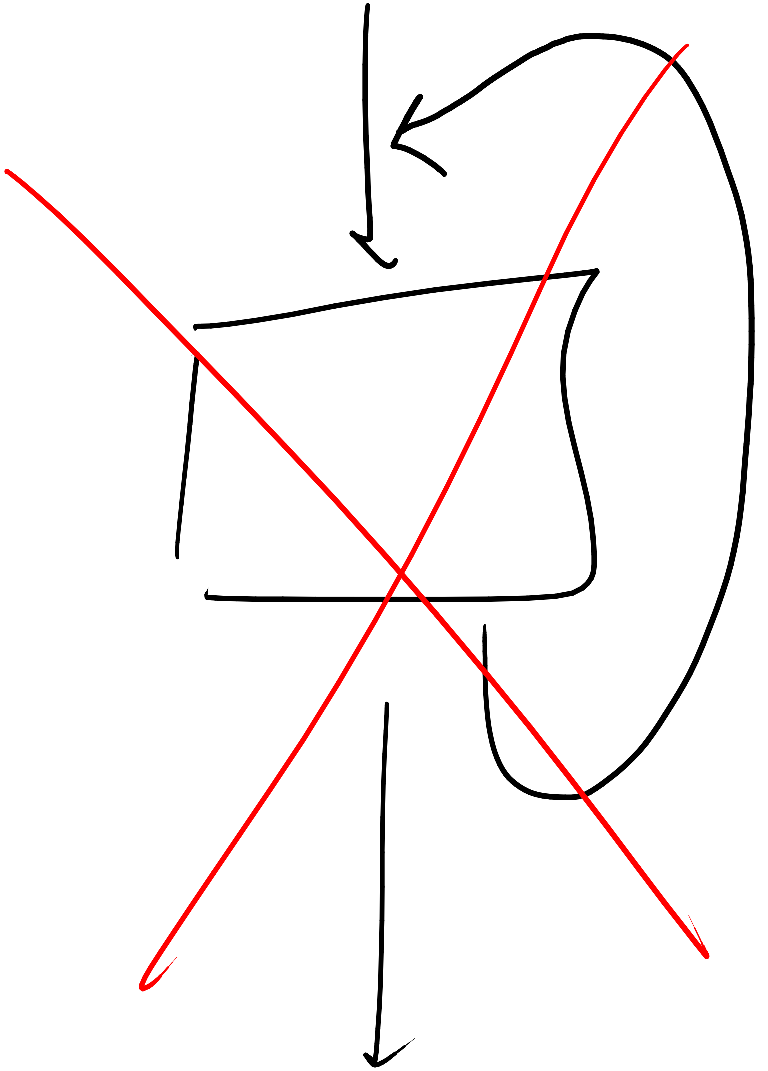
$$: C_0([\![L]\!], [\![K]\!])$$


β
entry block
⟦L⟧
cfg inputs
⟦L ⊢ ℒ ▷ K⟧
⟦K⟧
cfg outputs


β
entry block
⟦L⟧
ℒ
β₁
. . . .
β_n
⟦L⟧
⟦K⟧
⟦K⟧
⟦K⟧
⟦K⟧
⟦K⟧







CFG Semantics: Take II

$$[\![L \vdash \mathcal{L} \rhd K]\!]$$

$$: C_0([\![L]\!], [\![K]\!] + [\![L]\!])$$

CFG semantics : Take II

$$[\![L \vdash \mathcal{L} \rhd K]\!]$$

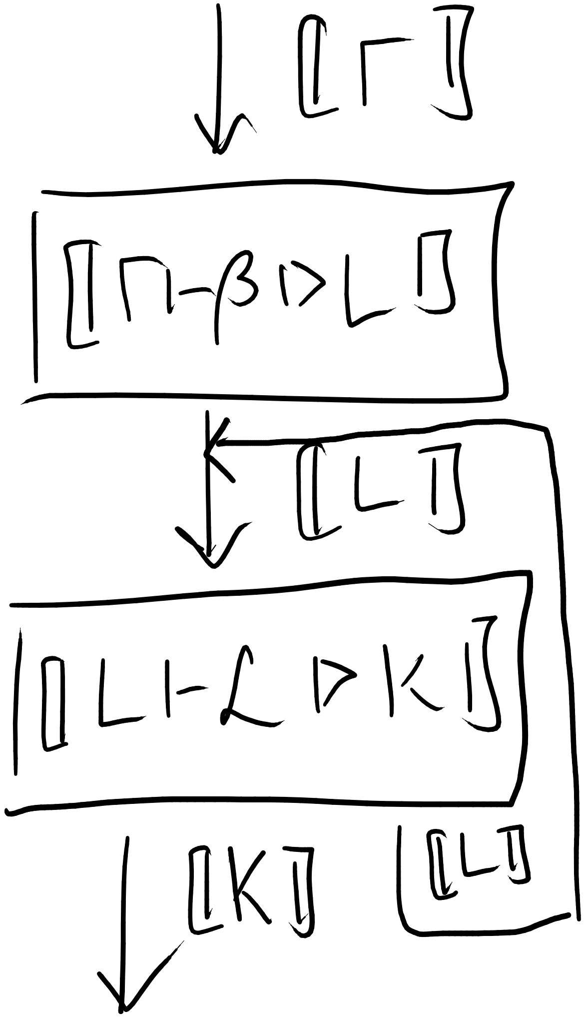
$$: C_0\left([\![L]\!],\ \underset{\substack{\uparrow \\ \text{outputs}}}{[\![K]\!]} + [\![L]\!]\right)$$

CFG Semantics : Take II

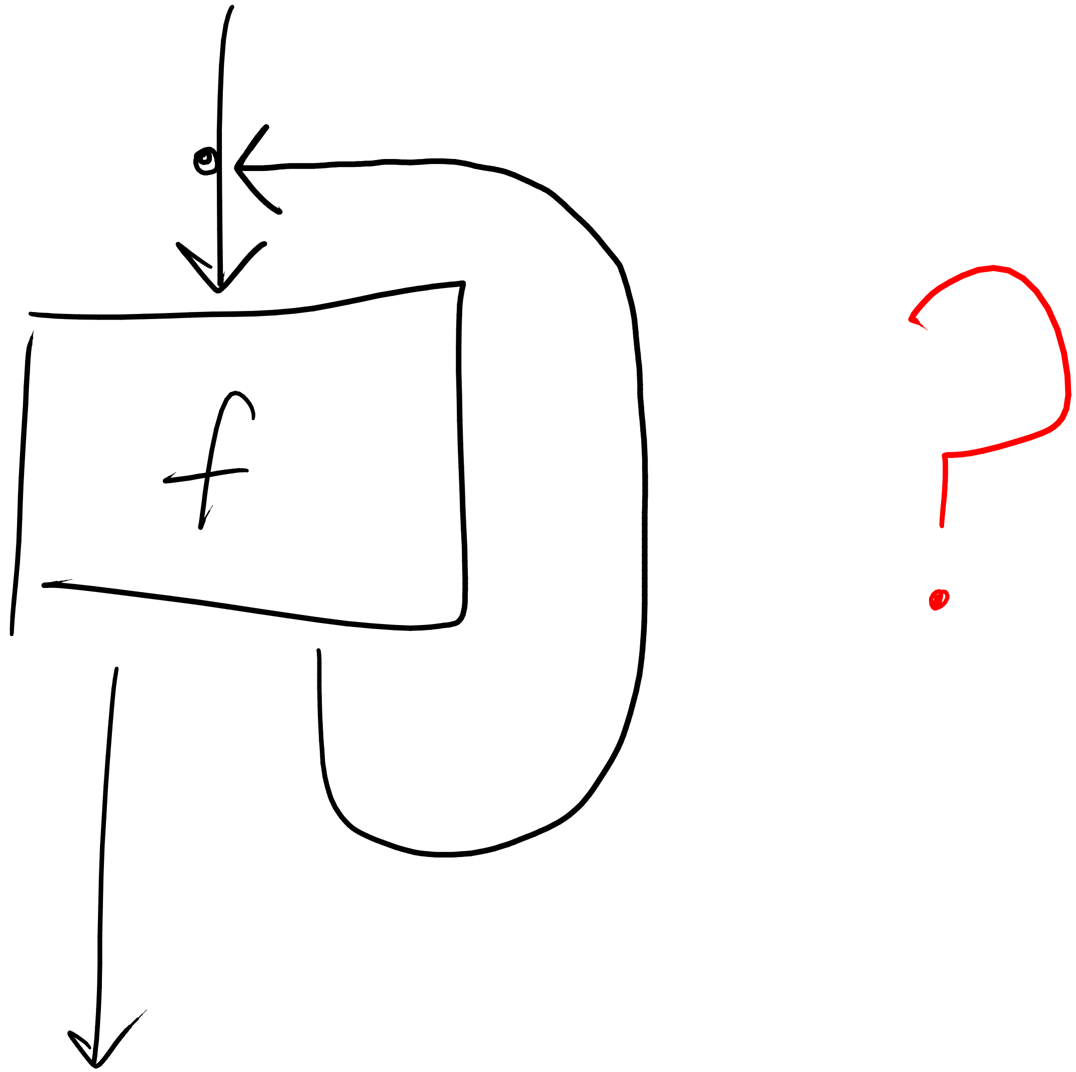
$$[\![L \vdash \mathcal{L} \rhd K]\!]$$

$$: C_0\left([\![L]\!],\ \underset{\substack{\uparrow \\ \text{outputs}}}{[\![K]\!]} + \underset{\substack{\uparrow \\ \text{jump to another} \\ \text{block inside CFG}}}{[\![L]\!]}\right)$$

$$\left\llbracket \frac{\Gamma \vdash \beta \rhd L \qquad L \vdash \mathcal{L} \rhd K}{\Gamma \vdash \beta \text{ where } \mathcal{L} \rhd K} \right\rrbracket =$$



Drawing CFGs

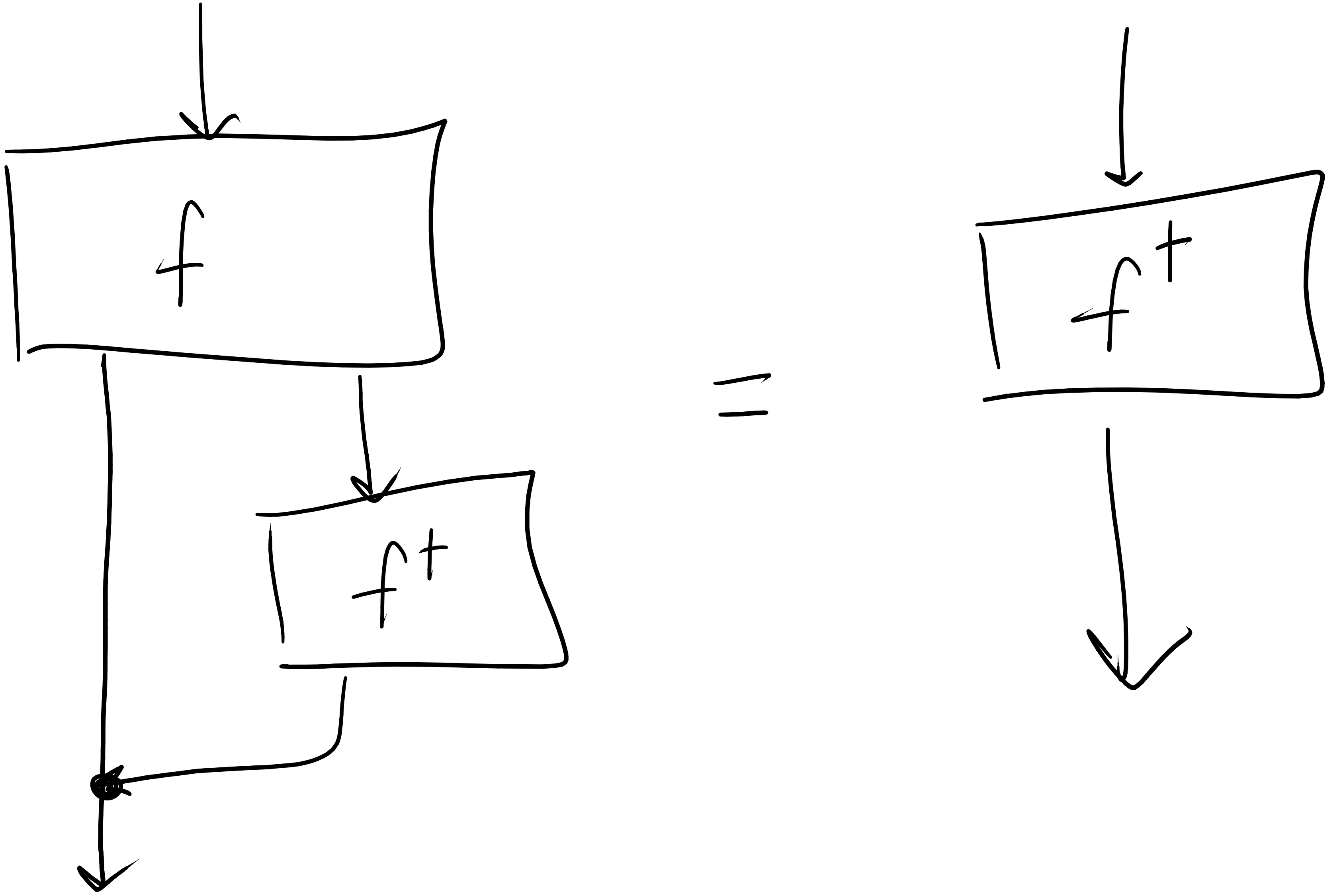


Elgot Structure

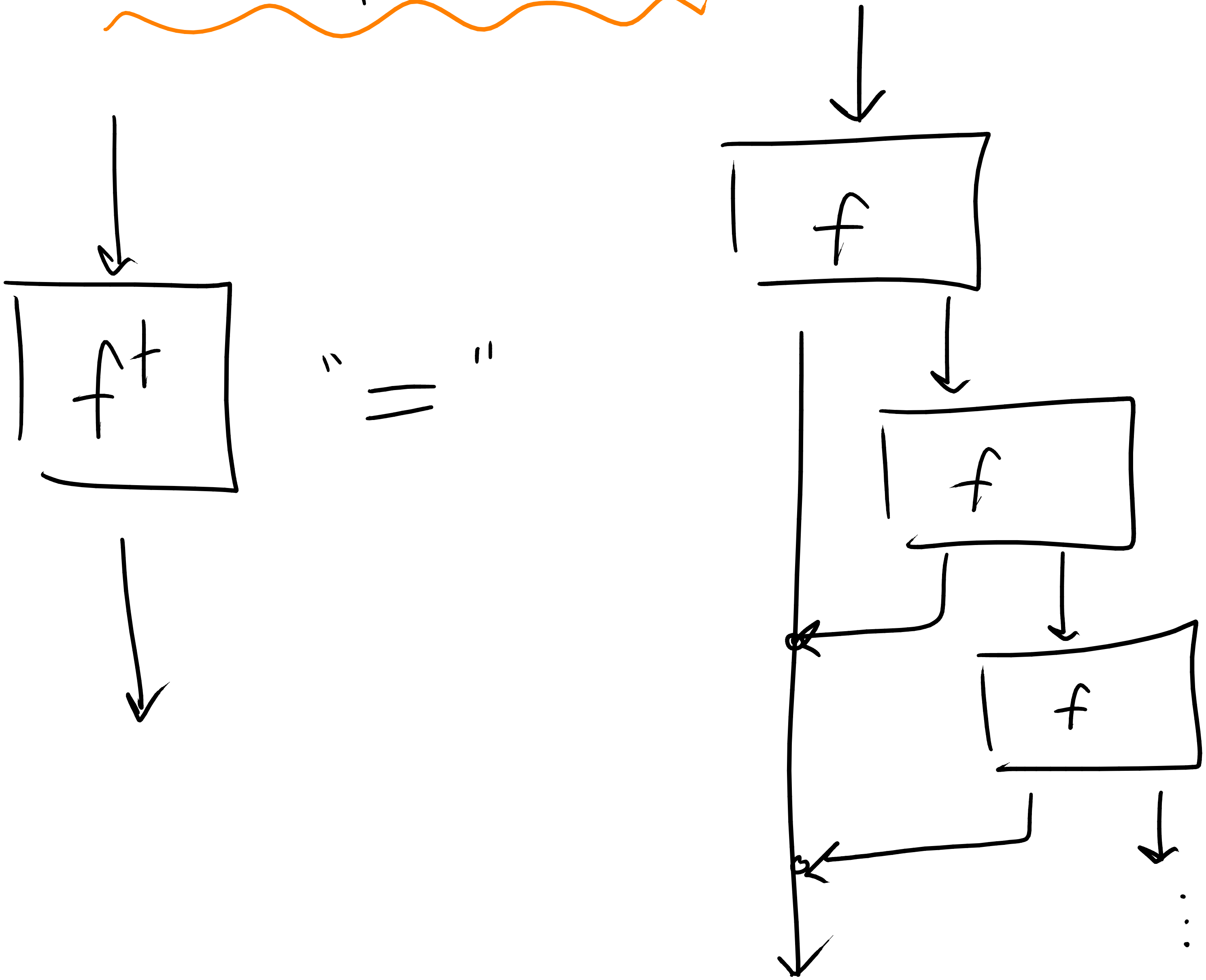
Given $f : A \to B + A$

Have $f^{+} : A \to B$

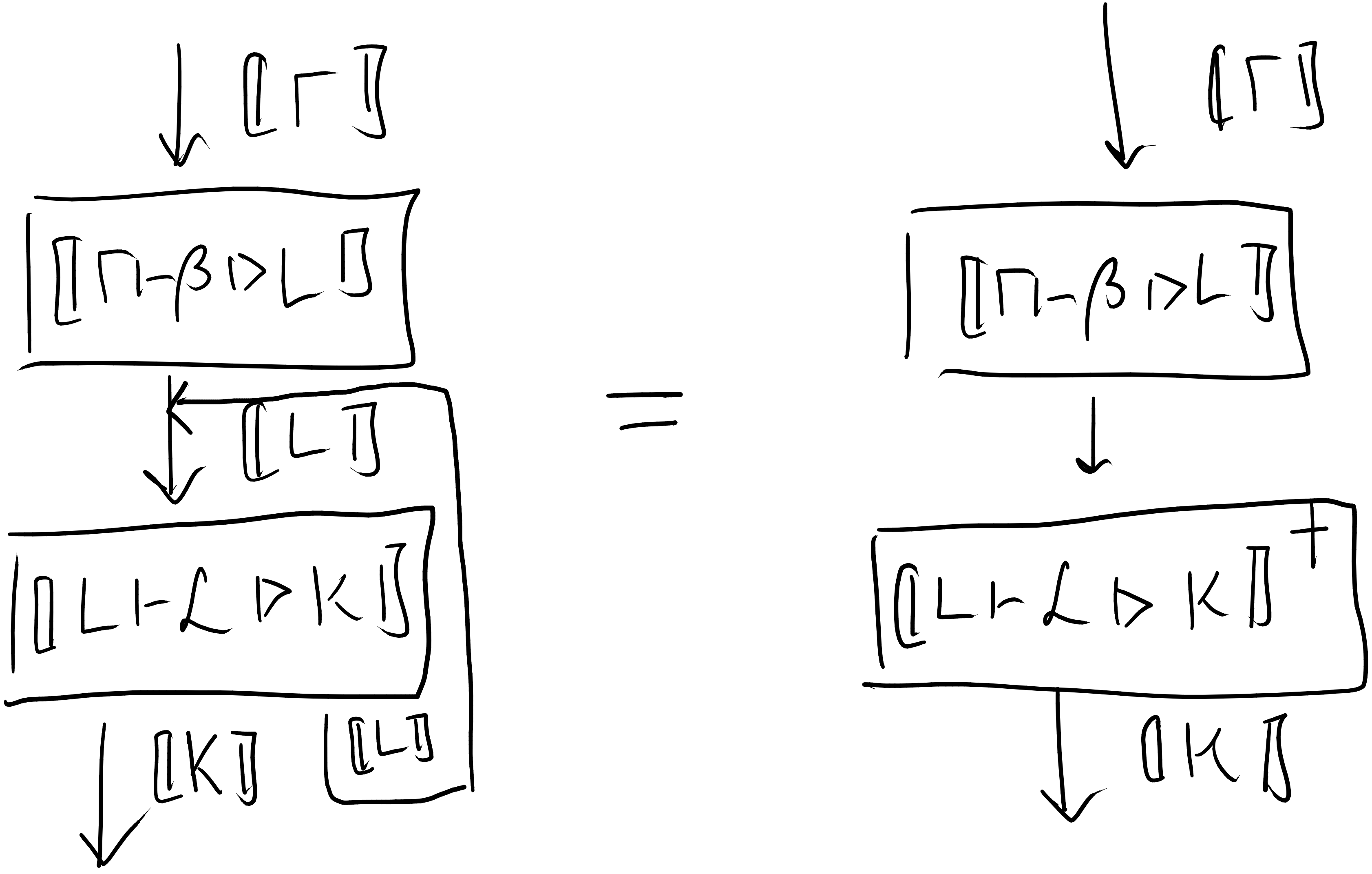
Fixpoints



Fixpoints



$$\left[\!\left[\frac{\Gamma \vdash \beta \triangleright L \qquad L \vdash \mathcal{L} \triangleright K}{\Gamma \vdash \beta \text{ where } \mathcal{L} \triangleright K} \right]\!\right] =$$

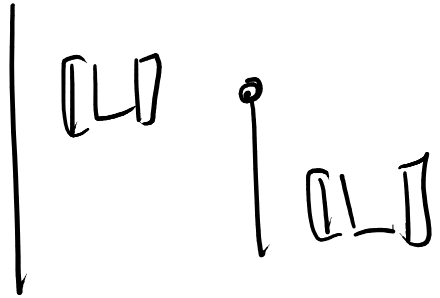


CFG semantics

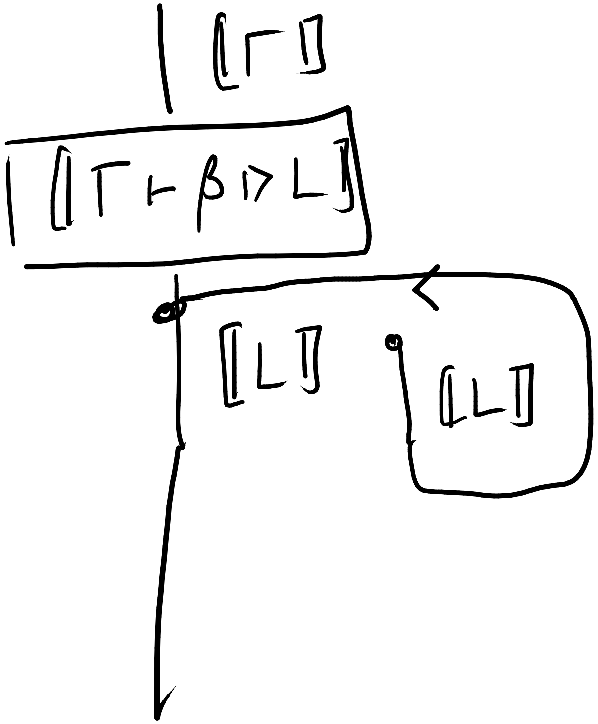
$$[\![\, \overline{L \vdash \cdot \triangleright L} \,]\!] = \mathrm{inl}_{[\![L]\!]} = \Big|^{[\![L]\!]} \quad i_{[\![L]\!]}$$

CFG semantics

$$[\![\frac{}{L \vdash \cdot \triangleright L}]\!] = \mathrm{id}_{[\![L]\!]} =$$

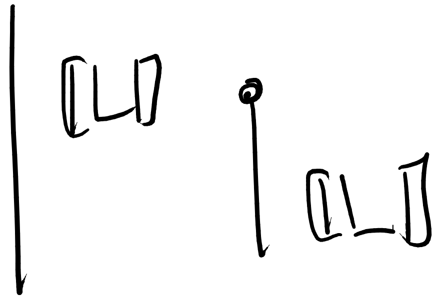


$$[\![\Gamma \vdash \beta \text{ where } \cdot \triangleright L]\!] =$$

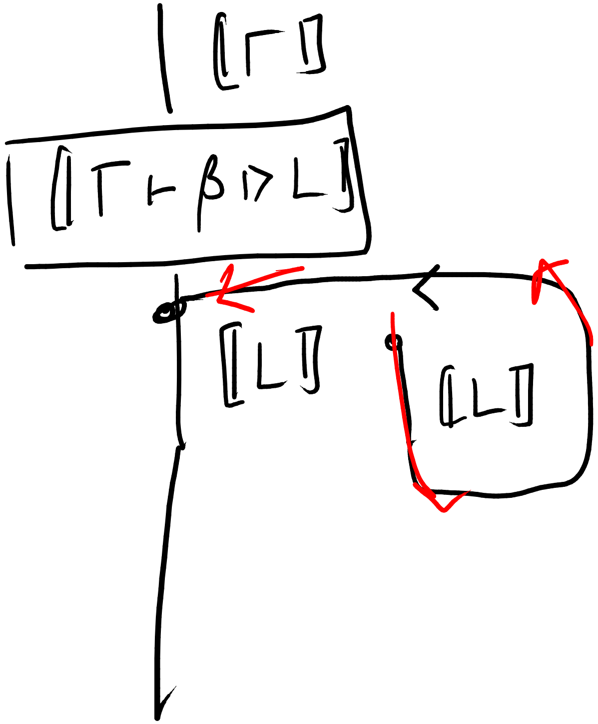


CFG semantics

$$[\![\frac{}{L \vdash \cdot \rhd L}]\!] = \mathrm{in}\ell_{[\![L]\!]} =$$



$$[\![\Gamma \vdash \beta \text{ where } \cdot \rhd L]\!] =$$



$$= [\![\Gamma \vdash \beta \rhd L]\!]$$

CFG semantics

$$\frac{L \vdash \mathcal{L} \triangleright K, {}^\wedge\ell[\Gamma](A) \qquad \Gamma, x:A \vdash \beta \triangleright L}{L \vdash \mathcal{L}, {}^\wedge\ell(x:A):\beta \triangleright K}$$

CFG semantics

Inputs

$$\frac{L \vdash \mathcal{L} \rhd K, {}^{\wedge}\ell[\Gamma](A) \qquad \Gamma, x : A \vdash \beta \rhd L}{L \vdash \mathcal{L}, {}^{\wedge}\ell(x : A) : \beta \rhd K}$$

CFGs semantics

Inputs outputs

$$\dfrac{L \vdash \mathcal{L} \triangleright K, \quad {}^\wedge\ell[\Gamma](A) \qquad \Gamma, x : A \vdash \beta \triangleright L}{L \vdash \mathcal{L}, {}^\wedge\ell(x : A) : \beta \triangleright K}$$

CFG semantics

Inputs

outputs

OR call ${}^\wedge\ell$ w/ Γ, A

$$\frac{L \vdash \mathcal{L} \triangleright K,\ {}^\wedge\ell[\Gamma](A) \qquad \Gamma, x : A \vdash \beta \triangleright L}{L \vdash \mathcal{L},\ {}^\wedge\ell(x : A) : \beta \triangleright K}$$

CFG semantics

Inputs ↓ outputs ↑ OR call $\hat{}\ell$ w/ Γ, A ↑ inputs to $\hat{}\ell$ ↓

$$\frac{L \vdash \mathcal{L} \triangleright K, \hat{}\ell[\Gamma](A) \qquad \Gamma, x : A \vdash \beta \triangleright L}{L \vdash \mathcal{L}, \hat{}\ell(x : A) : \beta \triangleright K}$$

CFGs semantics

$$\frac{\overset{\text{Inputs}}{L} \vdash \mathcal{L} \triangleright \overset{\text{outputs}}{K}, \overset{\text{OR call } {}^\wedge\ell \text{ w/ } \Gamma,\ A}{{}^\wedge\ell[\Gamma](A)} \qquad \overset{\text{inputs to } {}^\wedge\ell}{\Gamma, x:A} \vdash \overset{\text{body of } {}^\wedge\ell}{\beta} \triangleright L}{L \vdash \mathcal{L}, {}^\wedge\ell(x:A):\beta \triangleright K}$$

CFG semantics

$$\frac{\overset{\text{Inputs}}{L} \vdash \mathcal{L} \rhd \overset{\text{outputs}}{K},\ \overset{\text{OR call } {}^\wedge\ell \text{ w/ } \Gamma,\ A}{{}^\wedge\ell[\Gamma](A)} \qquad \overset{\text{inputs to } {}^\wedge\ell}{\Gamma, x:A} \vdash \overset{\text{body of } {}^\wedge\ell}{\beta} \rhd \overset{\text{inputs}}{L}}{L \vdash \mathcal{L}, {}^\wedge\ell(x:A):\beta \rhd K}$$

CFG semantics

$$\frac{\overset{\text{Inputs}}{L} \vdash \mathcal{L} \rhd \overset{\text{outputs}}{K}, \overset{\text{OR call } \hat{}\ell \text{ w/ } \Gamma,\ A}{\hat{}\ell[\Gamma](A)} \qquad \overset{\text{inputs to } \hat{}\ell}{\Gamma, x : A} \vdash \overset{\text{body of } \hat{}\ell}{\beta} \rhd \overset{\text{inputs}}{L}}{\underset{\text{inpts}}{L} \vdash \mathcal{L}, \hat{}\ell(x : A) : \beta \rhd \underset{\text{outputs}}{K}}$$

CFG semantics

Inputs

outputs

OR call $\hat{}\ell$ w/ Γ, A

inputs to $\hat{}\ell$

body of $\hat{}\ell$

inputs

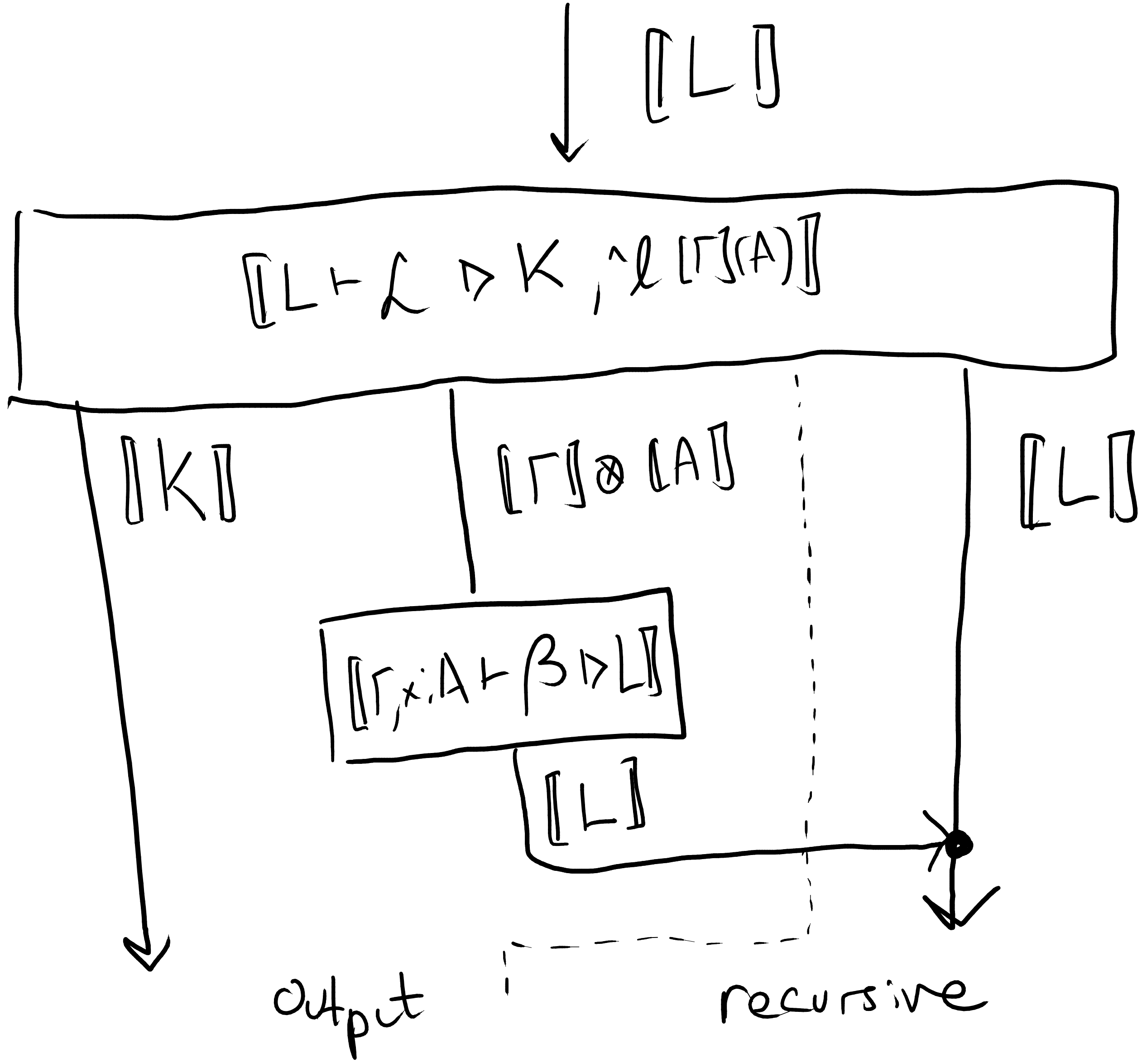
$$\frac{L \vdash \mathcal{L} \rhd K, \ \hat{}\ell[\Gamma](A) \qquad \Gamma, x : A \vdash \beta \rhd L}{L \vdash \mathcal{L}, \hat{}\ell(x : A) : \beta \rhd K}$$

outputs

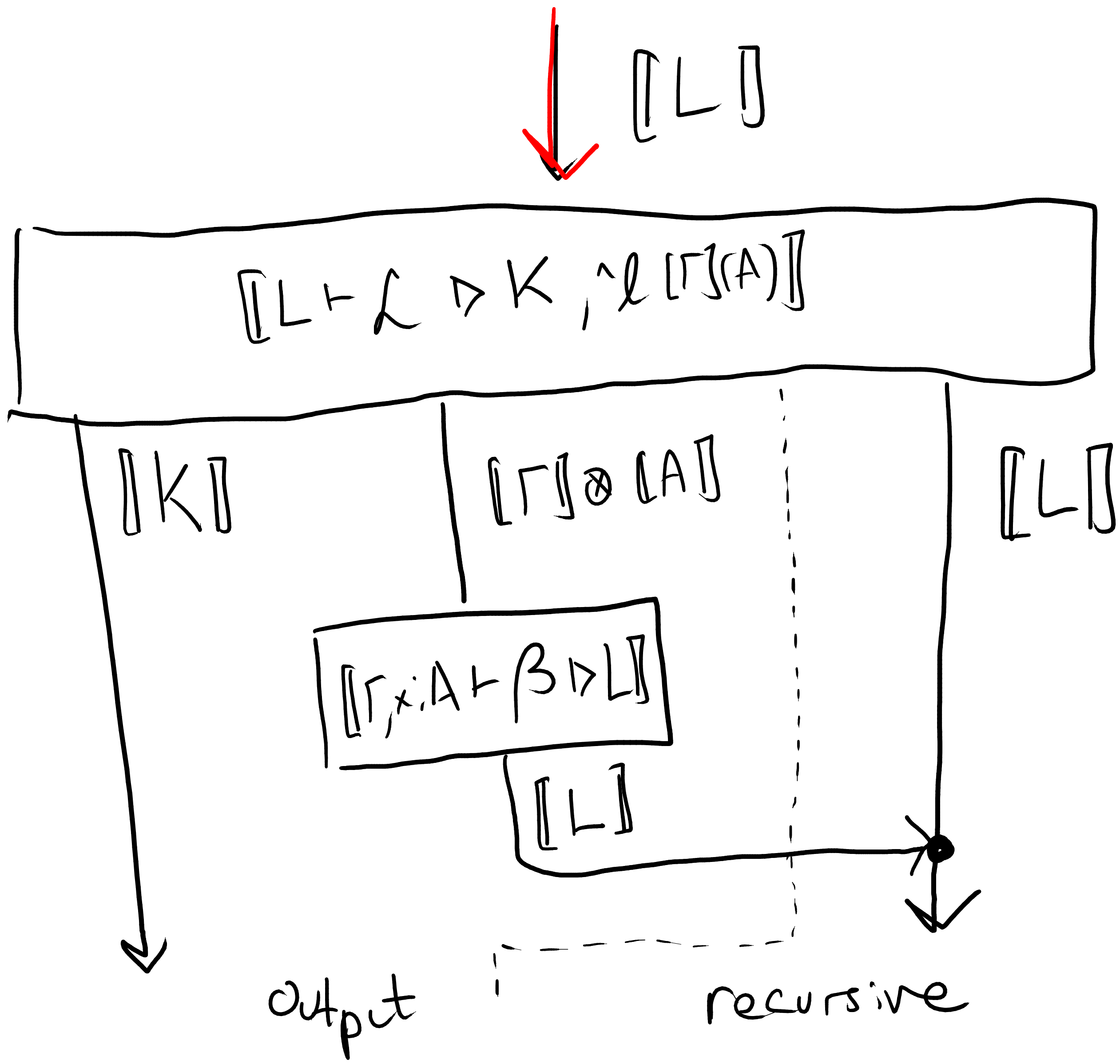
inpts

Since $\overline{L \vdash \cdot \rhd L}$, have "$L \subseteq K$"

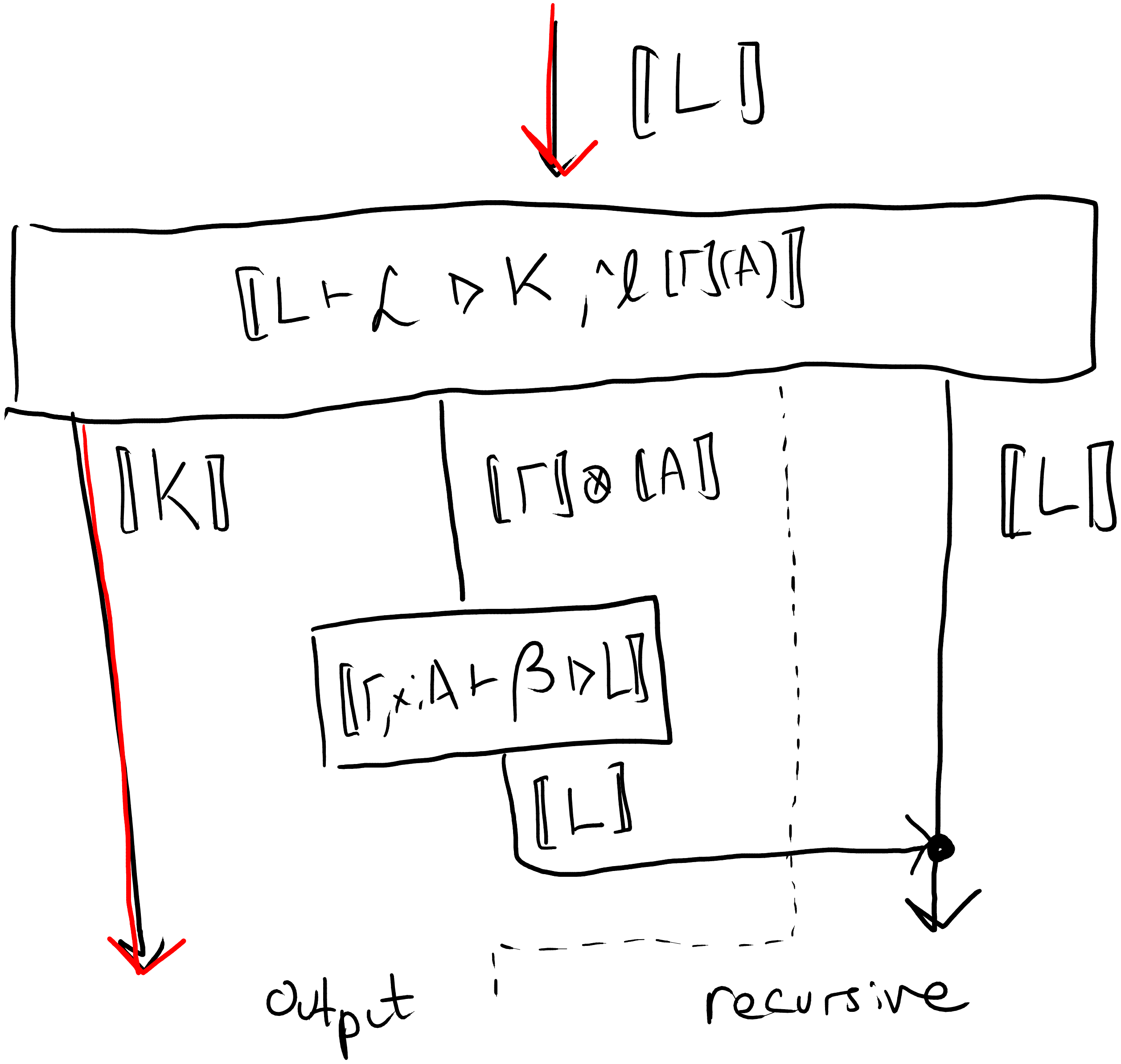
$$\left[\!\left[\frac{L \vdash \mathcal{L} \rhd K, {}^{\wedge}\ell[\Gamma](A) \qquad \Gamma, x{:}A \vdash \beta \rhd L}{L \vdash \mathcal{L}, {}^{\wedge}\ell(x{:}A){:}\beta \rhd K} \right]\!\right] =$$



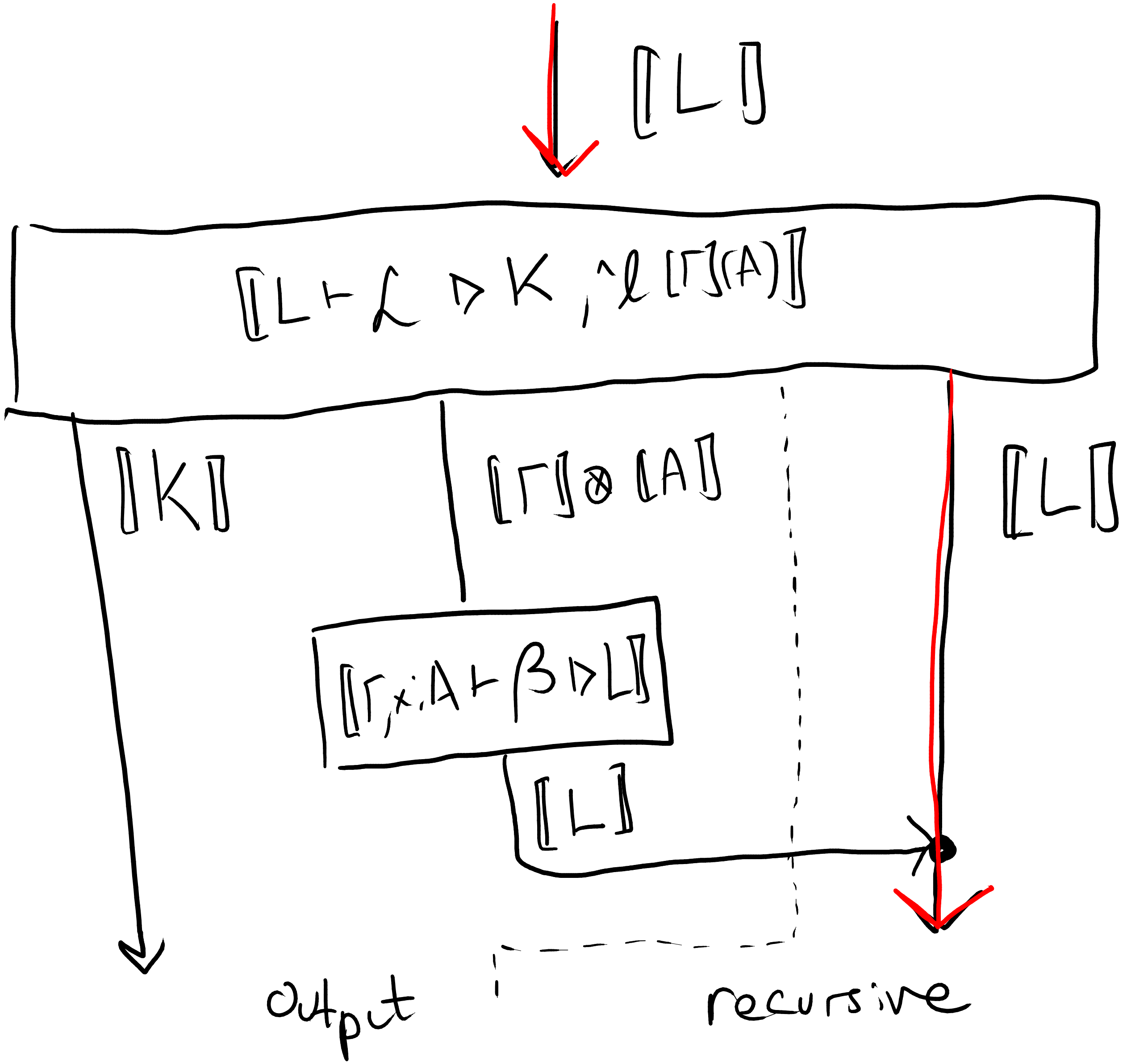
$$\left[\!\left[\frac{L \vdash \mathcal{L} \rhd K, {}^{\wedge}\ell[\Gamma](A) \qquad \Gamma, x{:}A \vdash \beta \rhd L}{L \vdash \mathcal{L}, {}^{\wedge}\ell(x{:}A){:}\beta \rhd K} \right]\!\right] =$$



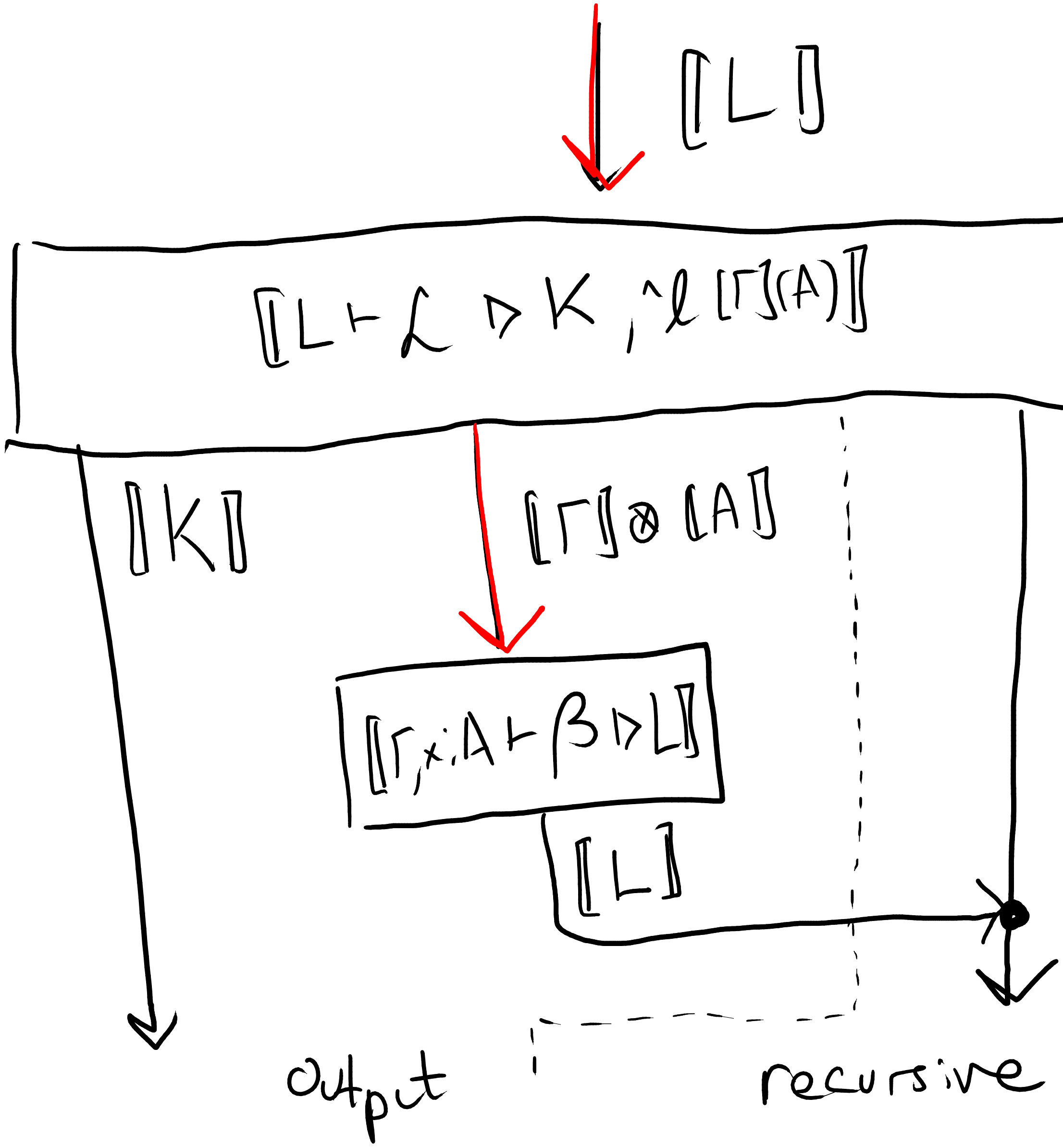
$$\left[\!\left[\frac{L \vdash \mathcal{L} \rhd K, {}^\wedge\ell[\Gamma](A) \qquad \Gamma, x{:}A \vdash \beta \rhd L}{L \vdash \mathcal{L}, {}^\wedge\ell(x{:}A){:}\beta \rhd K} \right]\!\right] =$$



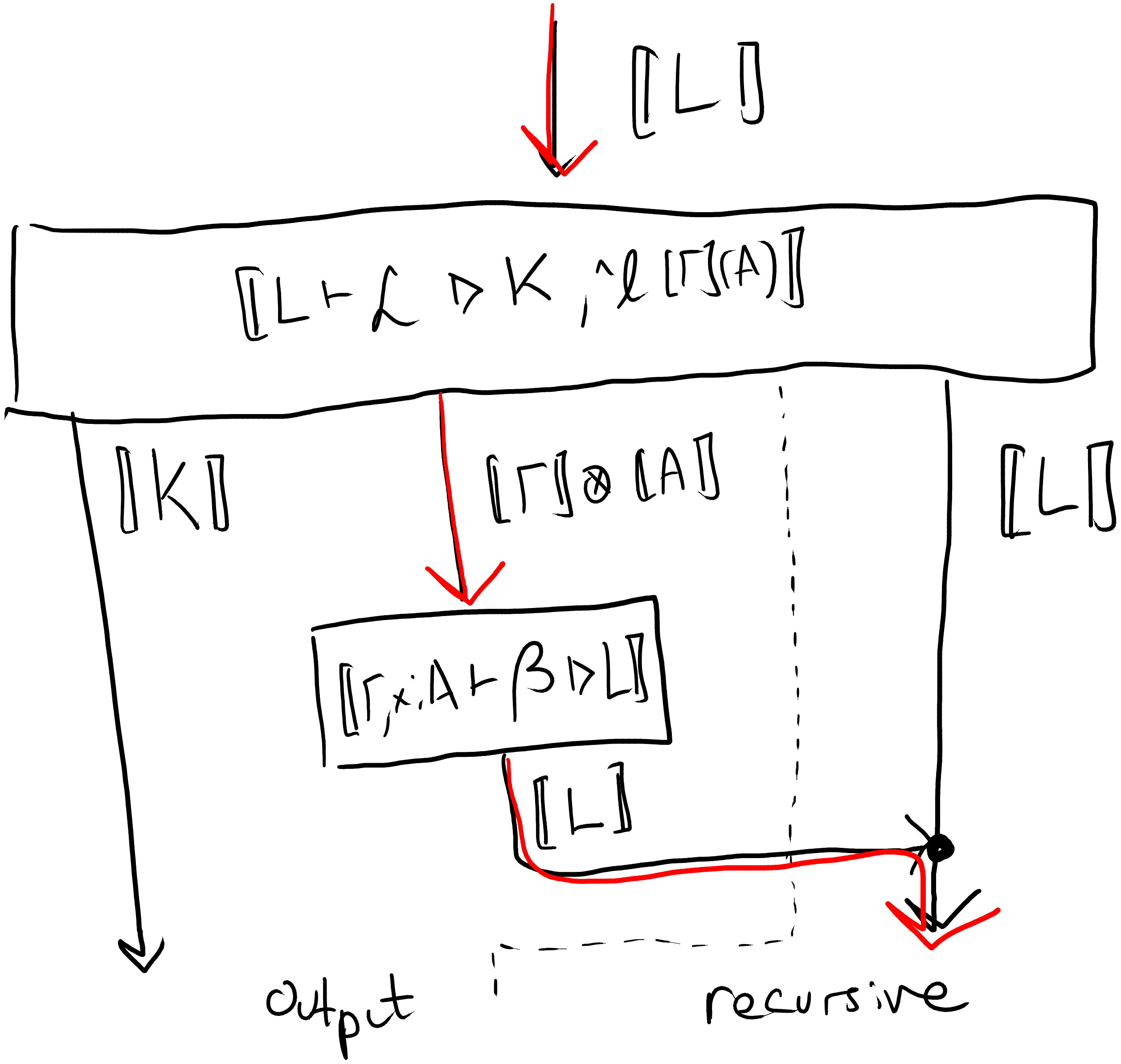
$$\left\llbracket \frac{L \vdash \mathcal{L} \rhd K, {}^{\wedge}\ell[\Gamma](A) \qquad \Gamma, x{:}A \vdash \beta \rhd L}{L \vdash \mathcal{L}, {}^{\wedge}\ell(x{:}A){:}\beta \rhd K} \right\rrbracket =$$



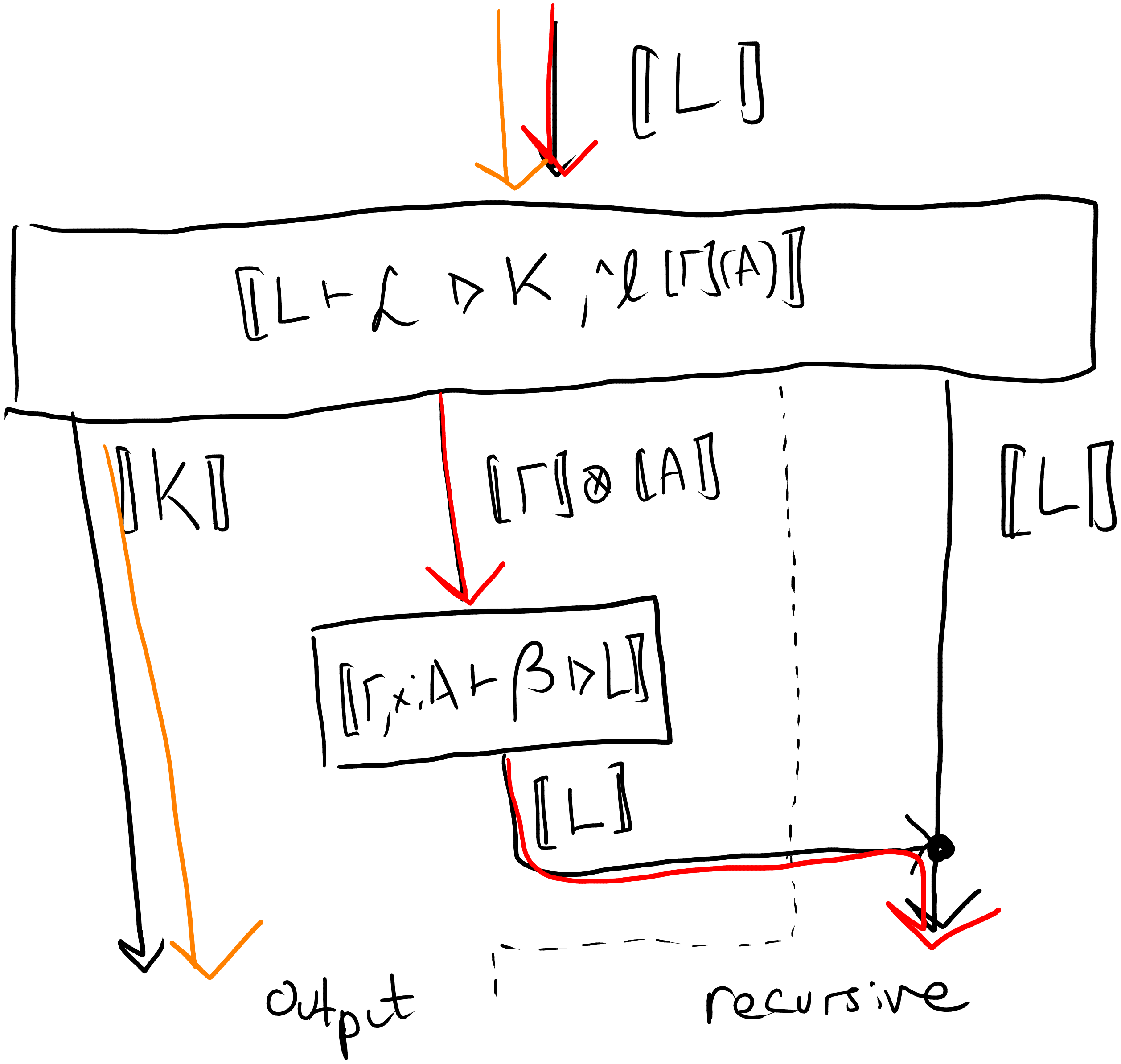
$$\left[\!\left[\frac{L \vdash \mathcal{L} \rhd K, {}^{\wedge}\ell[\Gamma](A) \qquad \Gamma, x{:}A \vdash \beta \rhd L}{L \vdash \mathcal{L}, {}^{\wedge}\ell(x{:}A) : \beta \rhd K} \right]\!\right] =$$



$$\left[\!\left[\frac{L \vdash \mathcal{L} \rhd K, {}^{\wedge}\ell[\Gamma](A) \qquad \Gamma, x{:}A \vdash \beta \rhd L}{L \vdash \mathcal{L}, {}^{\wedge}\ell(x{:}A){:}\beta \rhd K} \right]\!\right] =$$



$$\left[\!\left[\frac{L \vdash \mathcal{L} \rhd K, {}^\wedge\ell[\Gamma](A) \qquad \Gamma, x{:}A \vdash \beta \rhd L}{L \vdash \mathcal{L}, {}^\wedge\ell(x{:}A){:}\beta \rhd K} \right]\!\right] =$$



$$
{}^{\wedge}\ell_0[\Gamma_0](A_0),\ {}^{\wedge}\ell_1[\Gamma_1](A_1),\ {}^{\wedge}\ell_2[\Gamma_2](A_2) \vdash
$$

$$
{}^{\wedge}\ell_1(x_1 : A_1) : \beta_1,\ {}^{\wedge}\ell_2(x_2 : A_2) : \beta_2 \rhd\ {}^{\wedge}\ell_0[\Gamma_0](A_0)
$$

$${}^{\wedge}\ell_0[\Gamma_0](A_0),\ {}^{\wedge}\ell_1[\Gamma_1](A_1),\ {}^{\wedge}\ell_2[\Gamma_2](A_2) \vdash$$

$${}^{\wedge}\ell_1(x_1 : A_1) : \beta_1,\ {}^{\wedge}\ell_2(x_2 : A_2) : \beta_2 \rhd\ {}^{\wedge}\ell_0[\Gamma_0](A_0)$$

Basic blocks in CFG

callable by β_1, β_2 and entry block

$$
{}^\wedge\ell_0[\Gamma_0](A_0),\ {}^\wedge\ell_1[\Gamma_1](A_1),\ {}^\wedge\ell_2[\Gamma_2](A_2) \vdash
$$

$$
{}^\wedge\ell_1(x_1 : A_1) : \beta_1,\ {}^\wedge\ell_2(x_2 : A_2) : \beta_2 \rhd {}^\wedge\ell_0[\Gamma_0](A_0)
$$

Basic blocks in CFG

callable by β_1, β_2 and entry block

$$ {}^\wedge\ell_0[\Gamma_0](A_0),\ {}^\wedge\ell_1[\Gamma_1](A_1),\ {}^\wedge\ell_2[\Gamma_2](A_2) \vdash $$

$$ {}^\wedge\ell_1(x_1 : A_1) : \beta_1,\ {}^\wedge\ell_2(x_2 : A_2) : \beta_2 \rhd {}^\wedge\ell_0[\Gamma_0](A_0) $$

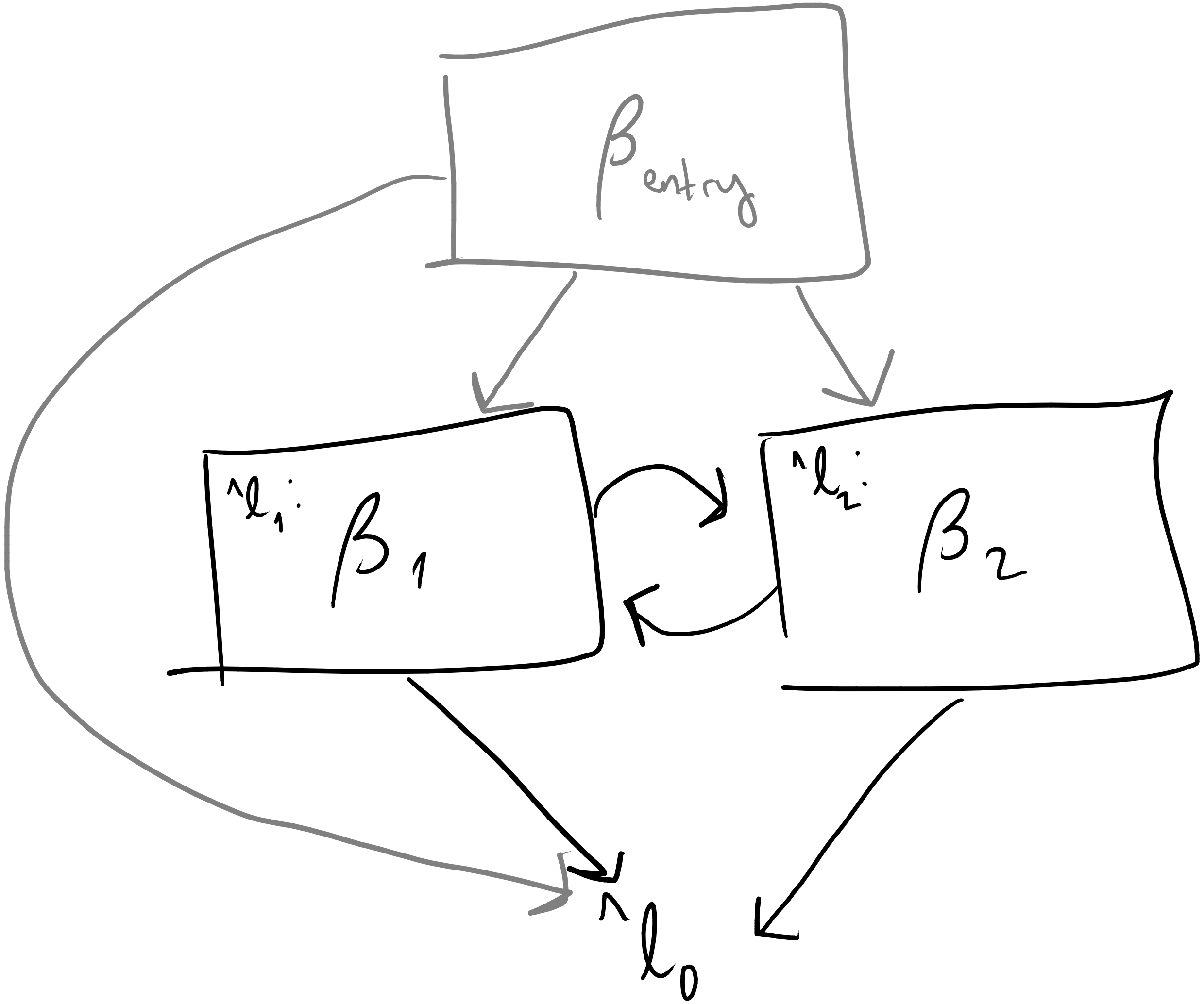
Basic blocks in CFG

No ${}^\wedge\ell_0$

exits of CFG (hence 1)

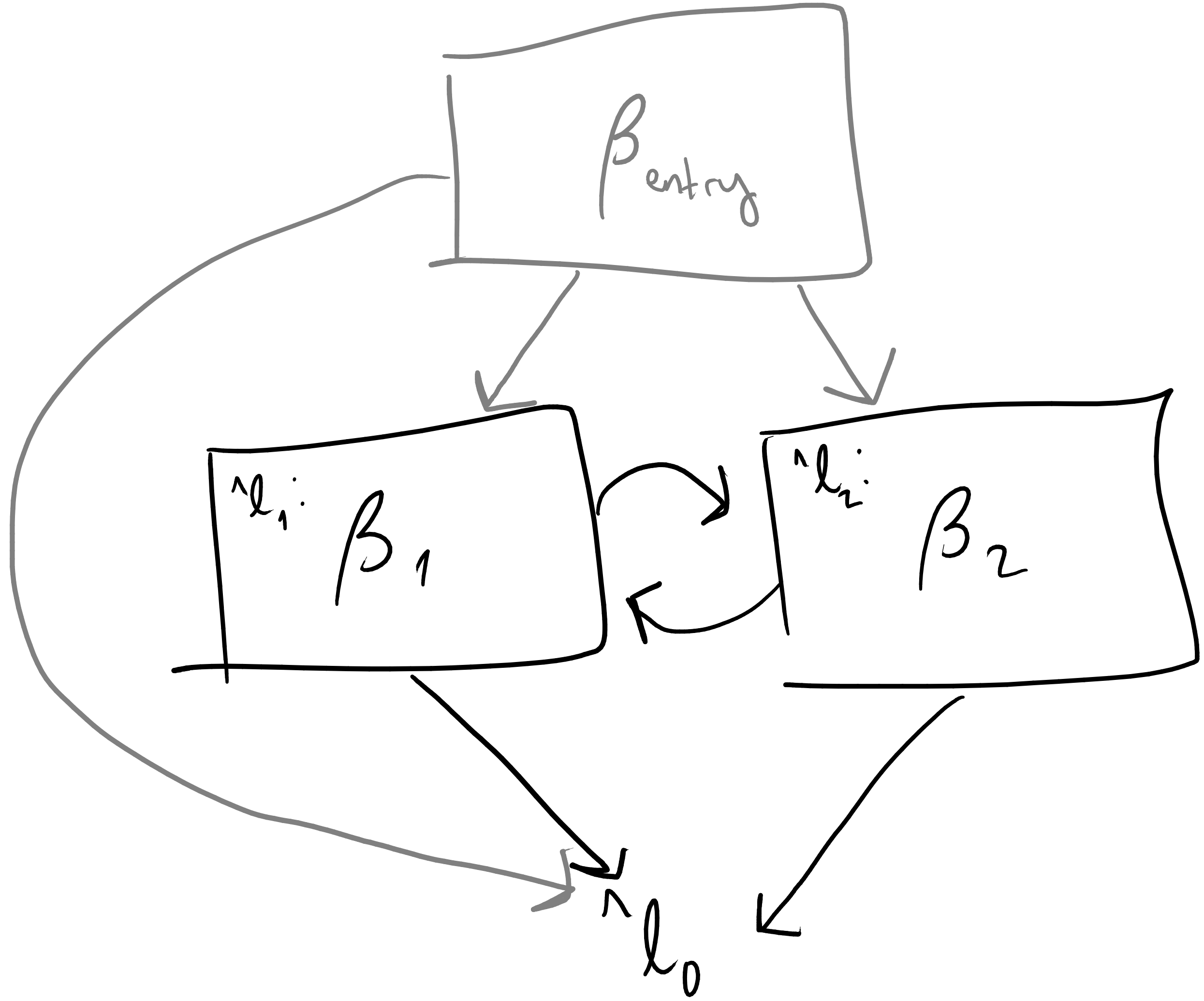
$${}^{\wedge}\ell_0[\Gamma_0](A_0),\ {}^{\wedge}\ell_1[\Gamma_1](A_1),\ {}^{\wedge}\ell_2[\Gamma_2](A_2) \vdash$$

$${}^{\wedge}\ell_1(x_1 : A_1) : \beta_1,\ {}^{\wedge}\ell_2(x_2 : A_2) : \beta_2 \rhd\ {}^{\wedge}\ell_0[\Gamma_0](A_0)$$



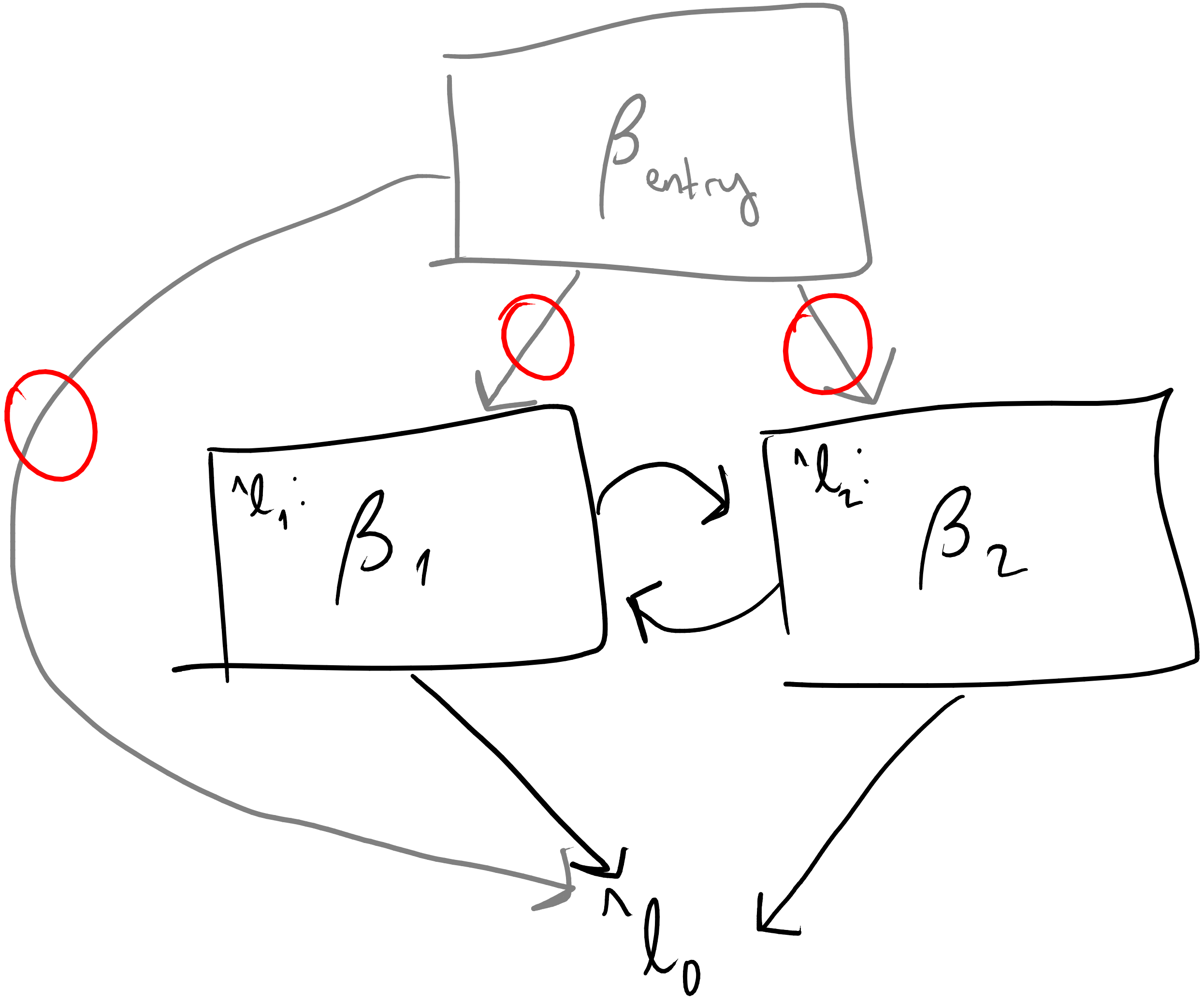
$$ {}^{\wedge}\ell_0[\Gamma_0](A_0),\ {}^{\wedge}\ell_1[\Gamma_1](A_1),\ {}^{\wedge}\ell_2[\Gamma_2](A_2) \vdash $$

$$ {}^{\wedge}\ell_1(x_1 : A_1) : \beta_1,\ {}^{\wedge}\ell_2(x_2 : A_2) : \beta_2 \triangleright {}^{\wedge}\ell_0[\Gamma_0](A_0) $$



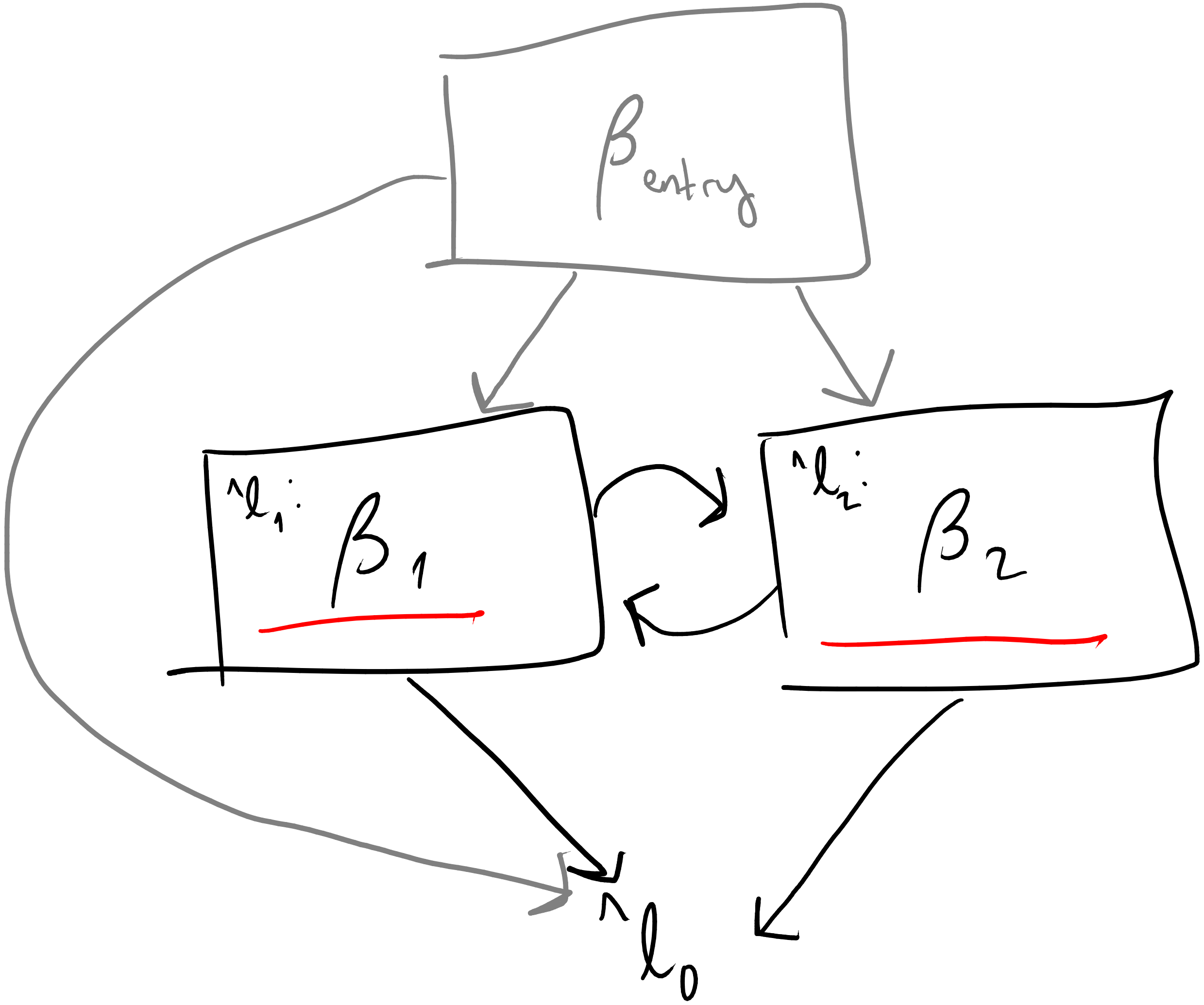
$$\underline{{}^\wedge\ell_0}[\Gamma_0](A_0),\ \underline{{}^\wedge\ell_1}[\Gamma_1](A_1),\ \underline{{}^\wedge\ell_2}[\Gamma_2](A_2) \vdash$$

$${}^\wedge\ell_1(x_1 : A_1) : \beta_1,\ {}^\wedge\ell_2(x_2 : A_2) : \beta_2 \rhd\ {}^\wedge\ell_0[\Gamma_0](A_0)$$



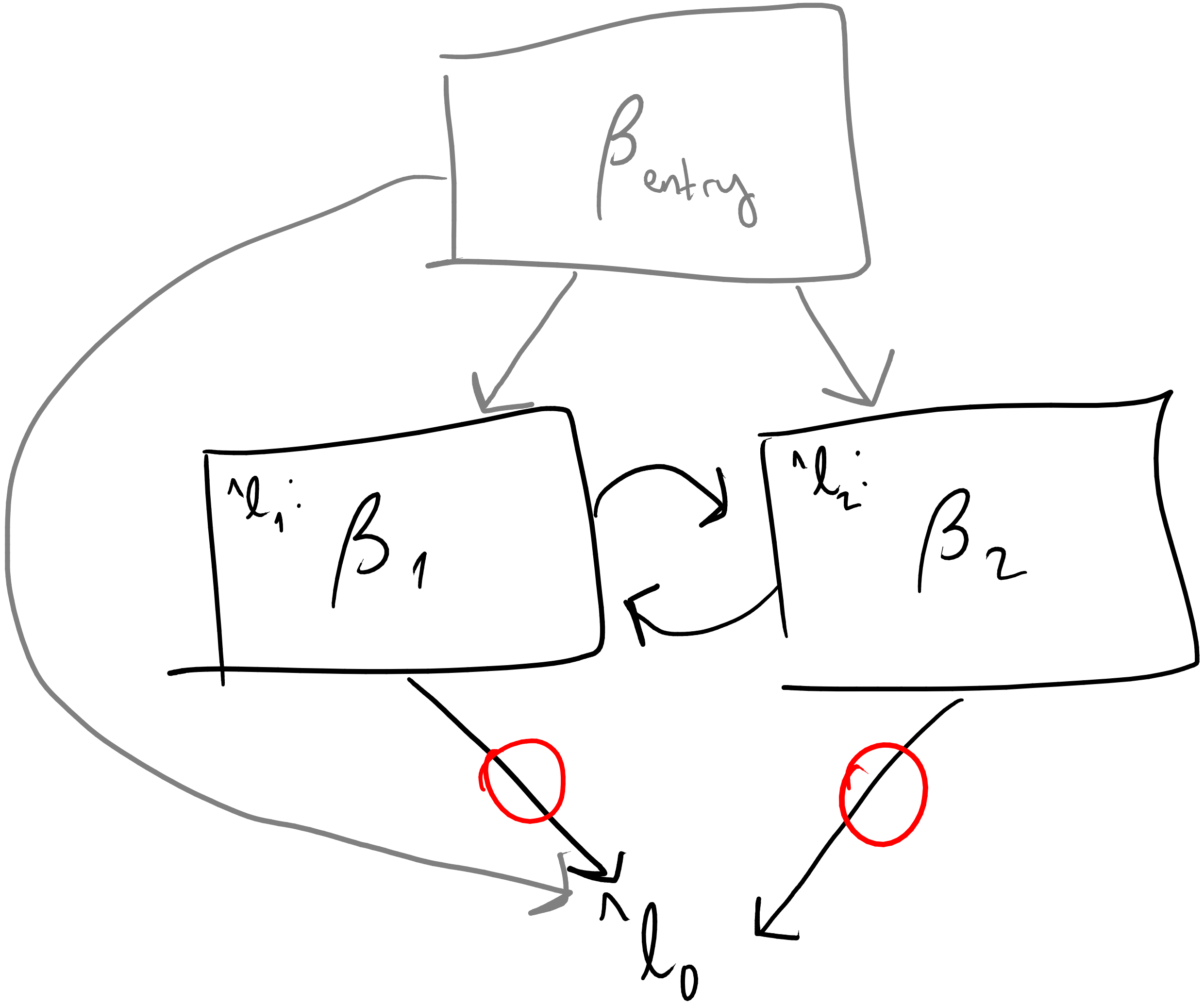
$$
{}^{\wedge}\ell_0[\Gamma_0](A_0),\ {}^{\wedge}\ell_1[\Gamma_1](A_1),\ {}^{\wedge}\ell_2[\Gamma_2](A_2) \vdash
$$

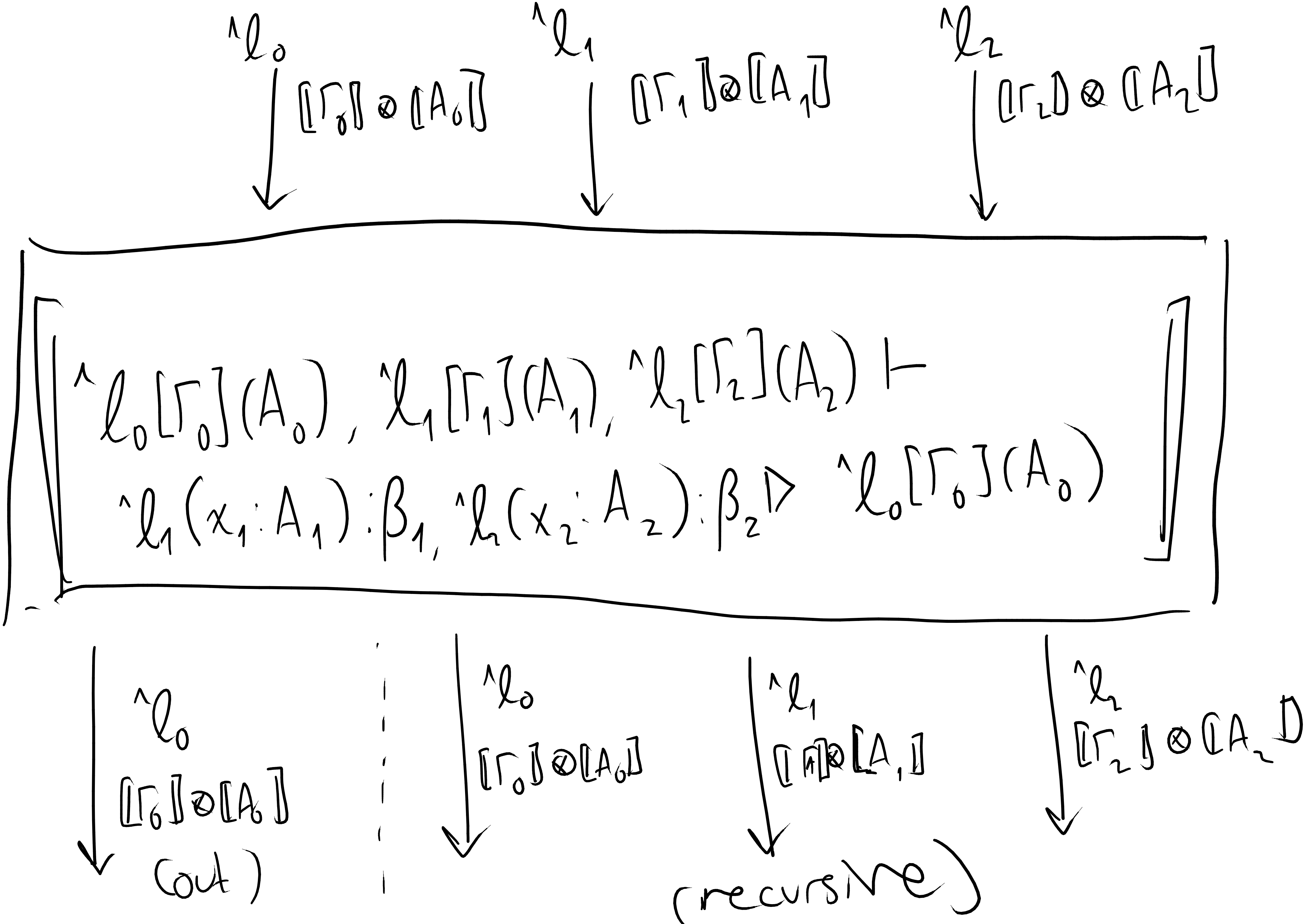
$$
\underline{{}^{\wedge}\ell_1(x_1 : A_1) : \beta_1},\ \underline{{}^{\wedge}\ell_2(x_2 : A_2) : \beta_2} \rhd {}^{\wedge}\ell_0[\Gamma_0](A_0)
$$



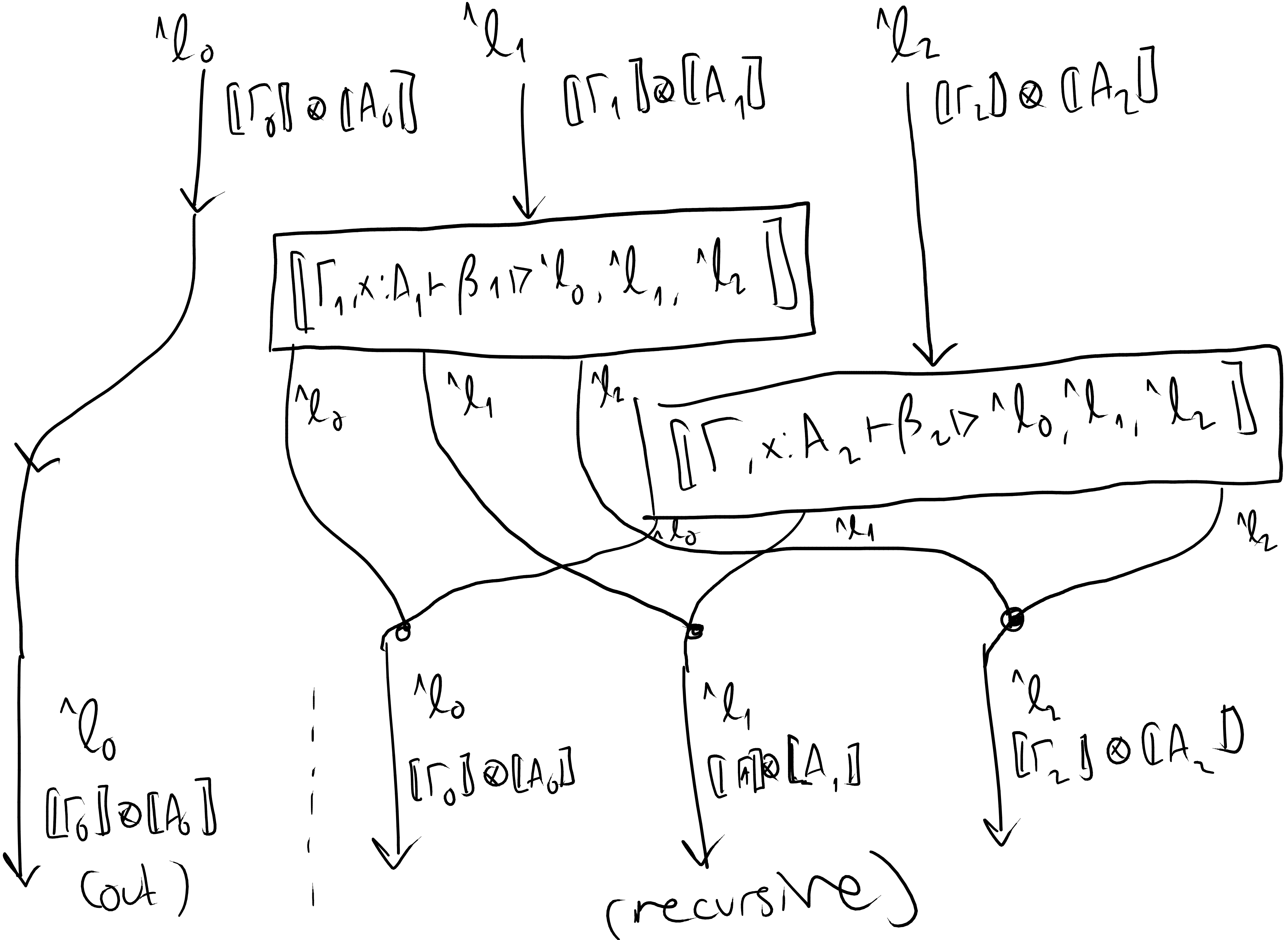
$$
{}^\wedge\ell_0[\Gamma_0](A_0),\ {}^\wedge\ell_1[\Gamma_1](A_1),\ {}^\wedge\ell_2[\Gamma_2](A_2) \vdash
$$

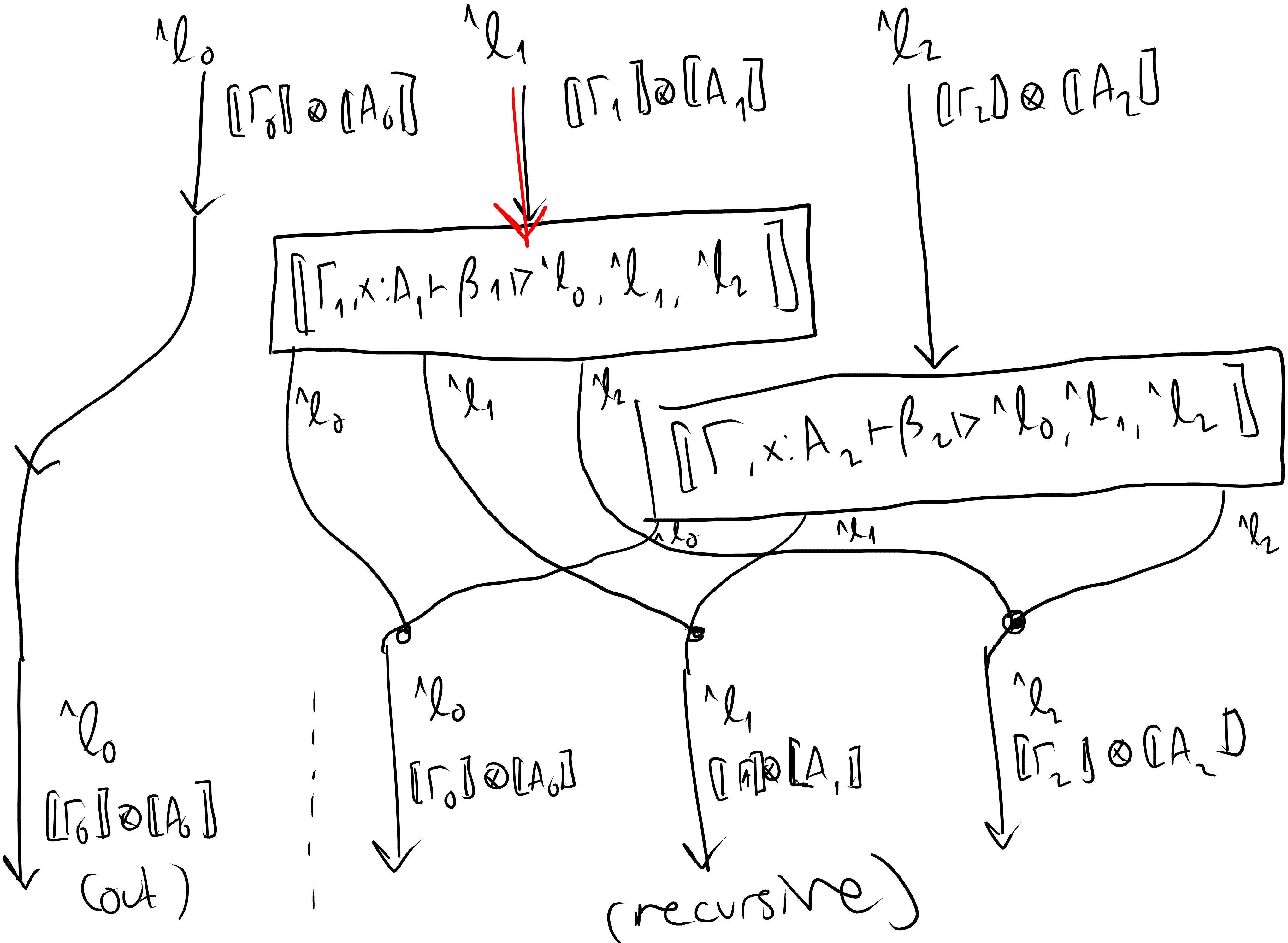
$$
{}^\wedge\ell_1(x_1 : A_1) : \beta_1,\ {}^\wedge\ell_2(x_2 : A_2) : \beta_2 \rhd \underline{{}^\wedge\ell_0}[\Gamma_0](A_0)
$$



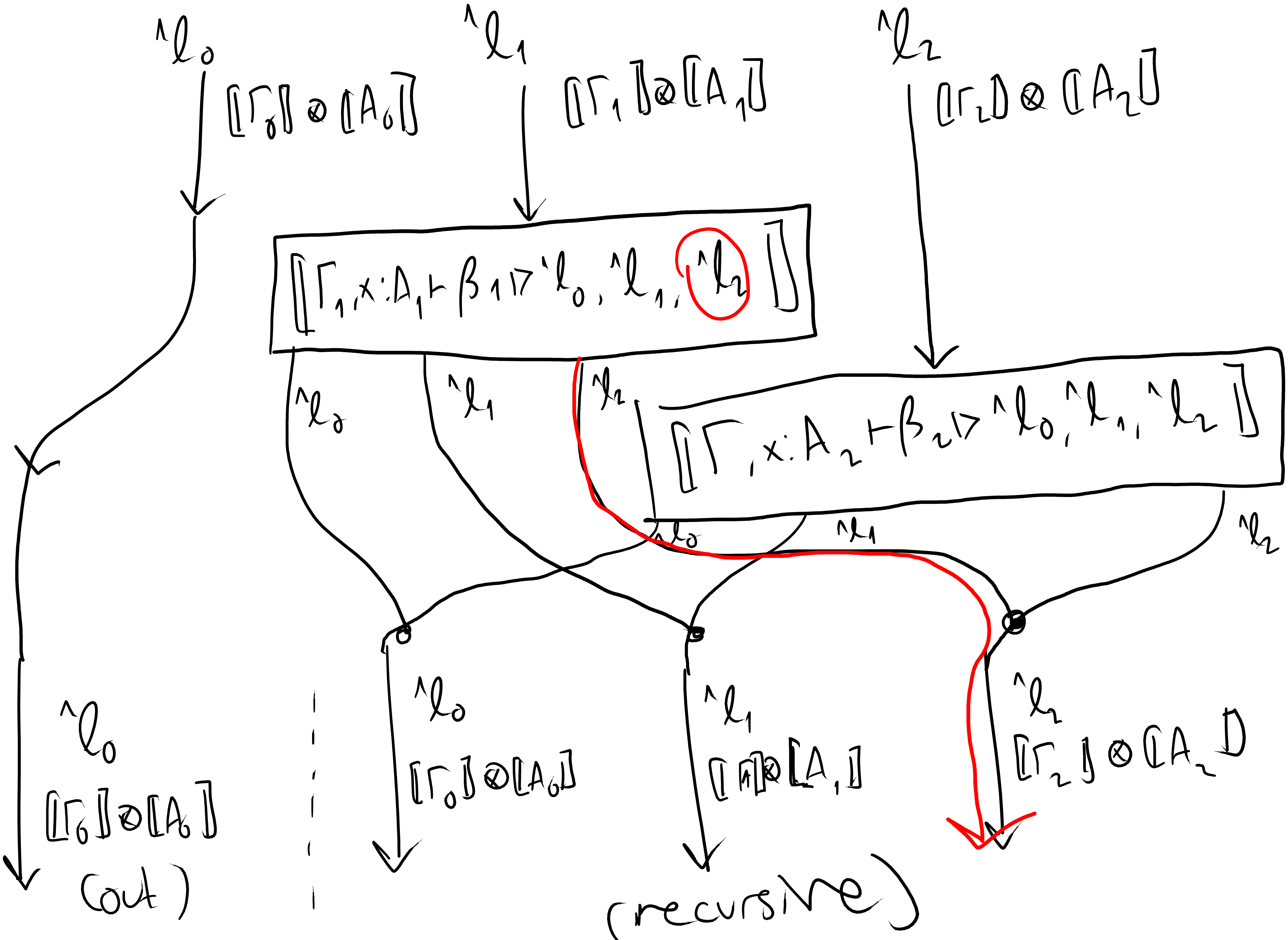


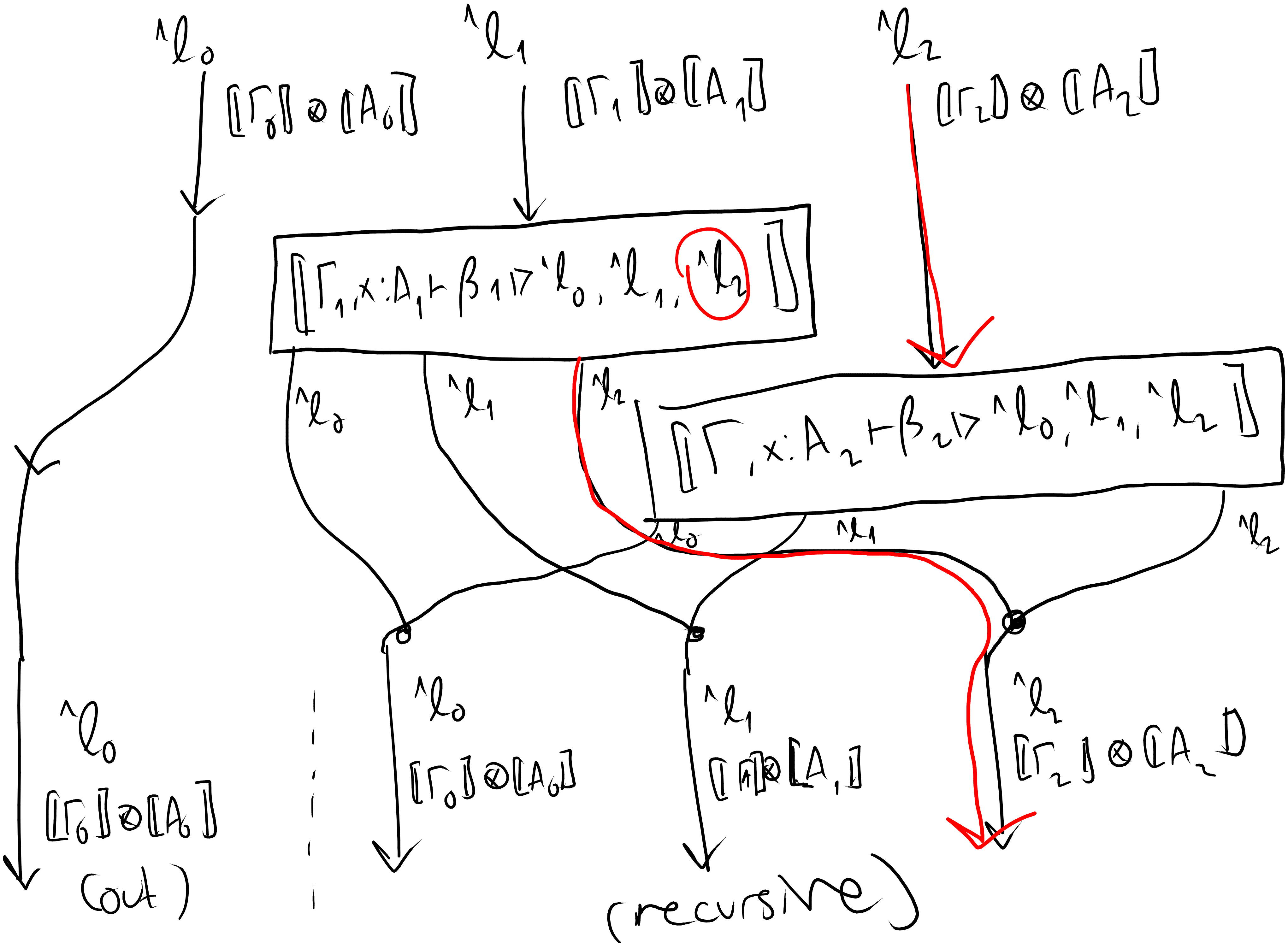
${}^{\wedge}\ell_0$
$\llbracket \Gamma_0 \rrbracket \otimes \llbracket A_0 \rrbracket$
${}^{\wedge}\ell_1$
$\llbracket \Gamma_1 \rrbracket \otimes \llbracket A_1 \rrbracket$
${}^{\wedge}\ell_2$
$\llbracket \Gamma_2 \rrbracket \otimes \llbracket A_2 \rrbracket$
$\llbracket {}^{\wedge}\ell_0[\Gamma_0](A_0), {}^{\wedge}\ell_1[\Gamma_1](A_1), {}^{\wedge}\ell_2[\Gamma_2](A_2) \vdash$
${}^{\wedge}\ell_1(x_1 : A_1) : \beta_1, {}^{\wedge}\ell_2(x_2 : A_2) : \beta_2 \triangleright {}^{\wedge}\ell_0[\Gamma_0](A_0) \rrbracket$
${}^{\wedge}\ell_0$
$\llbracket \Gamma_0 \rrbracket \otimes \llbracket A_0 \rrbracket$
(out)
${}^{\wedge}\ell_0$
$\llbracket \Gamma_0 \rrbracket \otimes \llbracket A_0 \rrbracket$
${}^{\wedge}\ell_1$
$\llbracket \Gamma_1 \rrbracket \otimes \llbracket A_1 \rrbracket$
${}^{\wedge}\ell_2$
$\llbracket \Gamma_2 \rrbracket \otimes \llbracket A_2 \rrbracket$
(recursive)

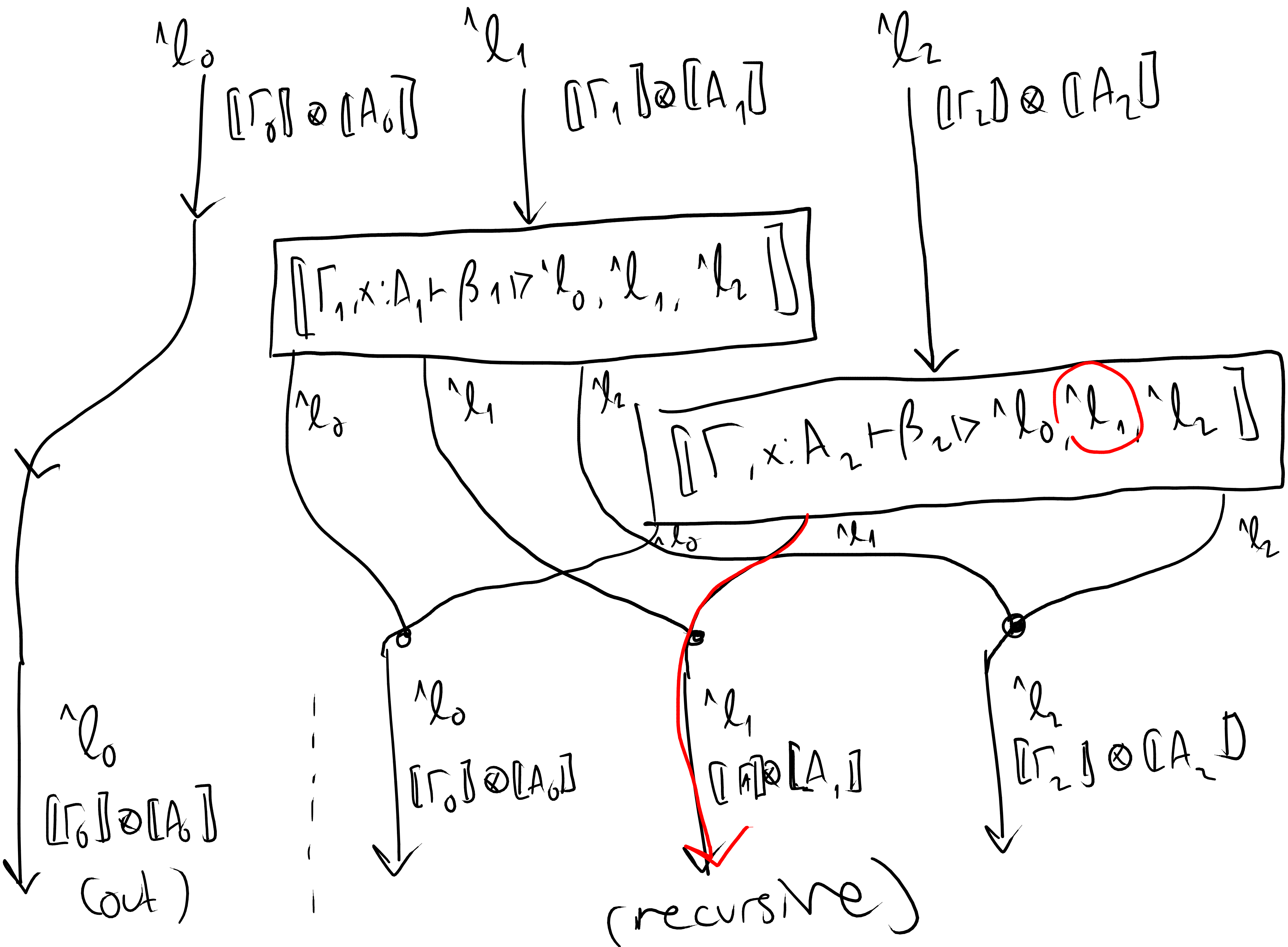

^l0
⟦Γ0⟧ ⊗ ⟦A0⟧
^l1
⟦Γ1⟧ ⊗ ⟦A1⟧
^l2
⟦Γ2⟧ ⊗ ⟦A2⟧
⟦Γ1, x:A1 ⊢ β1 ▷ ^l0, ^l1, ^l2⟧
^l0
^l1
^l2
⟦Γ, x:A2 ⊢ β2 ▷ ^l0, ^l1, ^l2⟧
^l0
^l1
^l2
^l0
⟦Γ0⟧ ⊗ ⟦A0⟧
(out)
^l0
⟦Γ0⟧ ⊗ ⟦A0⟧
^l1
⟦Γ1⟧ ⊗ ⟦A1⟧
^l2
⟦Γ2⟧ ⊗ ⟦A2⟧
(recursive)

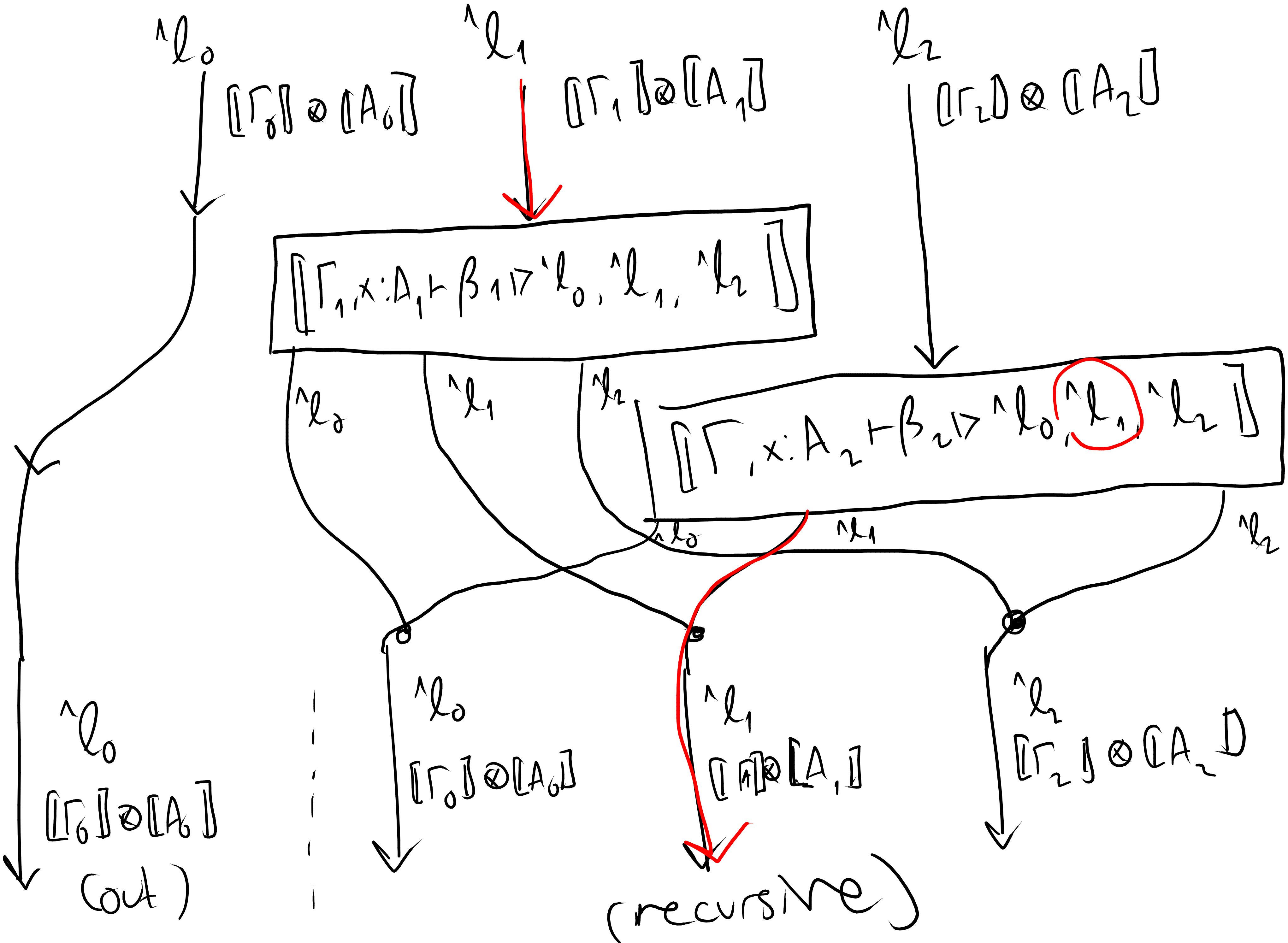


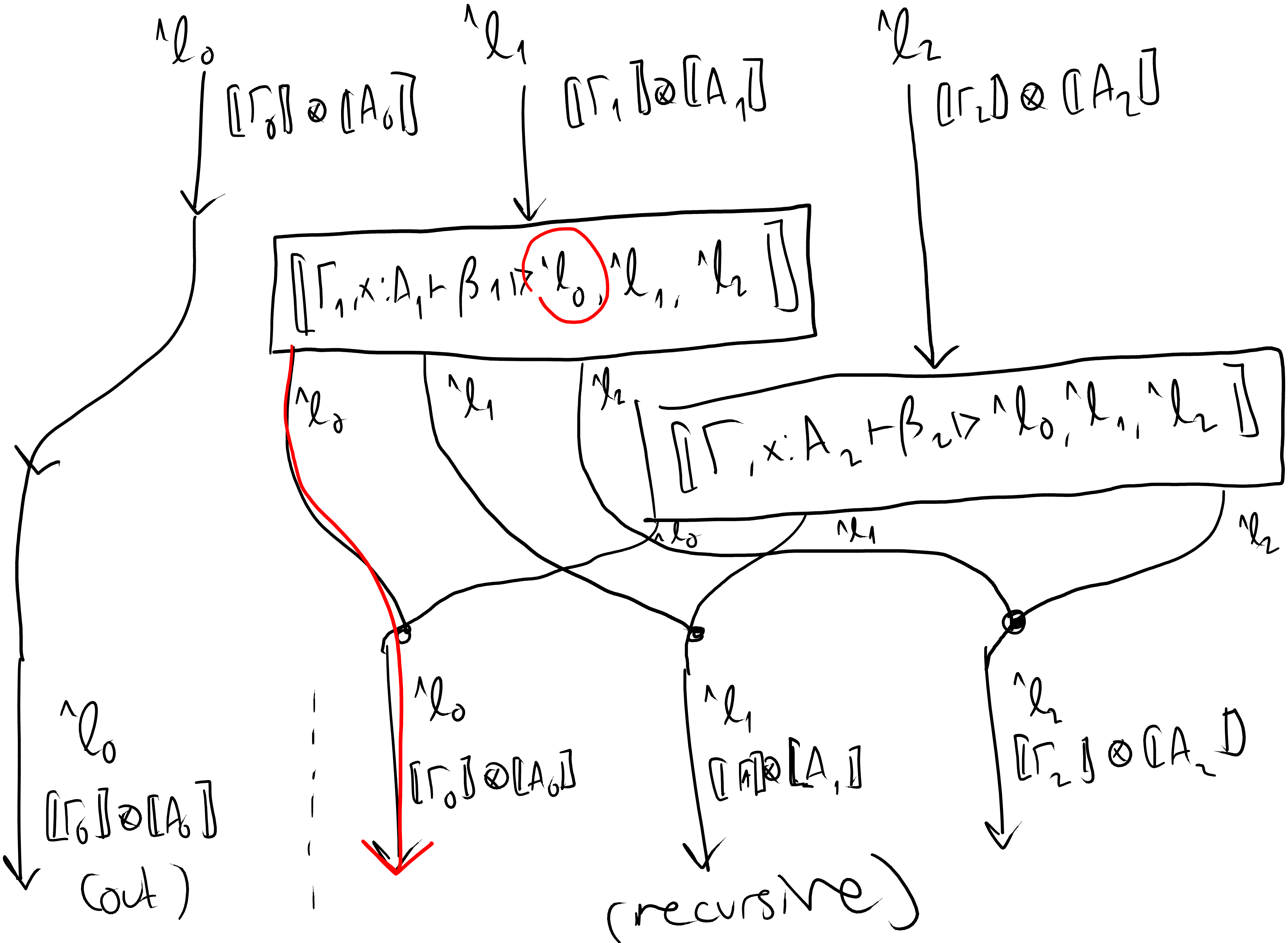
^l₀
^l₁
^l₂
[[Γ₀]] ⊗ [[A₀]]
[[Γ₁]] ⊗ [[A₁]]
[[Γ₂]] ⊗ [[A₂]]
[[Γ₁, x:A₁ ⊢ β₁ ▷ ^l₀, ^l₁, ^l₂]]
^l₀
^l₁
^l₂
[[Γ, x:A₂ ⊢ β₂ ▷ ^l₀, ^l₁, ^l₂]]
^l₀
^l₁
^l₂
^l₀
[[Γ₀]] ⊗ [[A₀]]
(out)
^l₀
[[Γ₀]] ⊗ [[A₀]]
^l₁
[[Γ₁]] ⊗ [[A₁]]
^l₂
[[Γ₂]] ⊗ [[A₂]]
(recursive)

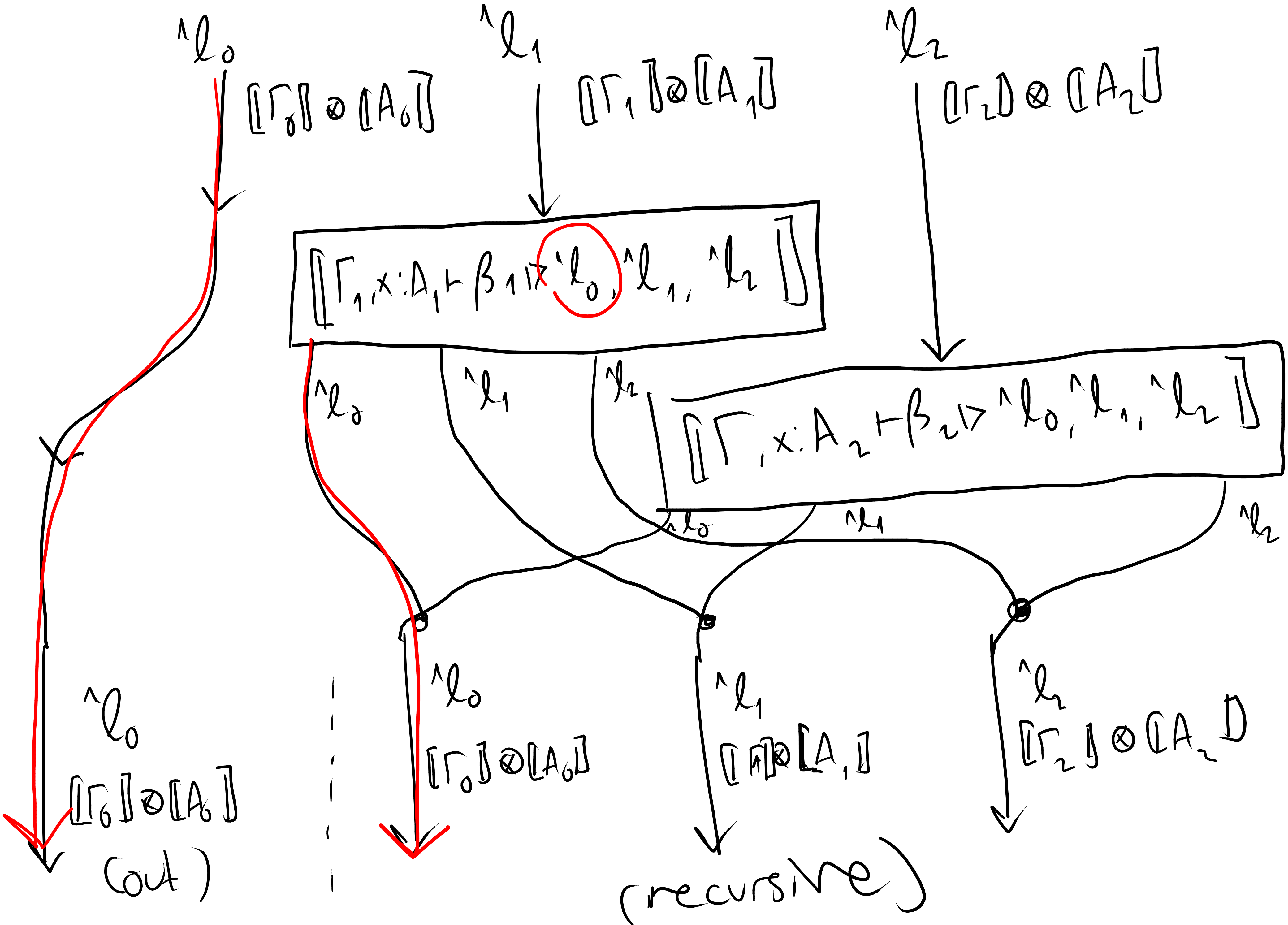

^l₀
[[Γ₀]] ⊗ [[A₀]]
^l₁
[[Γ₁]] ⊗ [[A₁]]
^l₂
[[Γ₂]] ⊗ [[A₂]]
[[Γ₁, x:A₁ ⊢ β₁ ▷ ^l₀, ^l₁, ^l₂]]
^l₀
^l₁
^l₂
[[Γ, x:A₂ ⊢ β₂ ▷ ^l₀, ^l₁, ^l₂]]
^l₀
^l₁
^l₂
^l₀
[[Γ₀]] ⊗ [[A₀]]
(out)
^l₀
[[Γ₀]] ⊗ [[A₀]]
^l₁
[[Γ₁]] ⊗ [[A₁]]
^l₂
[[Γ₂]] ⊗ [[A₂]]
(recursive)

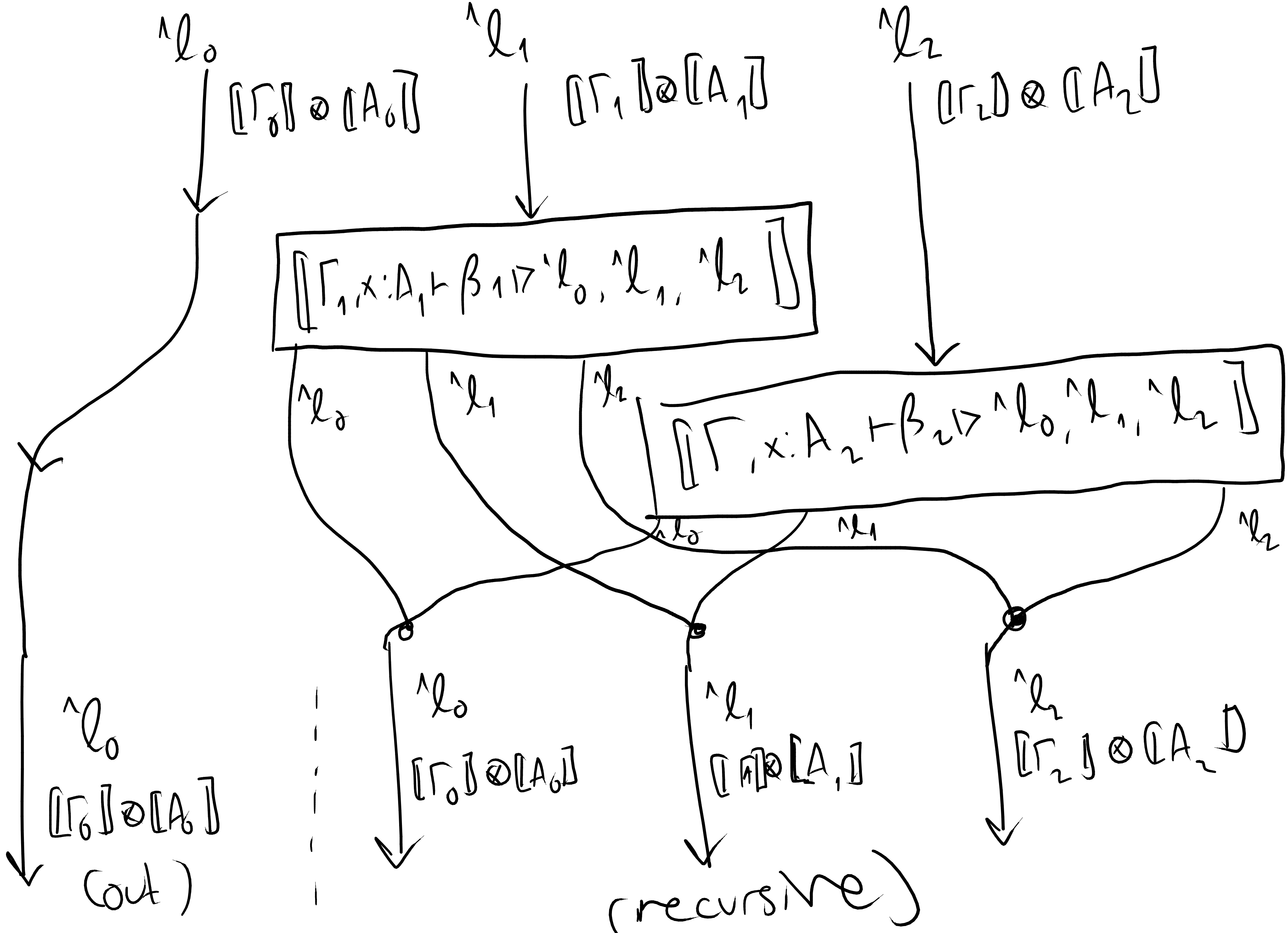

^l₀
[[Γ₀]] ⊗ [[A₀]]
^l₁
[[Γ₁]] ⊗ [[A₁]]
^l₂
[[Γ₂]] ⊗ [[A₂]]
[[Γ₁, x : A₁ ⊢ β₁ ▷ ^l₀, ^l₁, ^l₂]]
^l₀
^l₁
^l₂
[[Γ, x : A₂ ⊢ β₂ ▷ ^l₀, ^l₁, ^l₂]]
^l₀
^l₁
^l₂
^l₀
[[Γ₀]] ⊗ [[A₀]]
(out)
^l₀
[[Γ₀]] ⊗ [[A₀]]
^l₁
[[Γ₁]] ⊗ [[A₁]]
^l₂
[[Γ₂]] ⊗ [[A₂]]
(recursive)


$^\wedge\ell_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
$^\wedge\ell_1$
$[\![\Gamma_1]\!] \otimes [\![A_1]\!]$
$^\wedge\ell_2$
$[\![\Gamma_2]\!] \otimes [\![A_2]\!]$
$[\![\Gamma_1, x : A_1 \vdash \beta_1 \rhd {}^\wedge\ell_0, {}^\wedge\ell_1, {}^\wedge\ell_2]\!]$
$^\wedge\ell_0$
$^\wedge\ell_1$
$^\wedge\ell_2$
$[\![\Gamma, x : A_2 \vdash \beta_2 \rhd {}^\wedge\ell_0, {}^\wedge\ell_1, {}^\wedge\ell_2]\!]$
$^\wedge\ell_0$
$^\wedge\ell_1$
$^\wedge\ell_2$
$^\wedge\ell_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
(out)
$^\wedge\ell_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
$^\wedge\ell_1$
$[\![\Gamma_1]\!] \otimes [\![A_1]\!]$
$^\wedge\ell_2$
$[\![\Gamma_2]\!] \otimes [\![A_2]\!]$
(recursive)


^ℓ0
⟦Γ0⟧ ⊗ ⟦A0⟧
^ℓ1
⟦Γ1⟧ ⊗ ⟦A1⟧
^ℓ2
⟦Γ2⟧ ⊗ ⟦A2⟧
⟦Γ1, x:A1 ⊢ β1 ▷ ^ℓ0, ^ℓ1, ^ℓ2⟧
^ℓ0
^ℓ1
^ℓ2
⟦Γ, x:A2 ⊢ β2 ▷ ^ℓ0, ^ℓ1, ^ℓ2⟧
^ℓ0
^ℓ1
^ℓ2
^ℓ0
⟦Γ0⟧ ⊗ ⟦A0⟧
(out)
^ℓ0
⟦Γ0⟧ ⊗ ⟦A0⟧
^ℓ1
⟦Γ1⟧ ⊗ ⟦A1⟧
^ℓ2
⟦Γ2⟧ ⊗ ⟦A2⟧
(recursive)


$^\ell_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
$^\ell_1$
$[\![\Gamma_1]\!] \otimes [\![A_1]\!]$
$^\ell_2$
$[\![\Gamma_2]\!] \otimes [\![A_2]\!]$
$[\![\Gamma_1, x{:}A_1 \vdash \beta_1 \rhd {^\ell_0}, {^\ell_1}, {^\ell_2}]\!]$
$^\ell_0$
$^\ell_1$
$^\ell_2$
$[\![\Gamma, x{:}A_2 \vdash \beta_2 \rhd {^\ell_0}, {^\ell_1}, {^\ell_2}]\!]$
$^\ell_0$
$^\ell_1$
$^\ell_2$
$^\ell_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
(out)
$^\ell_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
$^\ell_1$
$[\![\Gamma_1]\!] \otimes [\![A_1]\!]$
$^\ell_2$
$[\![\Gamma_2]\!] \otimes [\![A_2]\!]$
(recursive)


$^\wedge l_0$
$^\wedge l_1$
$^\wedge l_2$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
$[\![\Gamma_1]\!] \otimes [\![A_1]\!]$
$[\![\Gamma_2]\!] \otimes [\![A_2]\!]$
$[\![\Gamma_1, x:A_1 \vdash \beta_1 \rhd {}^\wedge l_0, {}^\wedge l_1, {}^\wedge l_2]\!]$
$[\![\Gamma, x:A_2 \vdash \beta_2 \rhd {}^\wedge l_0, {}^\wedge l_1, {}^\wedge l_2]\!]$
$^\wedge l_0$
$^\wedge l_1$
$^\wedge l_2$
$^\wedge l_0$
$^\wedge l_1$
$^\wedge l_2$
$^\wedge l_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
(out)
$^\wedge l_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
$^\wedge l_1$
$[\![\Gamma_1]\!] \otimes [\![A_1]\!]$
$^\wedge l_2$
$[\![\Gamma_2]\!] \otimes [\![A_2]\!]$
(recursive)


$^\wedge l_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
$^\wedge l_1$
$[\![\Gamma_1]\!] \otimes [\![A_1]\!]$
$^\wedge l_2$
$[\![\Gamma_2]\!] \otimes [\![A_2]\!]$
$[\![\Gamma_1, x{:}A_1 \vdash \beta_1 \rhd {}^\wedge l_0, {}^\wedge l_1, {}^\wedge l_2]\!]$
$^\wedge l_0$
$^\wedge l_1$
$^\wedge l_2$
$[\![\Gamma, x{:}A_2 \vdash \beta_2 \rhd {}^\wedge l_0, {}^\wedge l_1, {}^\wedge l_2]\!]$
$^\wedge l_0$
$^\wedge l_1$
$^\wedge l_2$
$^\wedge l_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
(out)
$^\wedge l_0$
$[\![\Gamma_0]\!] \otimes [\![A_0]\!]$
$^\wedge l_1$
$[\![\Gamma_1]\!] \otimes [\![A_1]\!]$
$^\wedge l_2$
$[\![\Gamma_2]\!] \otimes [\![A_2]\!]$
(recursive)

Part III: Concrete Models

SSA is Freyd Categories

SSA is Freyd Categories

with Elgot Structure*

Models need:

Models need:

- Freyd category

Models need:

- Freyd category
- w/ coproducts

Models need:

- Freyd category
- w/ coproducts
- w/ fixpoints, Elgot

Monads!

Monads!

$$m : \mathrm{Set} \to \mathrm{Set}$$

Monads!

$m : \mathrm{Set} \to \mathrm{Set}$

$\mathrm{pure} : \forall \alpha.\ \alpha \to m\,\alpha$

Monads!

$m : \mathrm{Set} \to \mathrm{Set}$

$\mathrm{pure} : \forall \alpha.\ \alpha \to m\,\alpha$

$\mathrm{bind} : \forall \alpha\,\beta.\ m\,\alpha \to (\alpha \to m\,\beta) \to m\,\beta$

Monads!

$m : \mathrm{Set} \to \mathrm{Set}$

$\mathrm{pure} : \forall \alpha.\ \alpha \to m\,\alpha$

$\mathrm{bind} : \forall \alpha\,\beta.\ m\,\alpha \to (\alpha \to m\,\beta) \to m\,\beta$

$m = \mathrm{Option}$

$\mathrm{pure} = \mathrm{Some}$

$\mathrm{bind}\ \mathrm{None}\ f = \mathrm{None}$

$\mathrm{bind}\ (\mathrm{Some}\ a)\ f = f\ a$

Monads!

$$f : \alpha \to m\beta \in \mathrm{Set}_m(\alpha, \beta)$$

Monads!

$f : \alpha \to m\beta$ $\quad$ $g : \beta \to m\gamma$

Monads!

$f : \alpha \to m\beta \qquad g : \beta \to m\gamma$

$f >\!\!> g : \alpha \to m\gamma$

$= \lambda a.\ \mathrm{bind}\ \underbrace{(f\ a)}_{m\beta}\ \ g$

Monads Induce Freyd Categories

Define:

$$\mathrm{Set}_{m_1}(\alpha, \beta) = \{ f ; \mathrm{pure} \mid f : \alpha \to \beta \}$$

Monads Preserve Coproducts

$$\mathrm{inl}' = \mathrm{inl} \,;\, \mathrm{pure} : \alpha \to M(\alpha + \beta)$$

Monads Preserve Coproducts

$$\text{inl}' = \text{inl} \,;\, \text{pure} : \alpha \to M(\alpha + \beta)$$

$$\text{inr}' = \text{inr} \,;\, \text{pure} : \beta \to M(\alpha + \beta)$$

Monads Preserve Coproducts

$\mathrm{inl}' = \mathrm{inl} \,;\, \mathrm{pure} : \alpha \to M(\alpha + \beta)$

$\mathrm{inr}' = \mathrm{inr} \,;\, \mathrm{pure} : \beta \to M(\alpha + \beta)$

$f : \alpha \to M\gamma \qquad g : \beta \to M\gamma$

$[f, g] : \alpha + \beta \to M\gamma$

Elgot Monads

Given $f : \alpha \to M(\beta + \alpha)$

Want $f^{\dagger} : \alpha \to M\beta$

s.t. Set_M Elgot

Elgot Monads

Given $f : \alpha \to \text{Option}(\beta + \alpha)$

Define: $f^{+}\, a = $ if $\exists n,\ f^{(n)}(a) = \text{some } b$

then some b

else none

Elgot Monads

Given $f : \alpha \to \mathrm{Option}(\beta + \alpha)$

Define: $f^{\dagger}\, a = \text{if } \exists n,\ f^{(n)}(a) = \mathrm{some}(\mathrm{inl}\ b)$
$\text{then some } b$
else none

Where: $f^{(0)}\, a = \mathrm{some}\ (\mathrm{inr}\ a)$

$$f^{(n+1)}\, a = \mathrm{bind}\ (f^{(n)}\, a) \begin{bmatrix} \lambda \mid \mathrm{inl}\ b \Rightarrow \mathrm{some}\ \mathrm{inl}\ b \\ \mid \mathrm{inr}\ a \Rightarrow f\ a \end{bmatrix}$$

Elgot Monads

Given $f : \alpha \to \text{Option}(\beta + \alpha)$

Define: $f^{\dagger}\, a = \text{if } \exists n,\ f^{(n)}(a) = \text{some}(\text{inl } b)$

$\text{then some } b$

else none

Note: NOT computable!

Where: $f^{(0)}\, a = \text{some } (\text{inr } a)$

$$f^{(n+1)}\, a = \text{bind } (f^{(n)}\, a) \begin{bmatrix} \lambda \mid \text{inl } b \Rightarrow \text{some inl } b \\ \mid \text{inr } a \Rightarrow f\ a \end{bmatrix}$$

Monad Transformers

Monad Transformers

$$\text{StateT}\ \sigma\ m\ \alpha := \sigma \to m(\alpha \times \sigma)$$

Monad Transformers

$$\text{StateT}\ \sigma\ m\ \alpha := \sigma \to m(\alpha \times \sigma)$$

$$\text{bind}_{\text{StateT}}\ \underset{\sigma \to m(\alpha \times \sigma)}{\underset{\uparrow}{a}}\ \ \underset{\alpha \to \sigma \to m(\beta \times \sigma)}{\underset{\uparrow}{f}}$$

Monad Transformers

$$\text{StateT}\ \sigma\ m\ \alpha := \sigma \to m(\alpha \times \sigma)$$

$$\text{bind}_{\text{StateT}}\ \underset{\underset{\sigma \to m(\alpha\times\sigma)}{\nearrow}}{a}\ \ \underset{\underset{\alpha \to \sigma \to m(\beta\times\sigma)}{\uparrow}}{f} := \lambda s:\sigma.\ \text{bind}_m\ \overbrace{(a\ s)}^{m(\alpha\times\sigma)}\ \underbrace{(\lambda(a,s).\ f\ a\ s)}_{\alpha\times\sigma \to m(\beta\times\sigma)}$$

Monad Transformers

$$\mathrm{StateT}\ \sigma\ m\ \alpha := \sigma \to m(\alpha \times \sigma)$$

$$\mathrm{bind}_{\mathrm{StateT}}\ a\ f := \lambda s{:}\sigma.\ \mathrm{bind}_m\ (a\ s)\ (\lambda(a,s).\ f\ a\ s)$$

$$\mathrm{ReaderT}\ \rho\ m\ \alpha := \rho \to m\ \alpha$$

$$\mathrm{bind}_{\mathrm{ReaderT}}\ a\ f := \lambda r{:}\rho.\ \mathrm{bind}_m\ (a\ r)\ (\lambda a.\ f\ a\ r)$$

Basic Heap Model

Basic Heap Model

$\mathrm{Heap} := \mathbb{N} \xrightarrow{\mathrm{Fin}} \mathbb{N}$

Basic Heap Model

$$\text{Heap} := \mathbb{N} \xrightarrow{\text{Fin}} \mathbb{N}$$

$$m\ \alpha := \text{StateT}\ \ \text{Heap}\ \ \text{Option}$$

Basic Heap Model

$\text{Heap} := \mathbb{N} \xrightarrow{\text{Fin}} \mathbb{N}$

$m\ \alpha := \text{StateT Heap Option}$

$\text{load } n\ s := \text{if } n \in s \text{ then some } (s\,n, s)$
$\qquad\qquad\qquad\qquad\qquad \text{else none}$

Basic Heap Model

$\text{Heap} := \mathbb{N} \xrightarrow{\text{Fin}} \mathbb{N}$

$m\ \alpha := \text{StateT Heap Option}$

$\text{load}\ n\ s := \text{if}\ n \in s\ \text{then some}\ (s\,n, s)$
$\qquad\qquad\qquad\qquad\qquad\qquad \text{else none}$

$\text{Store}\ (\ell, n)\ s := \text{some}(((), [\ell \mapsto n]\, s)$

Questions?

jeg74@cl.cam.ac.uk

github.com/imbrem