From Hilbert space to Dilbert space: Context semantics as a language for games and flow analysis

Harry Mairson
Brandeis University

In memory of my father,
Theodore Mairson (1919-2002)

From: ICFP03 <shivers+icfp@cc.gatech.edu>
Date: Tue May 20, 2003 1:03:58 AM Canada/Eastern
To: "Harry Mairson" <mairson@cs.brandeis.edu>
Subject: Your ICFP03 paper 236
Reply-To: shivers@cc.gatech.edu

Dear Harry,

I'm very sorry to tell you that your submission #236
How Light Is Safe Recursion? Compositional Translations between Languages of Polynomial Time
to ICFP'03 was *rejected*.

It was a strong field -- there were 94 submissions altogether, of which we could accept no more than 24. Above and beyond the numbers, the members of the program committee all remarked upon the consistently high quality of the submissions this year; we were forced to reject many fine, publishable papers.

You will find attached reviews of your paper; you can also find them at URL http://edas.cc.gatech.edu/cgi-bin/PaperShow.cgi?m=236 The program committee and their colleagues lavished a huge amount of time and care on these reviews, so I hope that you will find that they offer constructive ideas for how to improve your paper.

Thank you for submitting to ICFP; I am sorry to have to disappoint you.

Best wishes,

Olin Shivers
Program Chair

===== Review =====

*** Rating (10=highest; 1=lowest): 7

*** **Confidence (7=high confidence; 1=low confidence): 1**

*** Comments (actual review): In Figure 1, rules "weak" and !1 appear to have typos.

===== Review =====

*** Rating (10=highest; 1=lowest): 5

*** **Confidence (7=high confidence; 1=low confidence): 3**

*** Summary (summary of paper): This ia a rather technical paper that shows a translation from one polynomial time language to another.

The paper is rather hard to read and follow. It concerns Bellantini and Cook's language BC, and Girard's light linear logic. While there are

*** Rating (10=highest; 1=lowest): 8

*** **Confidence (7=high confidence; 1=low confidence): 3**

*** Summary (summary of paper): Neergaard and Mairson investigate the construction of a compositional translation of Bellantoni-Cook's functional combinator language BC into Girard-Asperti's Light Affine Logic LAL. Both these languages capture polynomial-time functions.

The paper claims three new results: Murawski and Ong have devised a compositional translation of the linear fragment BC- into LAL. First Neergaard and Mairson show that this fragment can be evaluated in logspace. Second they show that the method used by Murawski and Ong cannot be extended to a translation of the full BC language into LAL. Third they give the first compositional translation of full BC into LAL. This is a new construction where the target code simulates a version of a SECD-machine.

*** Comments (actual review): The general aim behind the research is to develop a better understanding of the essence of the polynomial time complexity class. It is seen as a step in that direction to explain how programming languages that capture p-time can simulate each other. The paper has a rather technical character. In various places remarks are given on what is revealed in this research about the general principles that lie behind. I think that the analysis of this with advantage could have been highlighted and further discussed.

The logspace evaluation trades space for time by handling recursion in a non-standard way without storing the call stack. The authors say they have an implementation on their homepage, the reviewer couldn't find it. Beckmann and Weiermann have given a rewriting system for BC and have shown that it is only polynomial time when evaluated call by value. Therefore to translate the full BC into LAL, it is necessary to capture the evaluation order within the LAL encoding. This requires other methods than given by Murawski and Ong. A SECD machine can be represented as a LAL proof net. The translation takes a BC program into an equivalent LAL proof net simulating a SECD-like machine.

A few opening remarks about the general idea behind the construction in section 5 would improve readability. I would have appreciated if it had been emphasized and specified in more detail, how the implementation handles the two central issues: normal vs. safe and capturing of evaluation order.

The paper reveals aspects of connections between poly-time languages, and places the work in a broader scientific context. A more clear separation of principles in the actual methods used and the technical details (especially in the last part) would let the contribution stand out more and would make it more accessible to a broader audience.

From Hilbert space to Dilbert space: Context semantics as a language for games and flow analysis

Harry Mairson
Brandeis University

In memory of my father,
Theodore Mairson (1919-2002)

Do laptop presentations make you feel typeset, justified, filed, stamped, indexed, briefed, or de-briefed?

... sometimes when I read stuff put together with some markup language, I feel like a typographical **Prisoner** ...

Would you rather be typeset, justified, filed, stamped, indexed, briefed, or de-briefed?

Colored pens are friendlier, warmer, easier to read (if you write neatly!), more tactile, human, and smell better.

"I am not a number,
I am a free man!"

Second Letter.

Fellow Weavers!

SINCE I last wrote to you, I have been considering within myself what class of persons amongst you is injured by Machinery? I shall therefore ask each of them separately, and desire them to put the question to their own heart.

Who is injured by Machinery? As I understand that the *Colliers* were particularly active in the late wicked outrages, I shall address myself first to them:—

Colliers! Are ye injured by the demand for Coals to work the Steam-Engines, that set in motion the Machinery? Do the Steam-Engines or Machinery consume food?

No,—but they save that which must otherwise be consumed by *Horses*. They are even named from this, as the ten-horse power saves the food of ten horses, for the use of man.

Canal-Diggers! Are ye injured? Have not the canals been dug to carry our goods and our coals to other markets, and to bring us food?—At those markets we could not undersell other persons, unless the goods were made cheap by machinery; and both the goods and the coals cheaper by canals. If we could not sell our goods, how are we to pay for our food?

Spinners and *Weavers*, are ye injured? Least of all persons are ye entitled to complain. For four times your number are employed since the invention of machinery:—and why? because poor little children, by the help of machinery, can earn their own livelihood, and it is easy to rear a family.

Smiths, Carpenters, Engine-makers, in short tradesmen of all kinds, which of you is injured by having clothes made cheap, and employment more plentiful?

But if here and there one man has his employment because another makes goods cheaper by improvements in machinery, must it not be so in all trades?

Which of us will not go to the cheap baker, the cheap carpenter, and why not to the cheap manufacturer?—When all but a few receive benefit, what right have these few to complain?

What right has any man to prevent thousands from buying their goods cheap?

Besides in our great concerns cheapness in any one branch gives employment to all. If the thread is spun cheap, the whole piece, when woven, may be sold cheaper, and thus there will be a greater demand both for spinning and weaving:—So if the weaving is cheaper, there will be a greater demand for thread, and more work for spinners. What right then has any man, because his branch of work is made cheaper, to rise and put others out of employ?

I will tell you an instance, to shew you how unreasonable this is, and have done.

A New Bridge was built over the river Thames, about a quarter of a mile wide.

Before it was built, every body was forced either to go round some miles by the Old Bridge, or to go over himself, his horses, and carts by the Old Ferry: and sometimes they were stopped by the wind:—sometimes by the tide:—sometimes the horses were unwilling to go on board the ferry boat, and frequently a horse was lamed in getting on board.

And besides this, they had to pay a *considerable Toll*. All this trouble, loss of time, and expence, were saved to thousands, and thousands, by the New Bridge.—Yet the ferry-men had the impudence to make a riot!

They were not more unreasonable than those who now provoke a riot, because some weavers have contrived to make goods cheaper by improvements in machinery, and have thus found a *New Bridge* for sending our GOODS OVER THE WORLD.

I am, as before,

Your Well-Wisher,

An Old Weaver.

W. Cowdroy, Printer, Hunter-lane, Manchester.

WHEREAS,

Several EVIL-MINDED PERSONS have assembled together in a riotous Manner, and DESTROYED a NUMBER of

FRAMES,

In different Parts of the Country:

THIS IS

TO GIVE NOTICE,

That any Person who will give Information of any Person or Persons thus wickedly

BREAKING THE FRAMES,

Shall, upon CONVICTION, receive

50 GUINEAS

REWARD.

And any Person who was actively engaged in RIOTING, who will impeach his Accomplices, shall, upon CONVICTION, receive the same Reward, and every Effort made to procure his Pardon.

☞ Information to be given to Messrs. COLDHAM and ENFIELD.

Nottingham, March 16, 1811.

I have always wanted to do this...

What is context semantics?

[Gonthier, Abadi, Lévy — early 1990s]

Derived from Girard's geometry of interaction, only without any math, thus "from Hilbert to Dilbert."

> Hilbert spaces, C*-algebras, linear operators, execution formula,... → graphs, stacks, pairs, transitive closure, basic discrete math,...

Semantics of?... PCF, λ-calculus, fragments of linear logic,...

Context semantics is...

- a semantics of programs (PCF) and proofs (LL, MELL).
- an exact form of flow analysis, computed statically
  "Does call site χ ever call function φ?"
- an interactive way of talking about Böhm trees and normal forms.
- an explicit mechanization of the kinds of interactions found in game semantics [AJM, HON, Mc].
- a distilled form of Lévy's labelled λ-calculus (also a form of program analysis).
- an exact semantics of optimal reduction and maximal sharing, prouding a proof of its algorithmic correctness, with added insights in the presence of linear types.

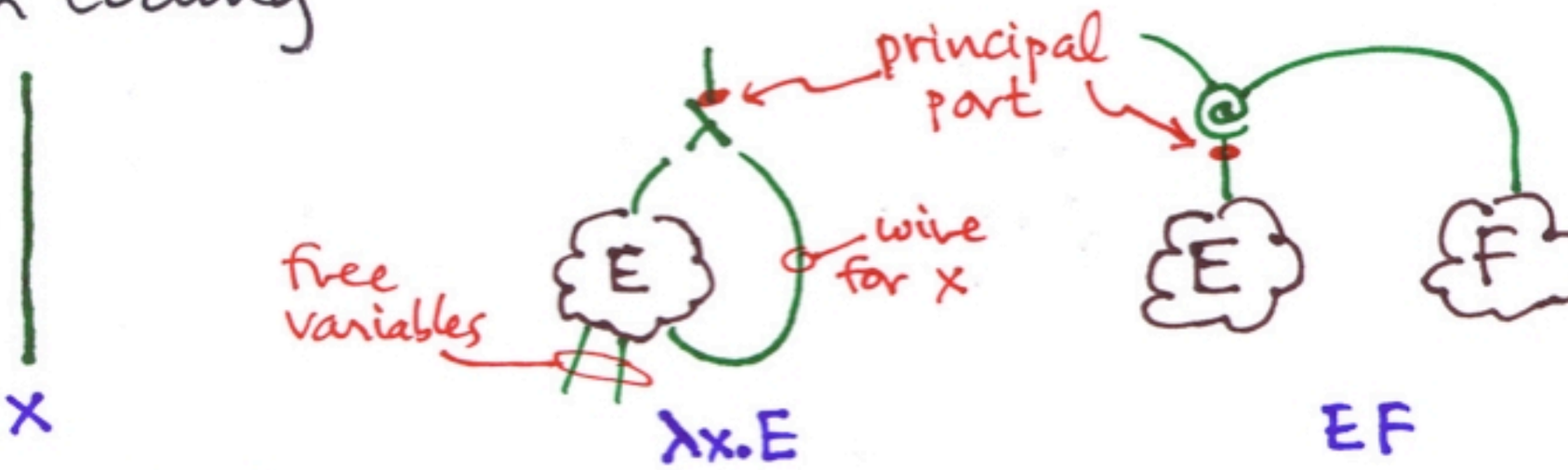
Outline

- **Linear λ-calculus:** where graph reduction and context semantics are easy.
- **Sharing:** boxes and sharing, CBN coding, global, insular, and local graph reduction.
- **Readback:** how to construct the normal form of a (λ, PCF)-term from its semantics. Relation to flow analysis, games, types.
- **Labels:** labelled reduction, paths and flows, labels and contexts.

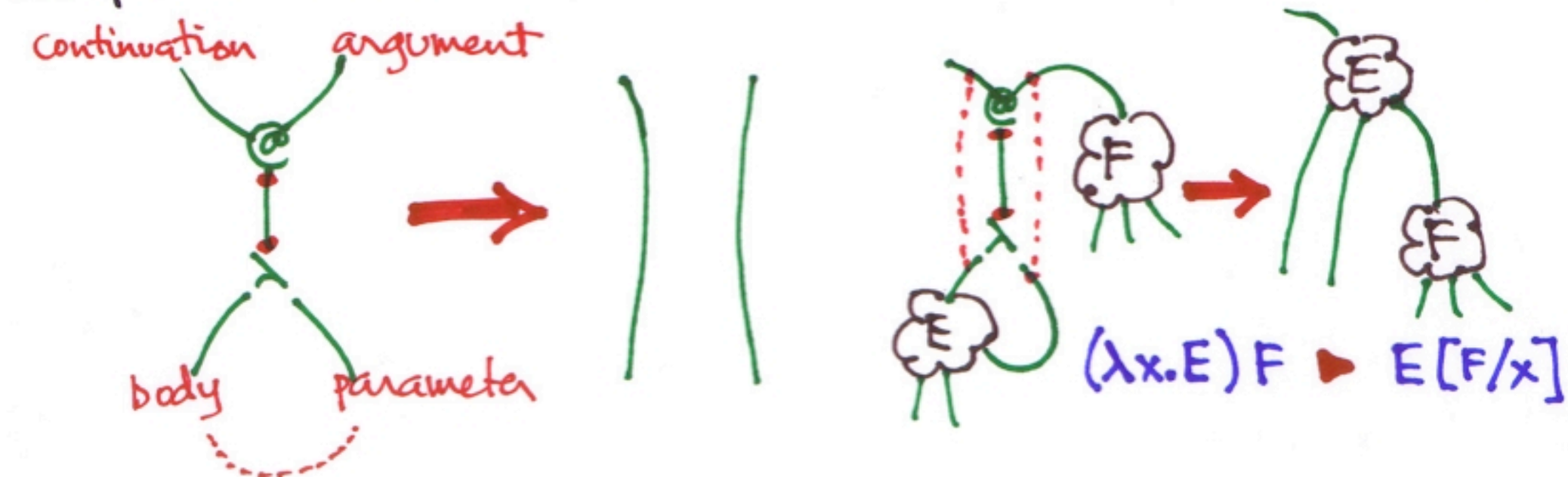
Linear λ-calculus (variable occurs ≤ 1 time)

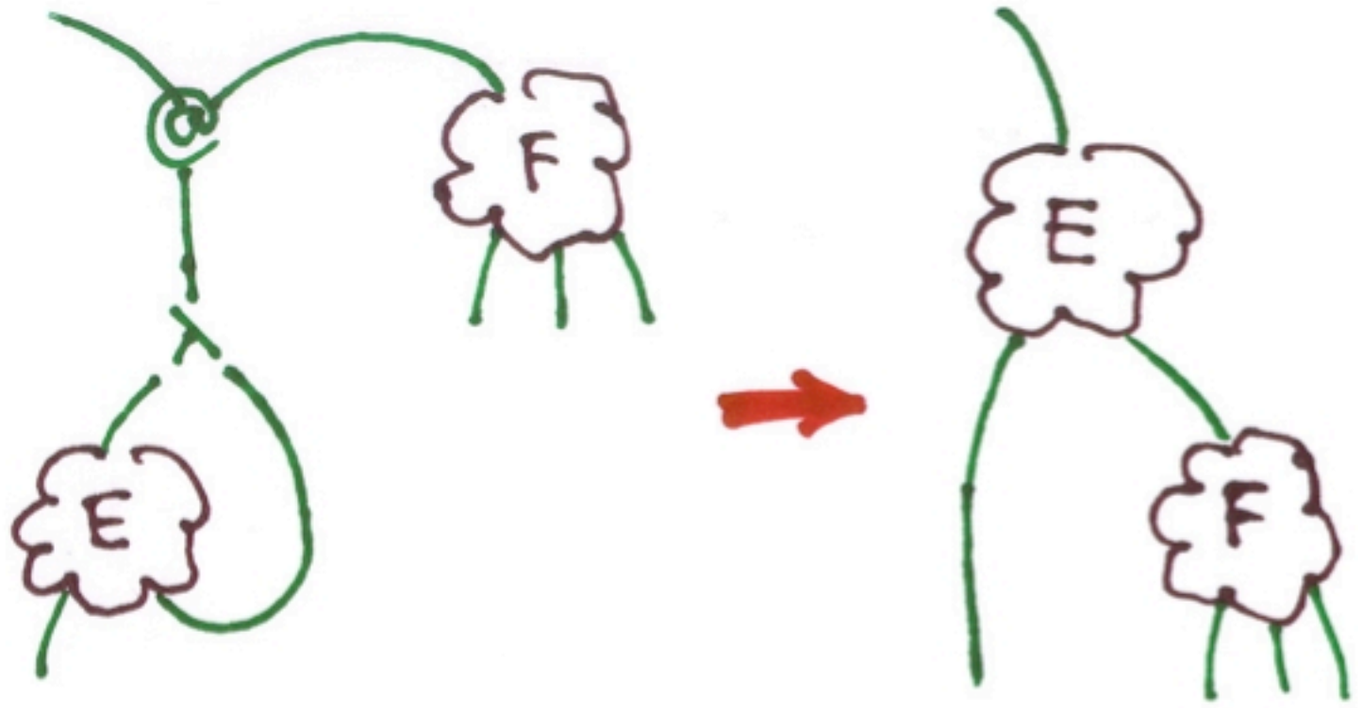
$$V \to x \mid y \mid z \mid \dots$$
$$E \to V \mid \lambda V.E \mid EE$$

Graph coding



Graph reduction





$(\lambda x.E)\,F$

$\blacktriangleright\; E[F/x]$

Proposition. $S \blacktriangleright^* T$ iff $G_S \to^* G_T$.

(The graphs are an equivalent, alternative syntax.)

Now, what about a semantics that this graph reduction can be said to preserve?

A graph has ports at its root p,
and at each of its free variables π_v.

CONTEXT SEMANTICS OF A LINEAR λ-TERM:

$$\mathcal{C}_E = \{(\langle c, \pi\rangle, \langle c', \pi'\rangle) \mid$$

symmetric!
reversible!

context c enters at port π, and exits as context c' at port π' $\}$

Proposition. If $E \blacktriangleright^* E'$, then $\mathcal{C}_E = \mathcal{C}_{E'}$

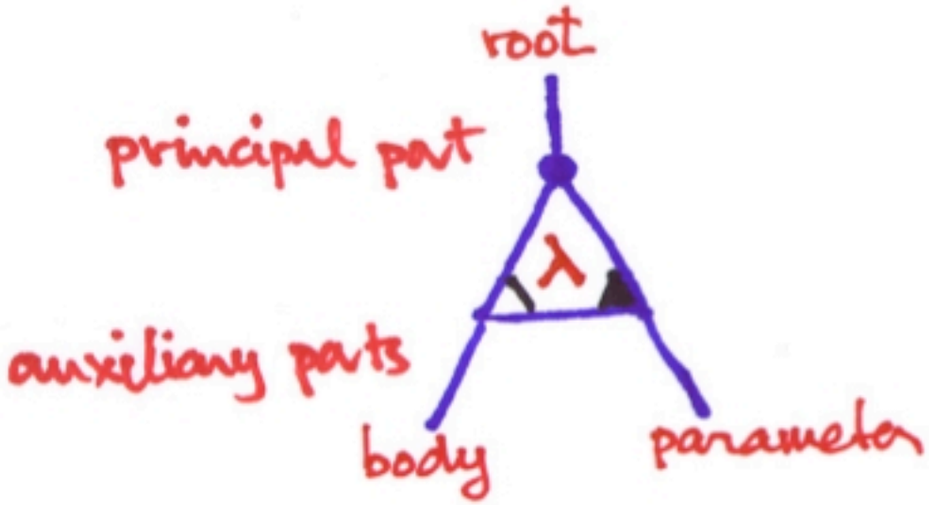
That's all fine and good — but what does the semantics intuitively tell us about a λ-term?

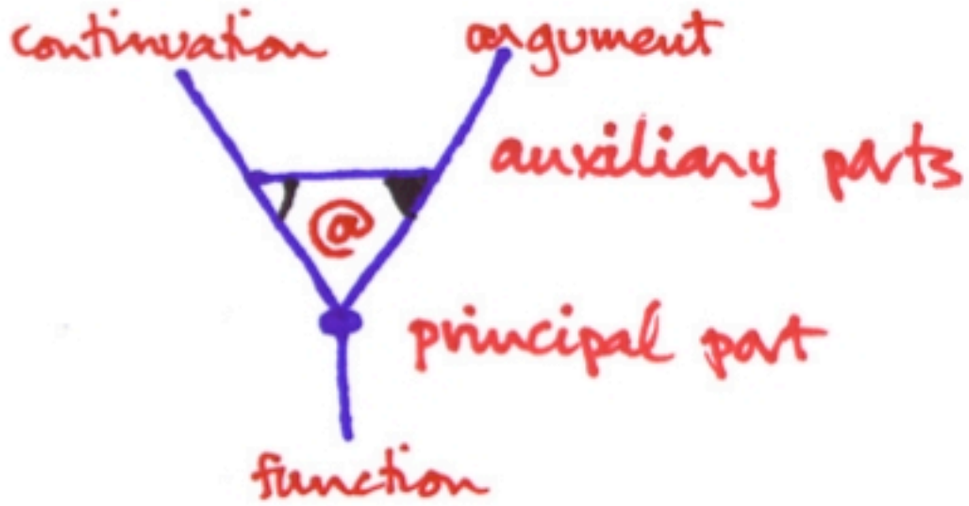
[Not only philosophers worry about "the meaning of meaning."]

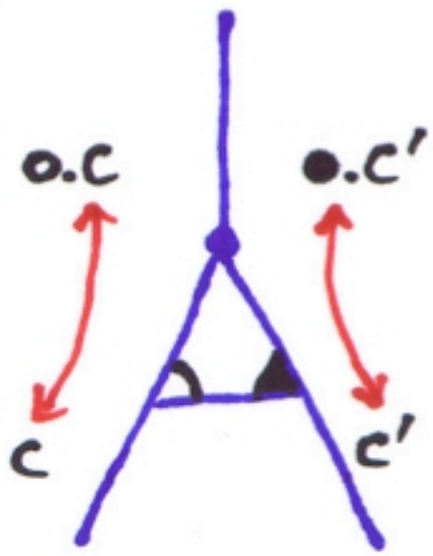
Context semantics for linear λ-calculus

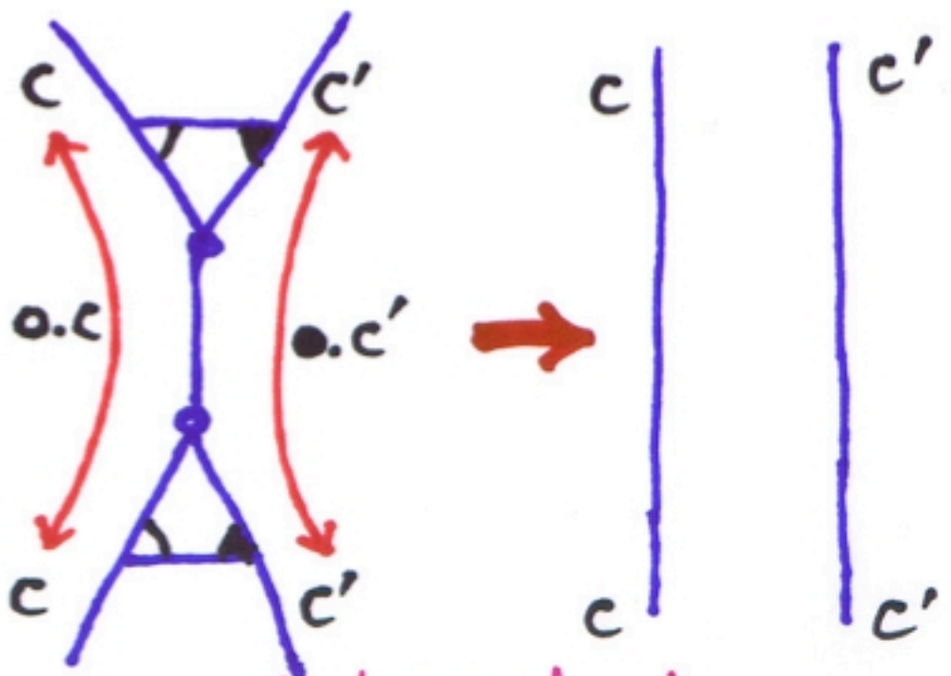
$\Sigma = \{\circ, \bullet\}$ tokens $\qquad c \in \Sigma^*$ contexts

Contexts are transformed by graph nodes @, λ:







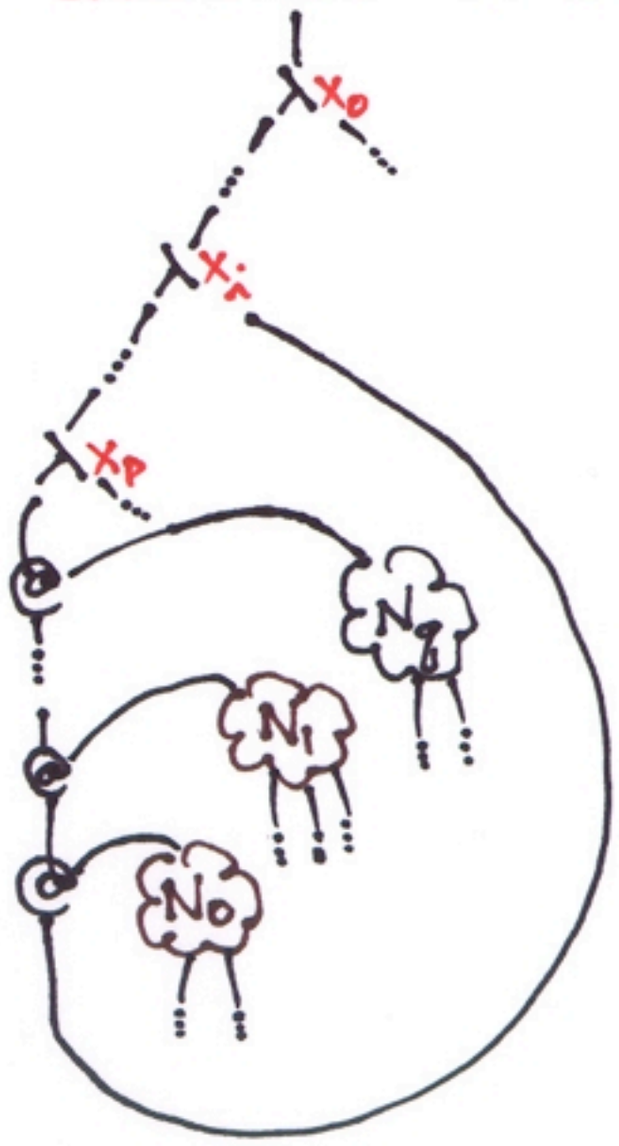


... semantics preserved by reduction ...

Linear λ-calculus is strongly normalizing since reduction  decreases the graph size

Context semantics thus describes the normal form — but how?

EXAMPLE. $N \equiv \lambda x_0.\lambda x_1.\cdots \lambda x_p.(\cdots((x_i\ N_0)\ N_1)\cdots)\ N_q$



INPUT: $o^{p+1} \equiv \underbrace{oo\cdots o}_{p+1}$ at the root

OUTPUT: $o^i \bullet o^{q+1}$ at the root

What's learned? Head variable, number of arguments

x_i $q+1$

Now input $o^i \bullet\ o^i \bullet o^?$ to learn...

Head variable, number of arguments in subterm N_j

$$N \equiv \lambda x_0. \cdots \lambda x_p.\ x_i\ N_0 \cdots \underbrace{(\lambda y_0. \cdots \lambda y_r.\ w\ F_0 \cdots F_s)}_{N_j} \cdots N_q$$

INPUT at root	o^{p+1}	"Head variable of N?"
OUTPUT at root	$o^i \bullet o^{q+1}$	"x_i"
INPUT at root	$o^i \bullet o^j \bullet o^{r+1}$	"Head variable of N_j?"
OUTPUT at root	$o^i \bullet o^j \bullet o^k \bullet o^{s+1}$	"y_k"
	or $o^k \bullet o^{s+1}$	or "x_k"
	or o^{s+1}	or "w" (free)

What is this? **Readback**: how to read back the normal form of a λ-term from its context semantics

... or equivalently ...

A **game** between an **Opponent** (computational environment or context) who wants to learn the λ-term known by a **Player**.

Readback (cont.)

$E \equiv \lambda x_0. \cdots . \lambda x_p . \; x_i \, E_0 \cdots E_q$ (assume E closed)

$\text{subterms}(E) = \{E\} \cup \bigcup_j \text{subterms}(E_j)$

$\text{headvariables}(E) = \{x_i\} \cup \bigcup_j \text{headvariables}(E_j)$

Uniqueness of paths: every subterm, head variable has a unique address in $\{o, \bullet\}^*$

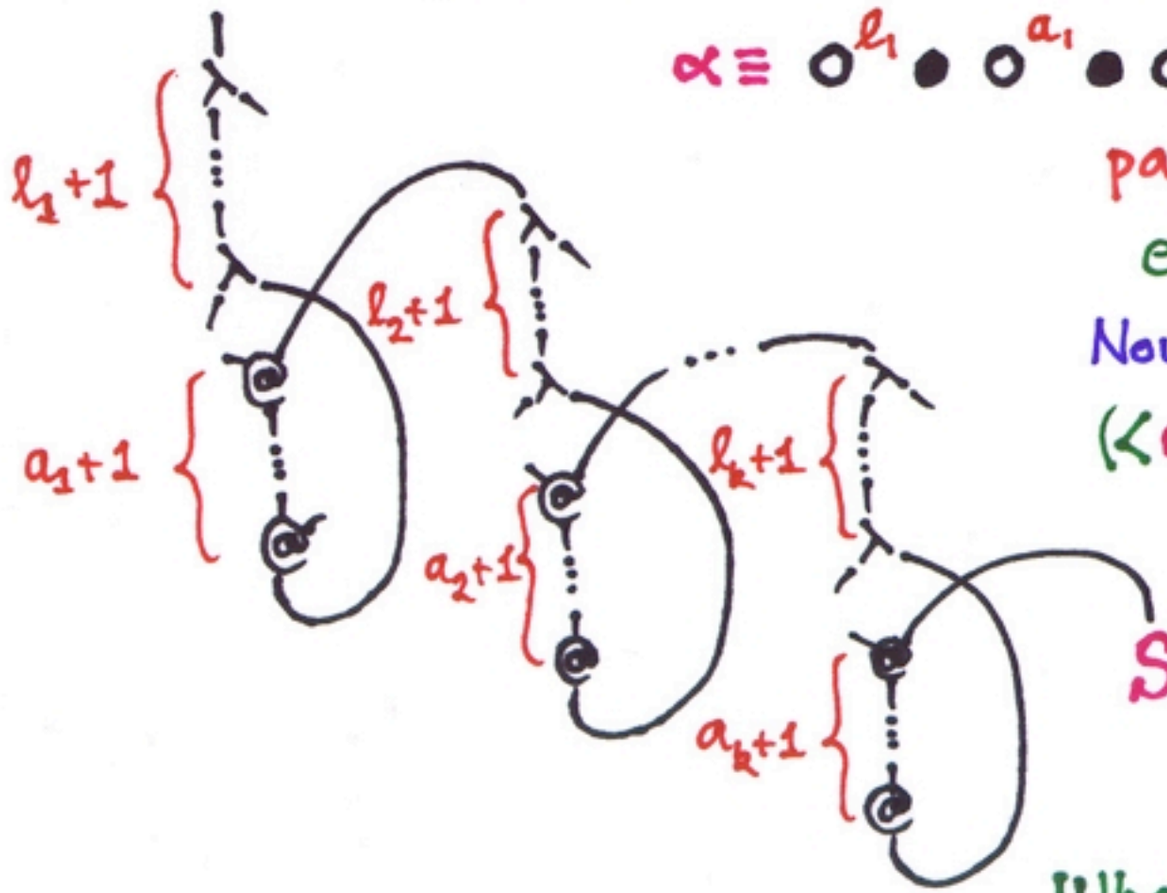
$\alpha \equiv o^{l_1} \bullet o^{a_1} \bullet o^{l_2} \bullet o^{a_2} \bullet \cdots o^{l_k} \bullet o^{a_k} \bullet$

parity = # $\bullet$ in context

even → subterm / odd → head variable

Now, find smallest k where

$(\langle \alpha o^k, \text{root} \rangle, \langle \alpha' o^{k'}, \text{root} \rangle) \in C_E$



λ-binders at top level of S

head variable of S

arguments of head variable

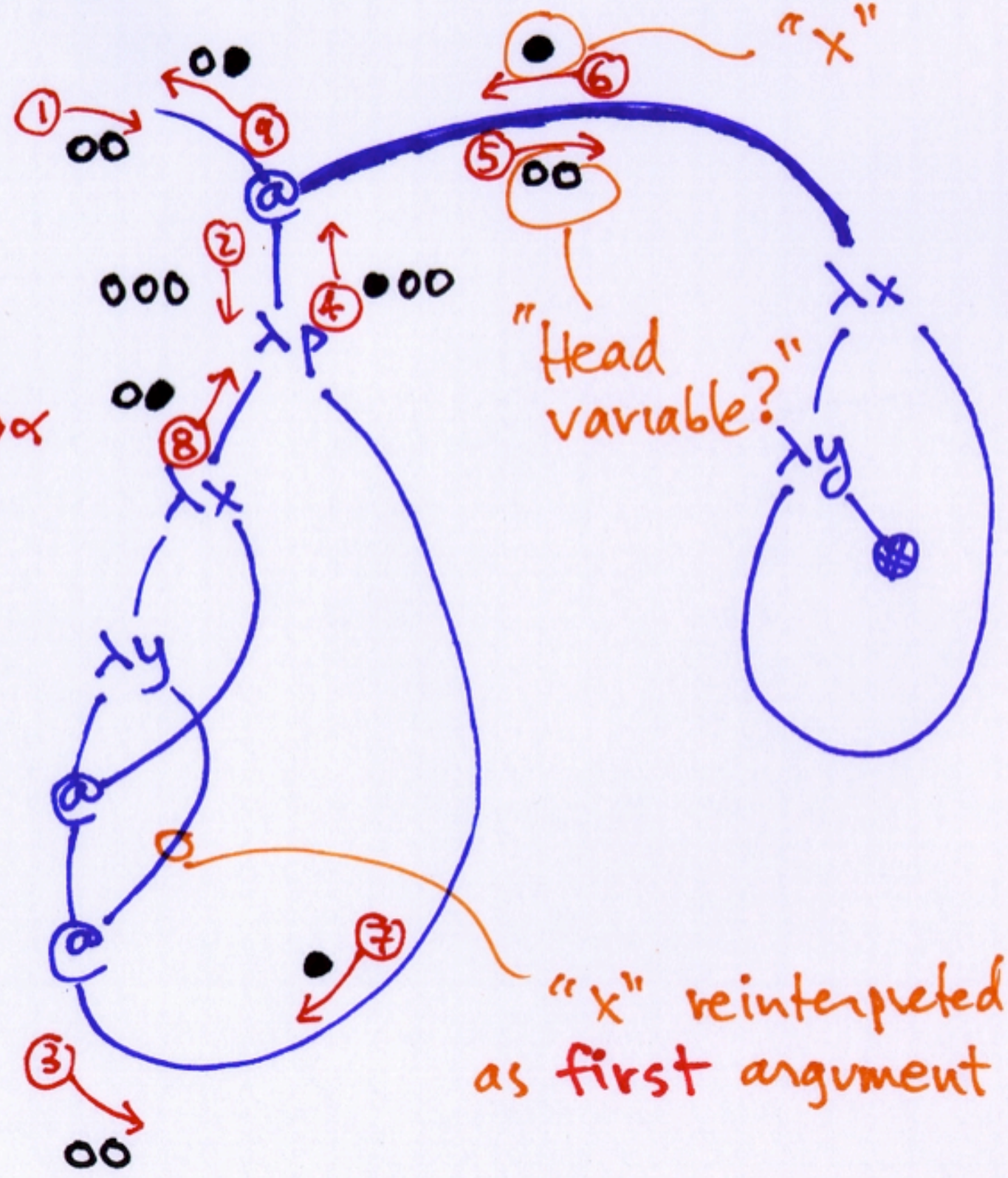
What is a "legal" choice of α'?
What head variables are in view?

Flow analysis: linear terms

not T ≡ (λp. λx. λy. p y x) (λx. λy. x)

Bool ≡ $\circ\alpha\to\circ\alpha\to\alpha$

not: $(\alpha\to\alpha\to\alpha)\to\circ\alpha\to\circ\alpha\to\alpha$



Sharing

How do we implement multiple references?

Move graph reduction technology...

to delineate
sharable expressions

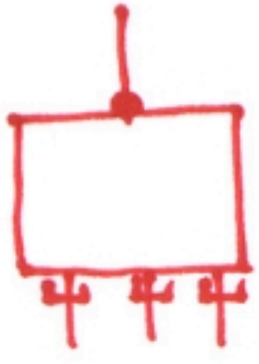
with sharing nodes
to so share them

croissants to open boxes
so their contents can be
used...

brackets so one box
can absorb another
(substitution)

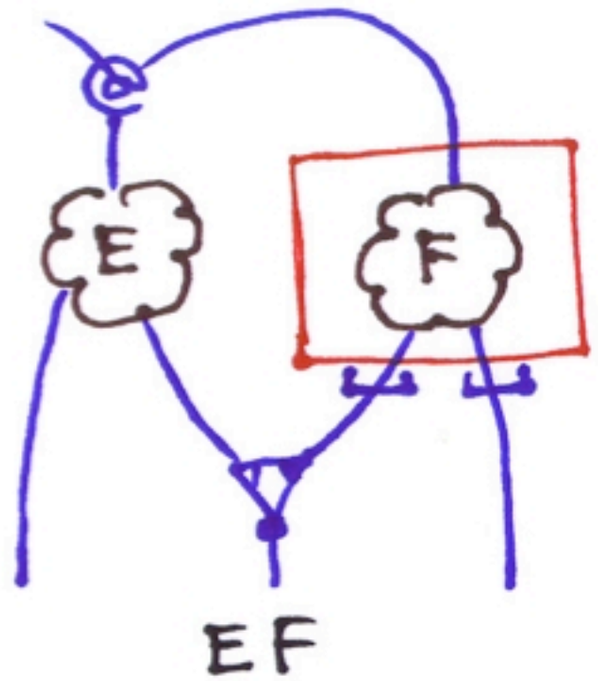
Call-by-name (CBN) coding (one among many)

marks a
variable occurrence

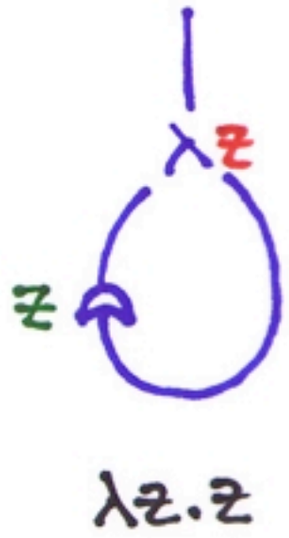


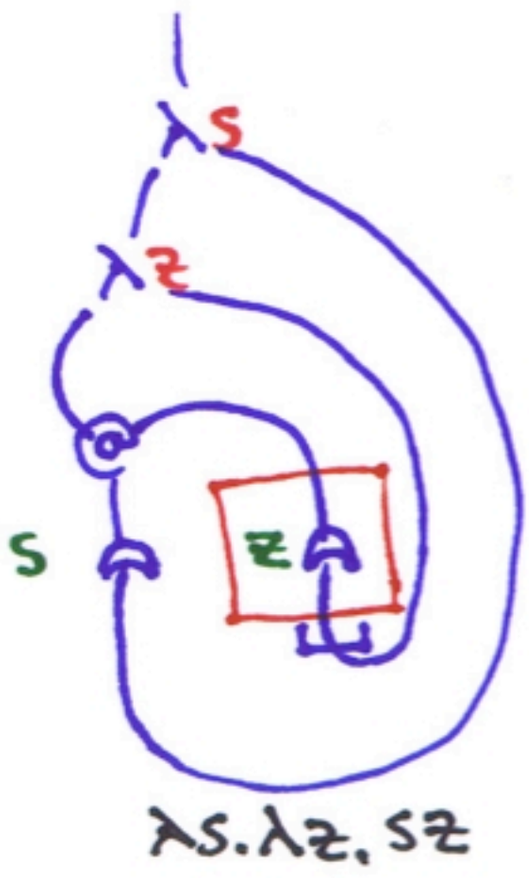
encloses every
argument of a
function application

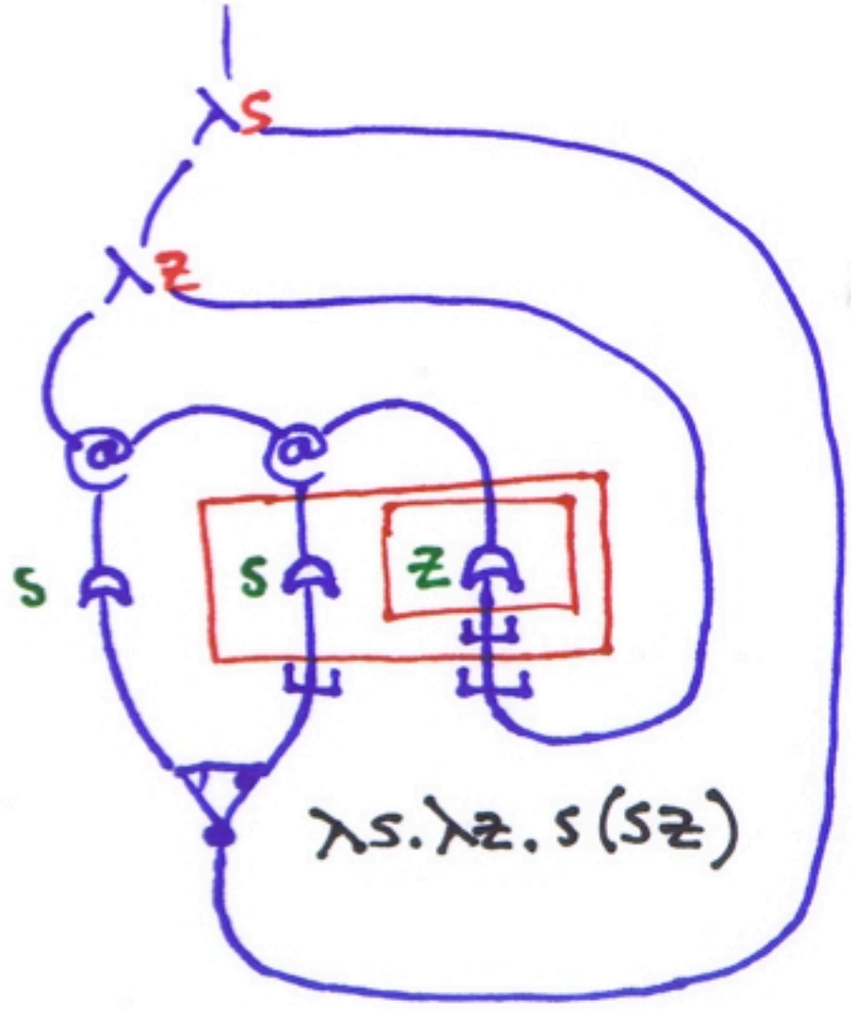
CBN coding



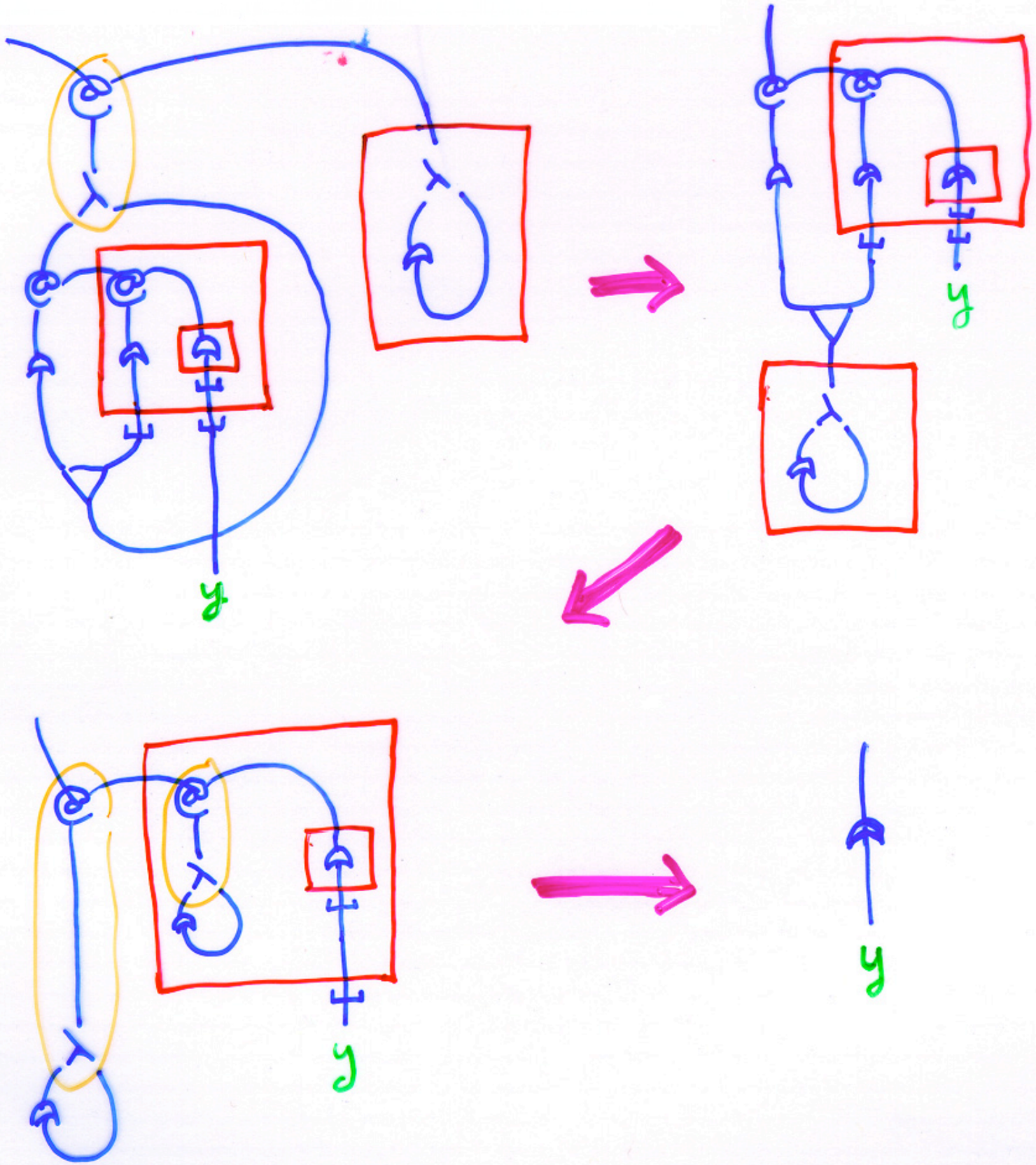
Examples



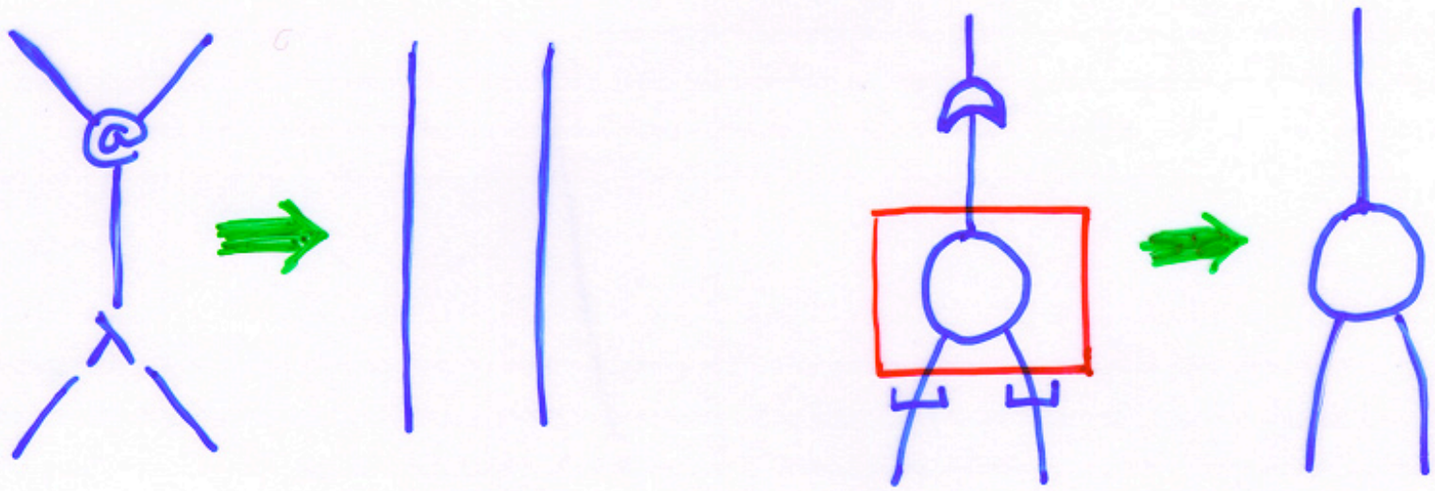


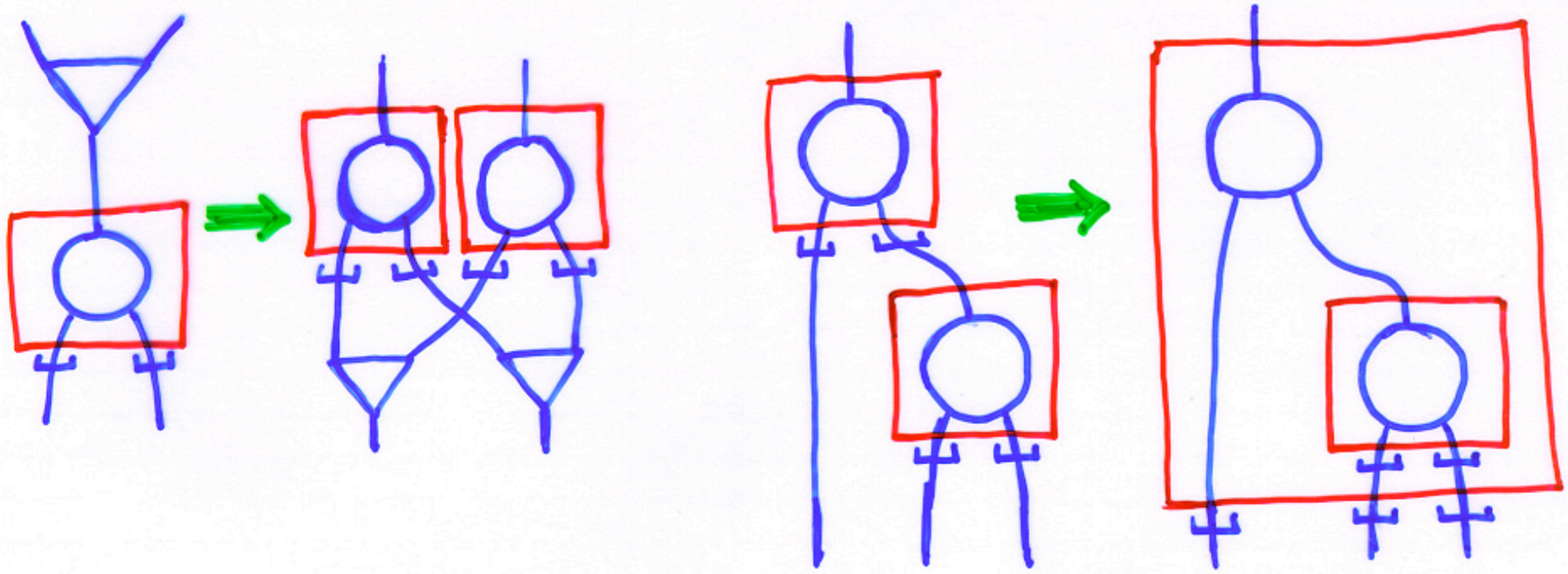


Global reduction: $(\lambda f.f\,(f\,y))\,(\lambda x.x)$



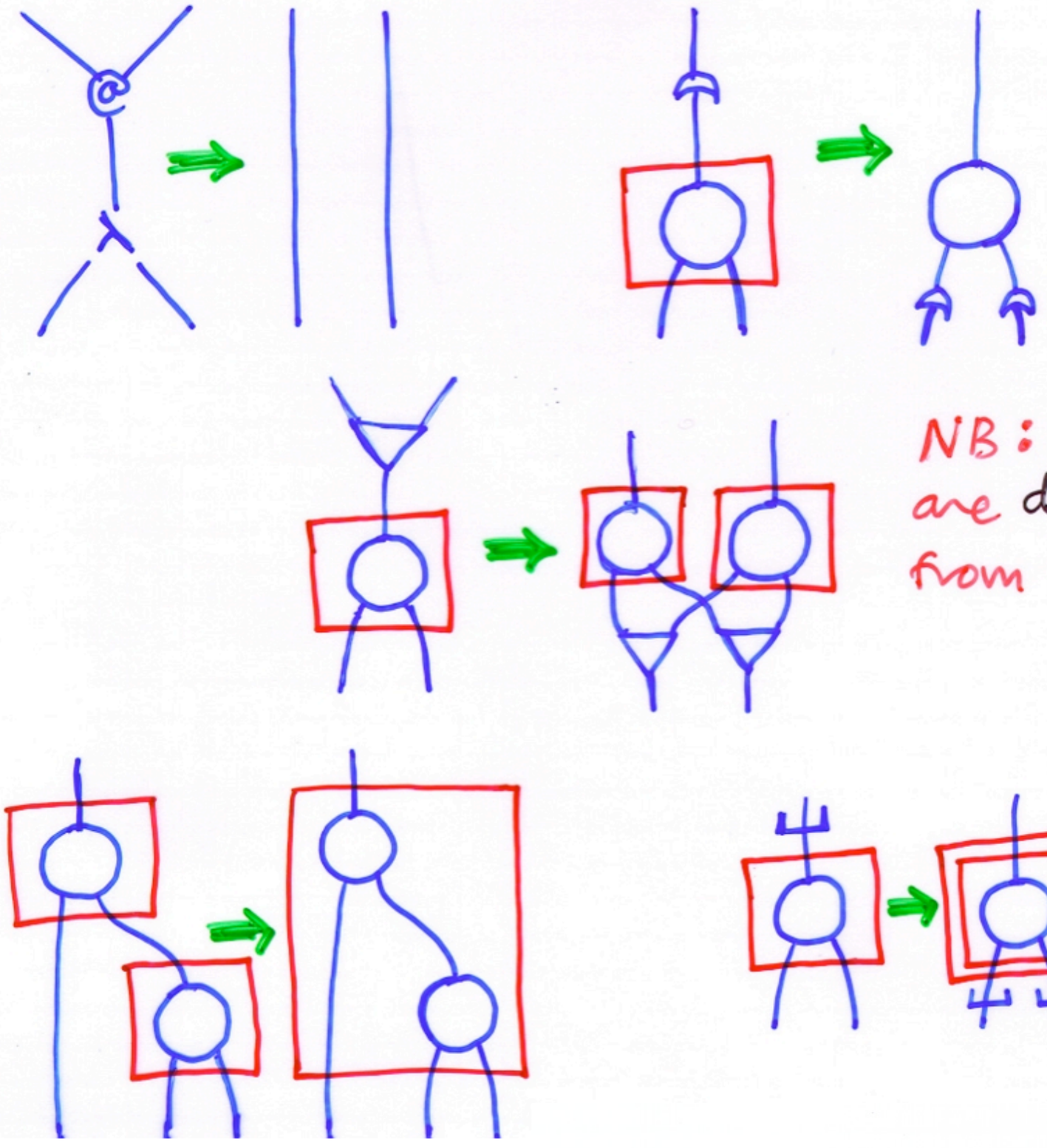
Global reduction rules





Global β-step: Use  , then other rules as much as possible.

Insular reduction rules: a semantics-preserving alternative

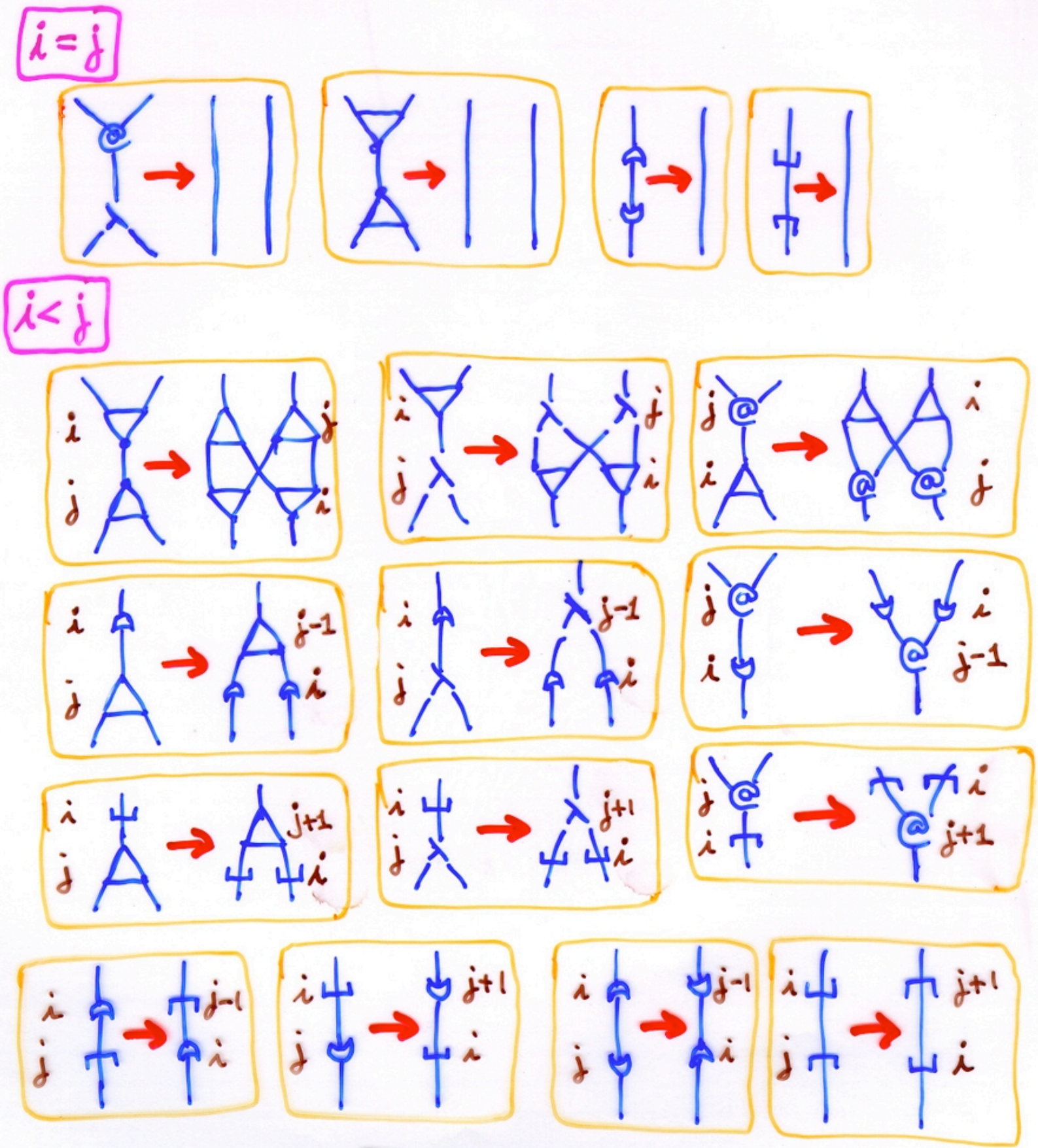


NB: Brachets are detached from boxes

Optimal reduction

- semantics preserving
  (what semantics?)
- maximal sharing
  of β-redexes
- local rules only
  (boxes handled
  incrementally)

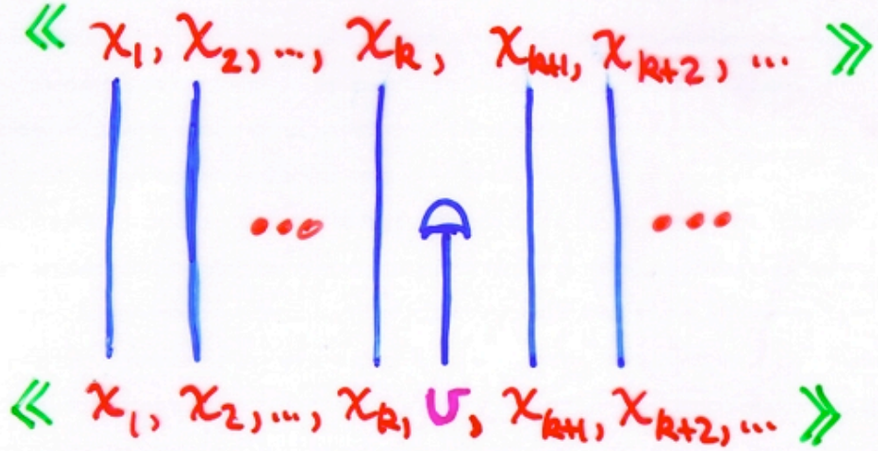
RECALL... that the index of a node is the number of boxes drawn around it

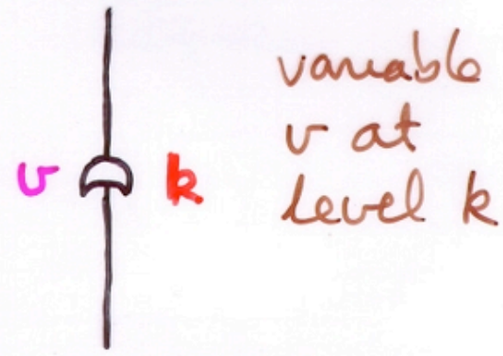


Context semantics with sharing: the bus

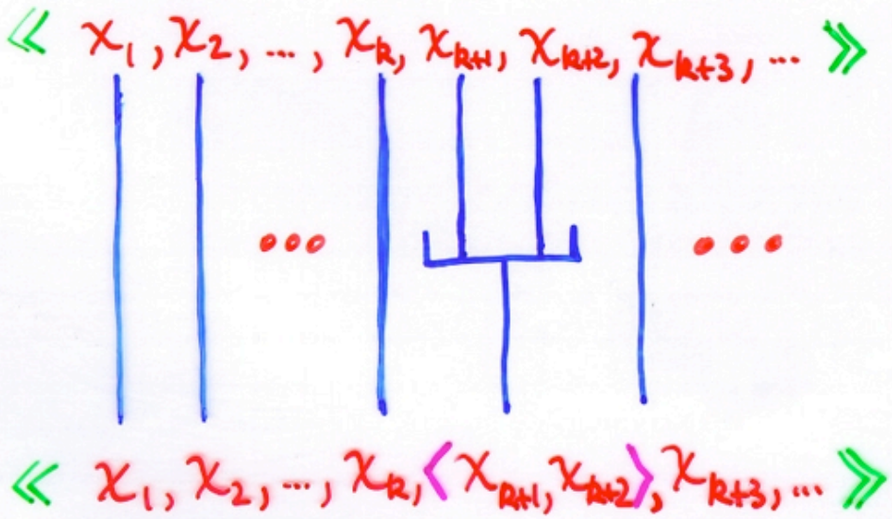
- level of a graph node = # enclosing boxes
- expand each graph edge to a bus of wires:
  context transformations by nodes become
  level dependent.

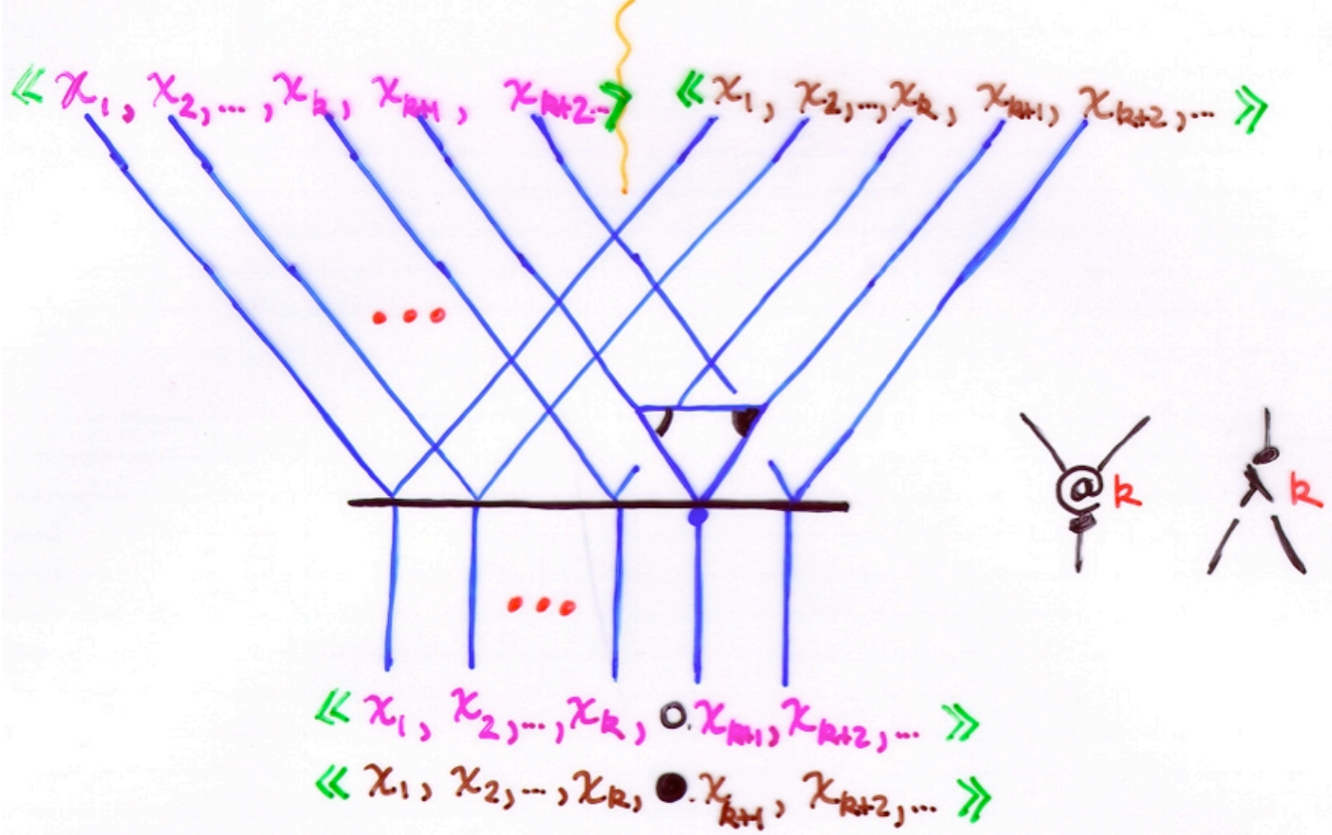
in turn...





variable v at level k



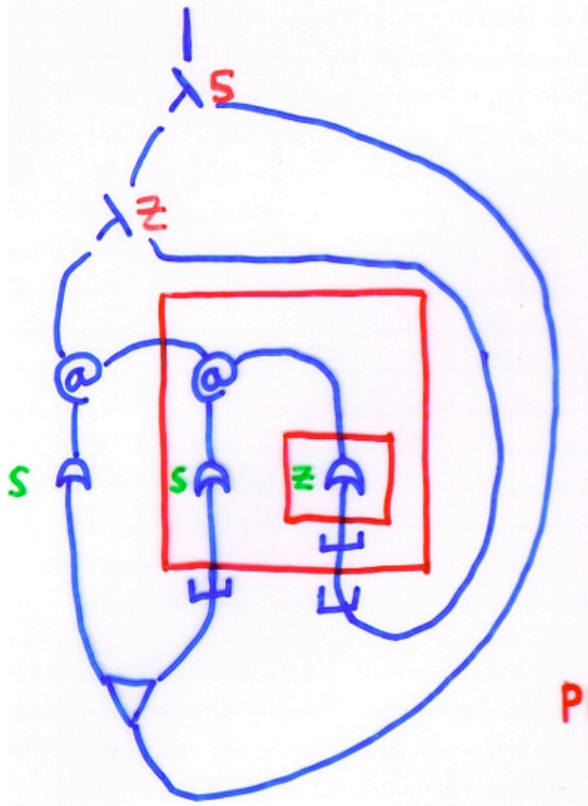


Sharing node  : change O, ● to L, R.

Why All This? Refined readback (and flow analysis)

"head variable x, λ-bound at subterm S, the m^{th} argument of head variable y, λ-bound at subterm T, the n^{th} argument of head variable z, λ-bound at ..."

... Which occurrence of x, y, z, ... ?



Readback Revisited

Game: Opponent tries to guess λ-term that is known by the Player

Op_1: o^+ (initial move)
$= \langle\!\langle o^+, -, -, \ldots \rangle\!\rangle$
"What's your head variable?"

Pl_1: $\bullet Ls, o^+$
"The left occurrence of s."

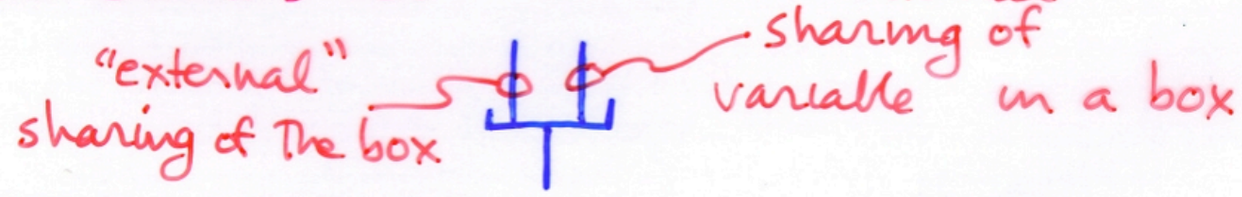
Op_2: $\bullet Ls, \bullet\alpha, o^+$
"What's the head variable of its first argument?"

Pl_2: $\bullet R\langle\alpha, s\rangle, o^+$
"The right occurrence of s, in the box bound to α."

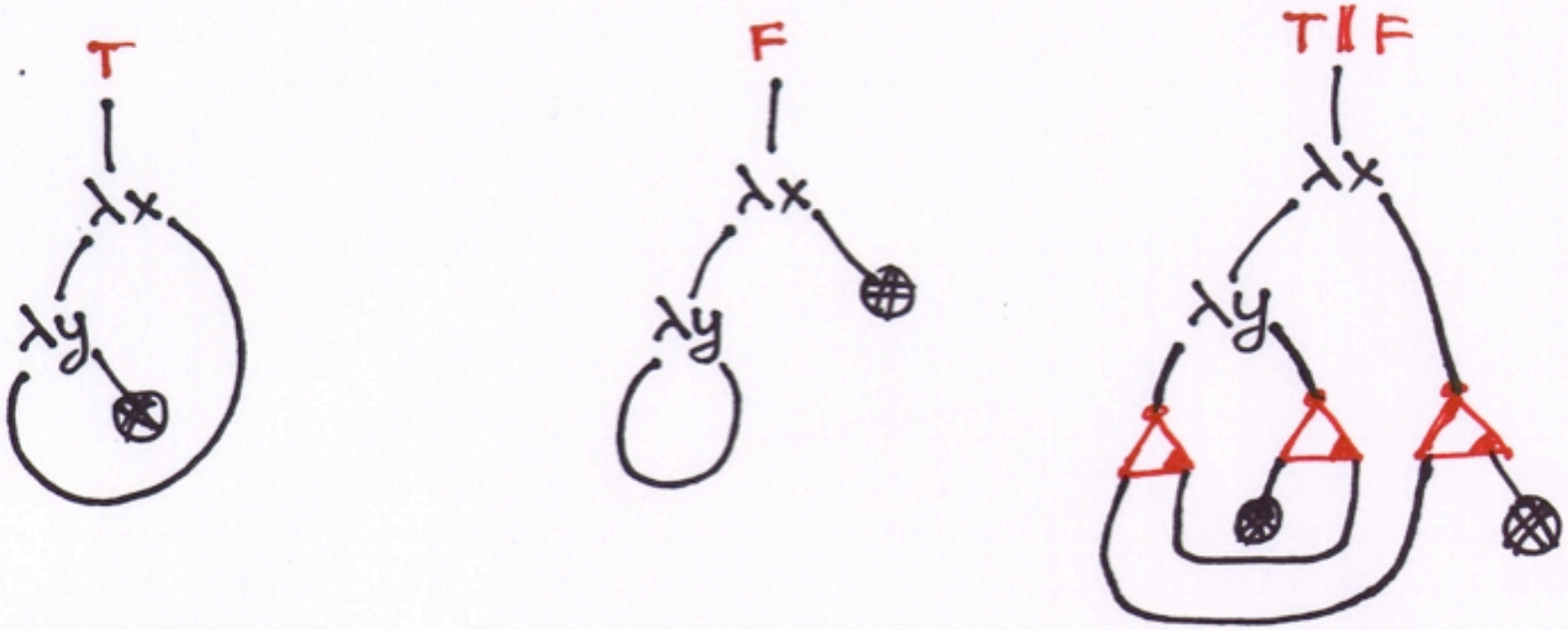
Op_3: $\bullet R\langle\alpha, s\rangle, \bullet\beta, o^+$ "And the head variable of its first argument?"

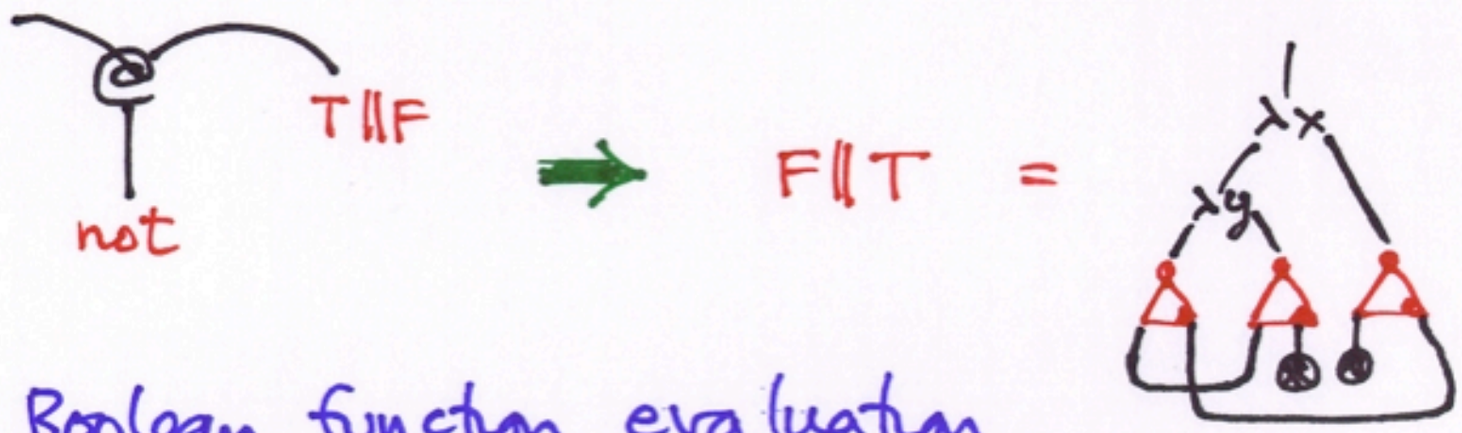
Pl_3: $oo\langle\alpha, \langle\beta, z\rangle\rangle$ "z, in the box bound to β, in the box bound to α."

What Brackets Do:



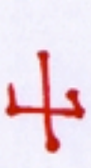
The context semantics is REVERSIBLE,
and we also have SUPERPOSITION...



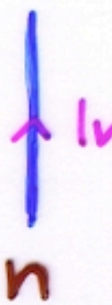


- Boolean function evaluation can be superposed (P = NP ?)
- Extensions to "higher level sharing"
- Readback: can we read a superposed state?

Why Lamping's optimal evaluation algorithm is correct...

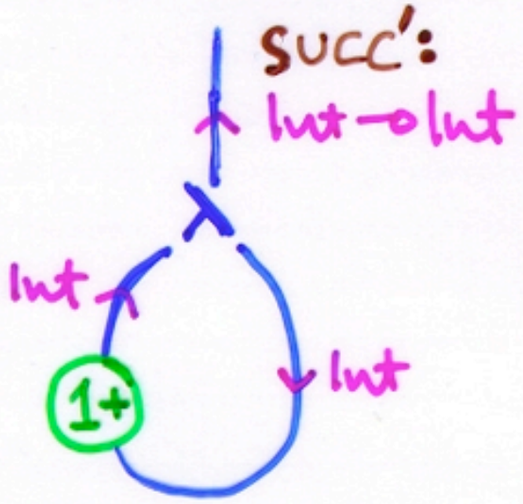
- Readback is correct on graph codings of normal forms — and remains so if the control nodes () are put in funny places.
- Insular and global reduction produce the same normal forms (up to funny placement of ).
- Insular reduction doesn't change the context semantics used by readback.
- Local reduction (Lamping's algorithm) doesn't change the context semantics at all.

PCF: Adding base types, operations, &c, where linear logic describes sharing

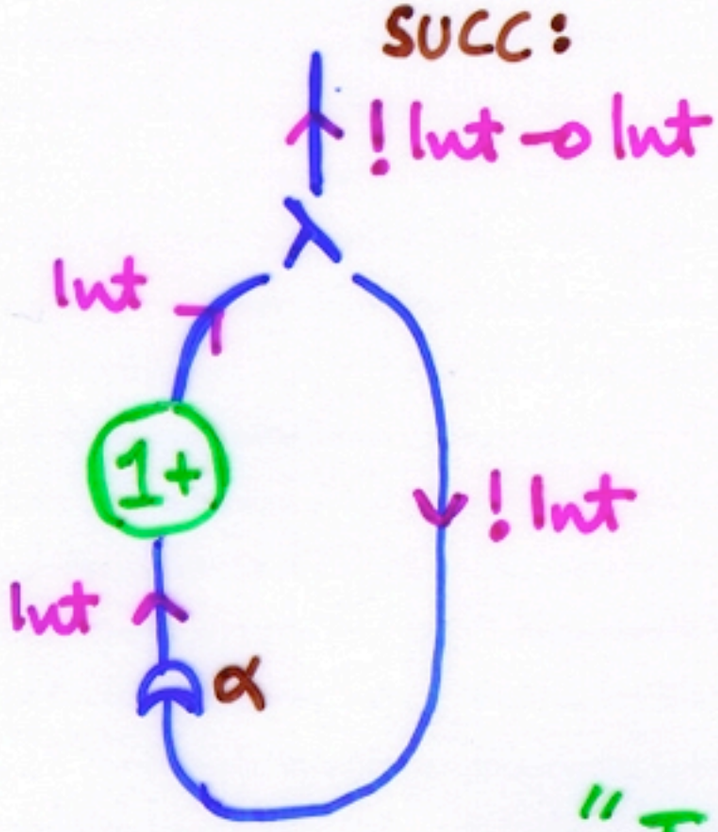


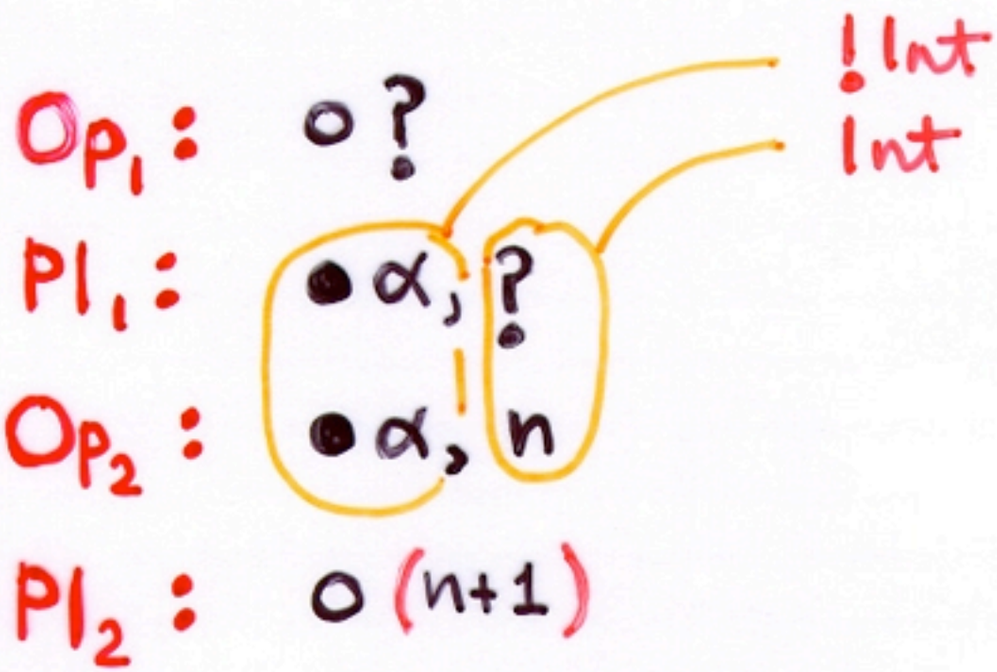
Op: ?

Pl: n



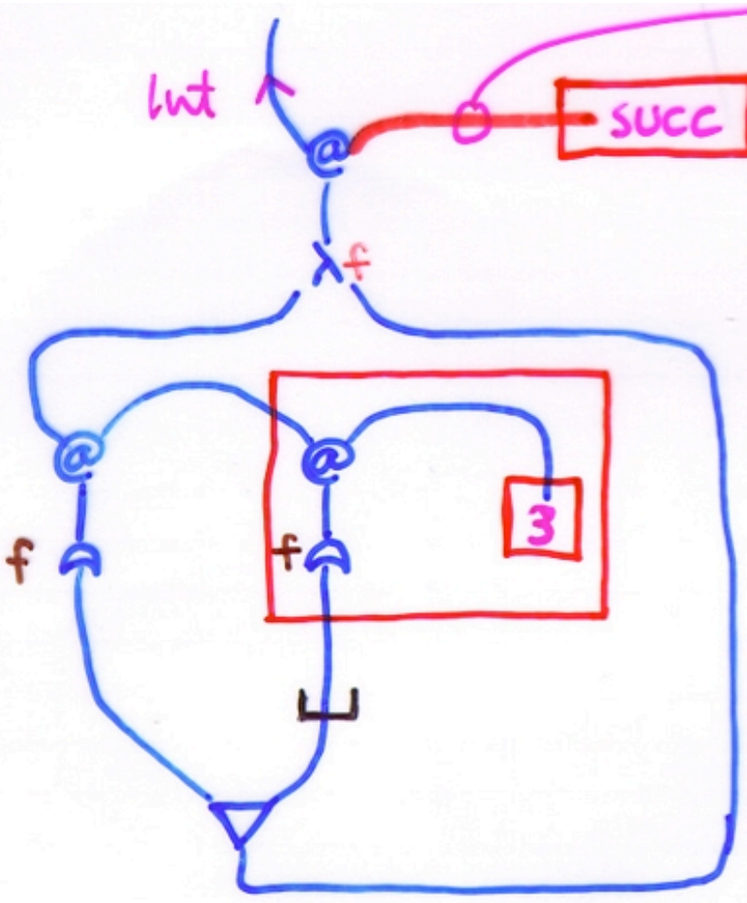
Op_1:	○?	Output?
Pl_1:	●?	Input?
Op_2:	● n	The input is n
Pl_2:	○ (n+1)	The output is n+1





"Types are arenas for games
('Let's play Int⊸Int!'),
terms are winning Player strategies"

Geometry of Interaction: λf.f(f 3) vs. succ



Information flow across this wire

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

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

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

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

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

S_3: R⟨α, f⟩, o4 "The output is 4."

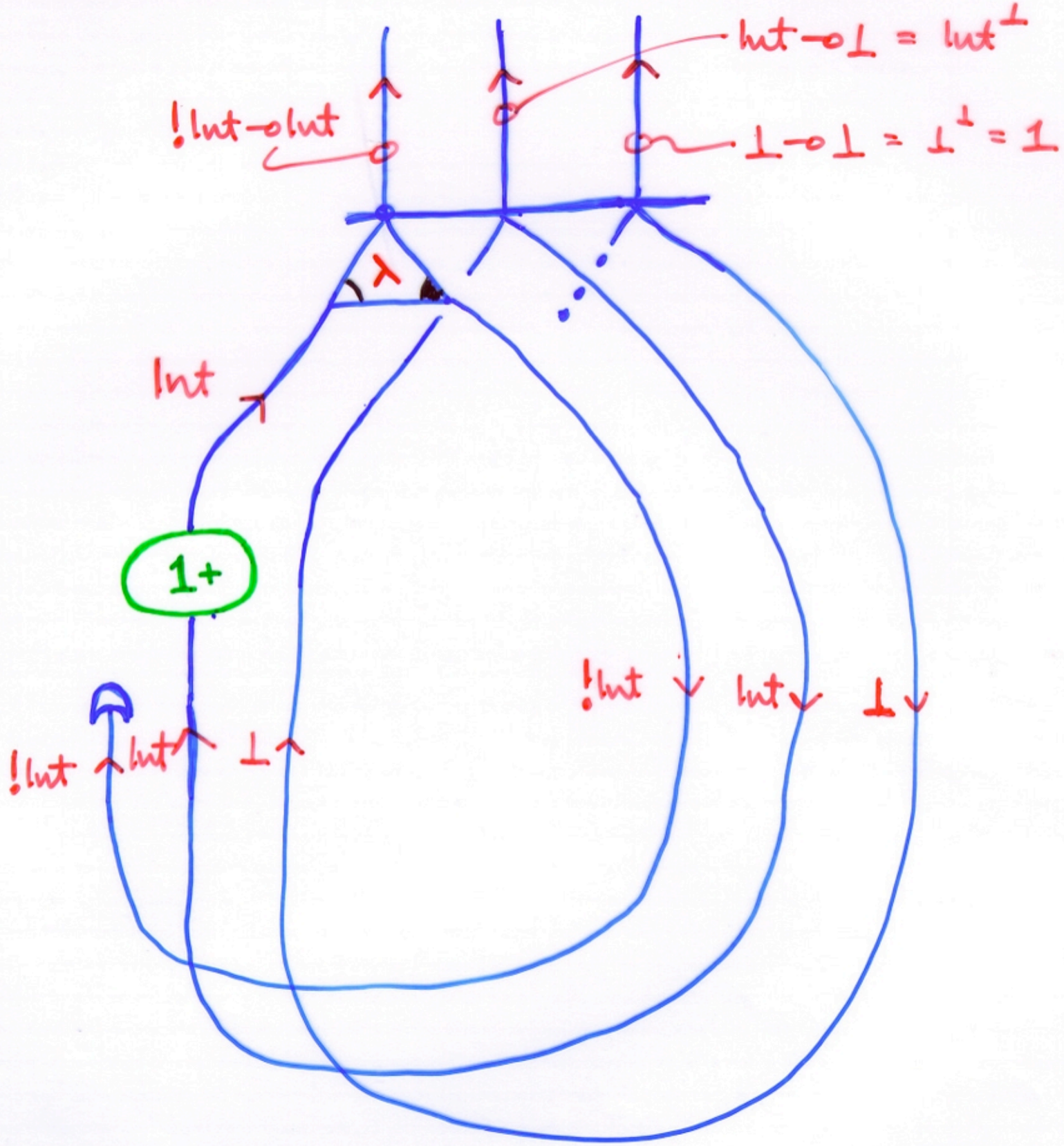
C_4: Lf, •α, 4 "The input is 4." (for the first call)

S_4: Lf, o5 "The output is 5."

"Thou wall, o wall, o sweet and lovely wall:
Show me thy chink to blink through with mine eyne."

Each wire has a dedicated function:

SUCC: !Int ⊸ Int viewed from the bus



Labelled λ-calculus and paths

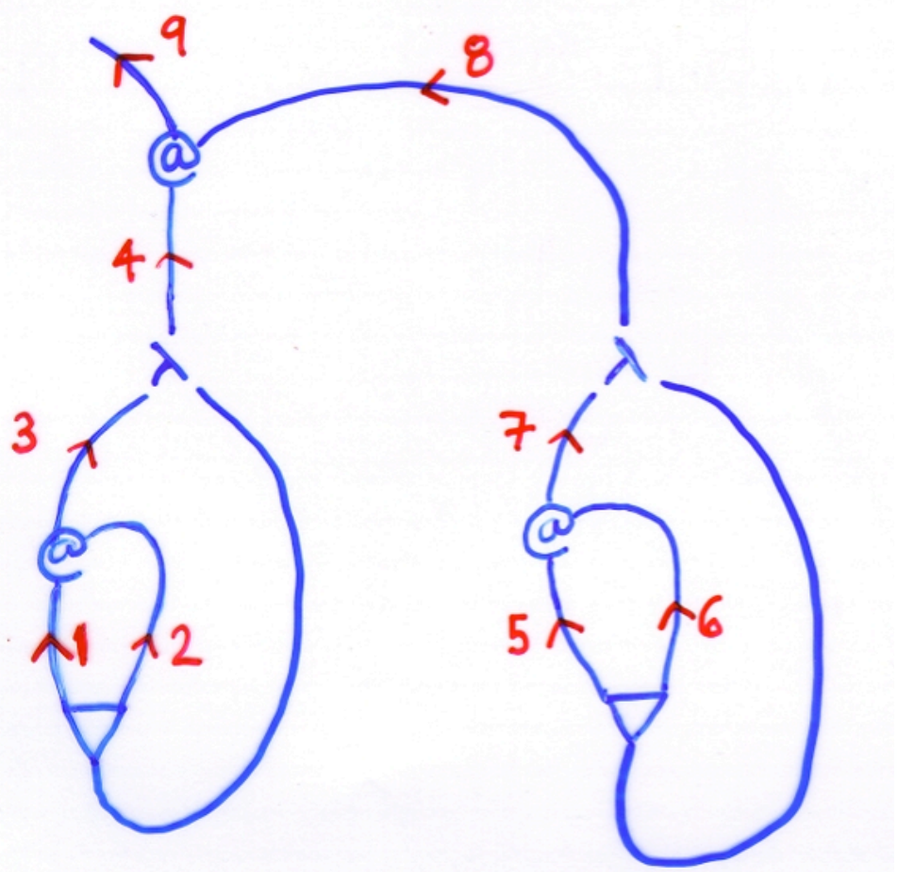
$$(\lambda x.E)^{\ell} F \blacktriangleright ([F^{\ell}/x]E)^{\ell} \qquad (\beta)$$

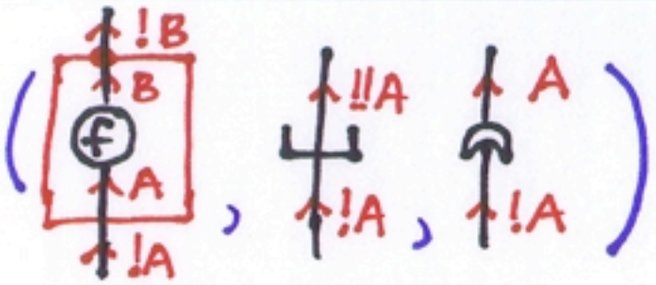
$$((\lambda x.(x^1 x^2)^3)^4(\lambda x.(x^5 x^6)^7)^8)^9$$

$$\blacktriangleright ((\lambda x.(x^5 x^6)^7)^{841}(\lambda x.(x^5 x^6)^7)^{842})^{349}$$

$$\blacktriangleright ((\lambda x.(x^5 x^6)^7)^{8421\,8415}(\lambda x.(x^5 x^6)^7)^{8421\,8416})^{7841349}$$

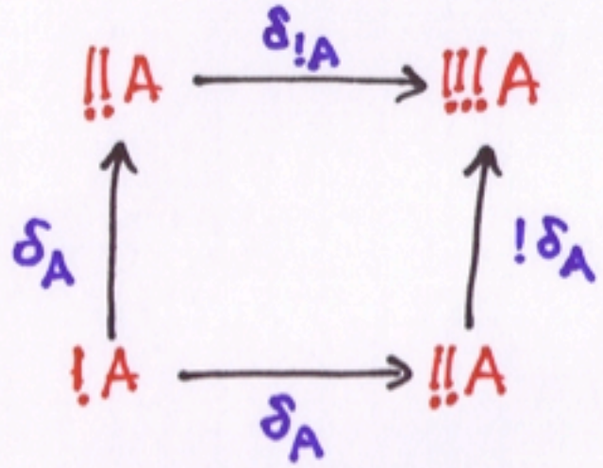
Read $\underline{\ell} = \ell^R$: $\underline{\underline{\ell}} = \ell$, $\underline{\ell\ell'} = \underline{\ell'}\,\underline{\ell}$, $8421\,8415 = 8421485 \approx 215$

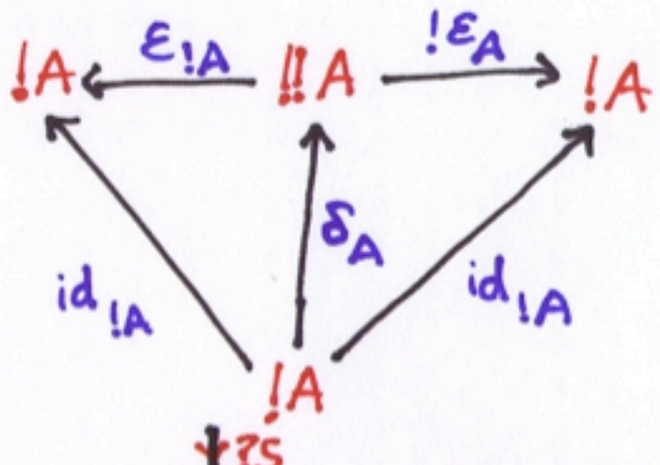


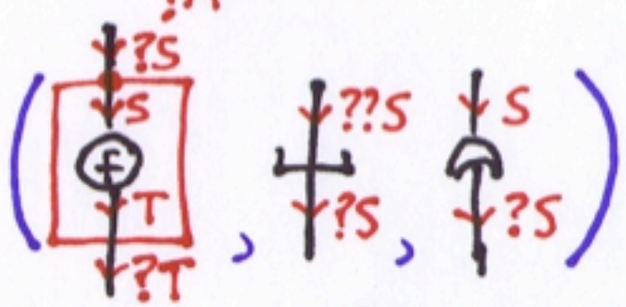
Sharing! The comonad $(!, \delta, \varepsilon) =$ 

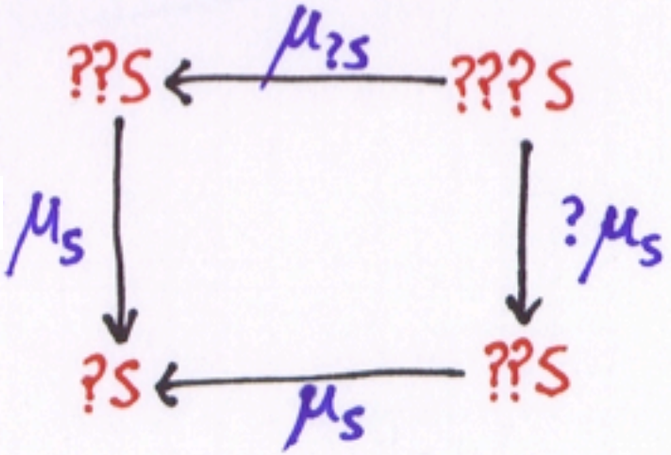
$! : \mathcal{G} \to \mathcal{G}$ a covariant functor

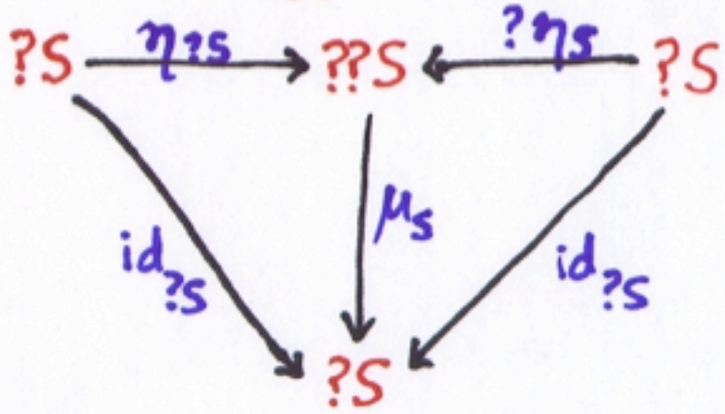
$\delta : ! \to !!$
$\varepsilon : ! \to 1$
} natural transformations (polymorphic functions)





Unsharing? The monad $(?, \mu, \eta) =$ 





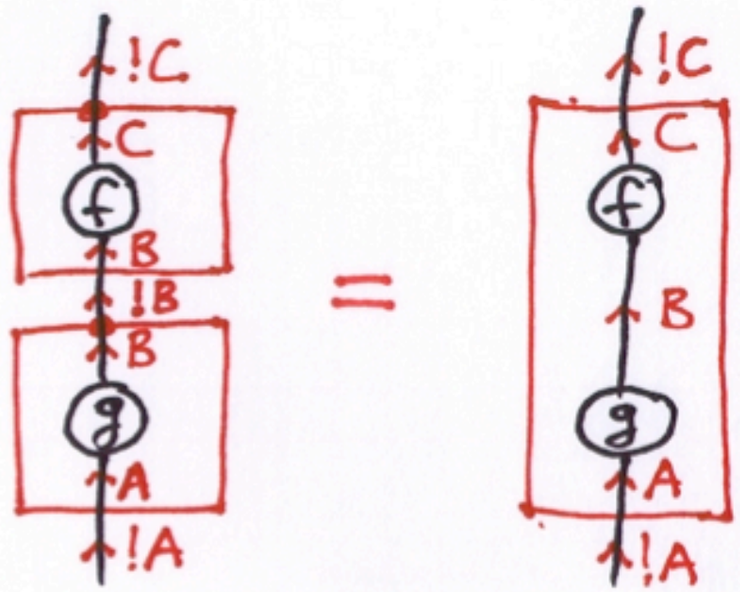
... but what's this got to do with GRAPHS? ...

What's all this got to do with GRAPHS ?

! (box) is a functor

$!(f \circ g) = !f \circ !g$

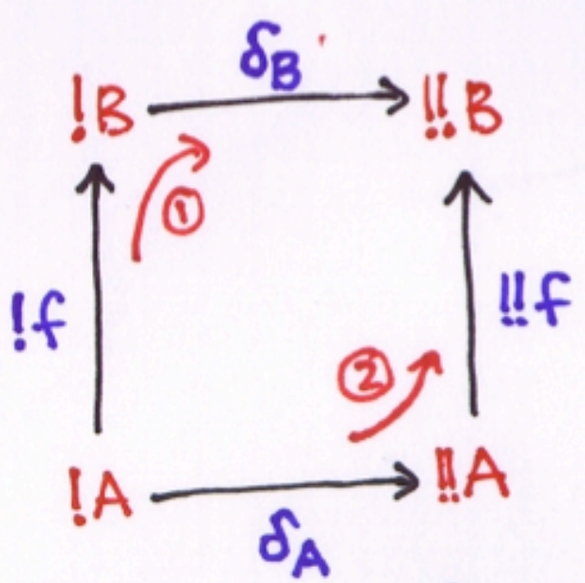
(box fusing)

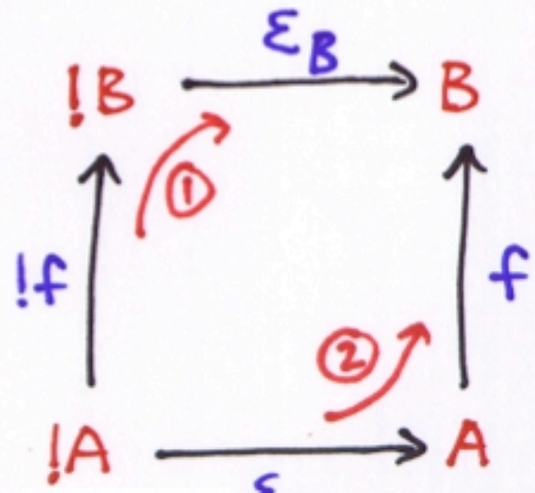


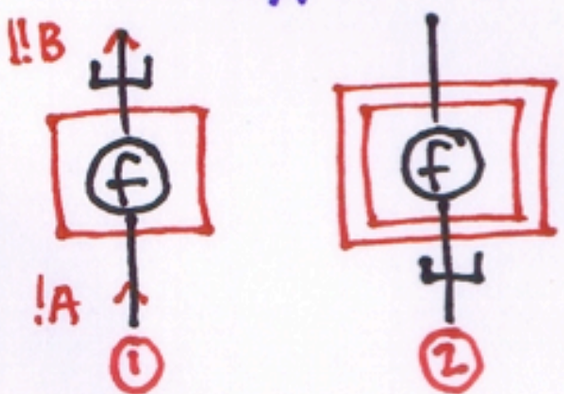
$\delta =$ 

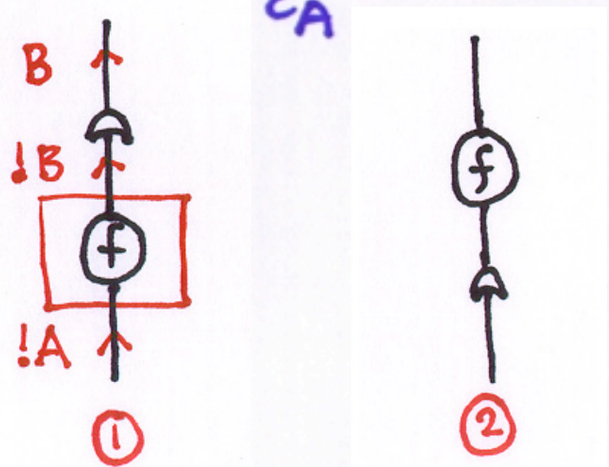
$\varepsilon =$ 

are natural transformations



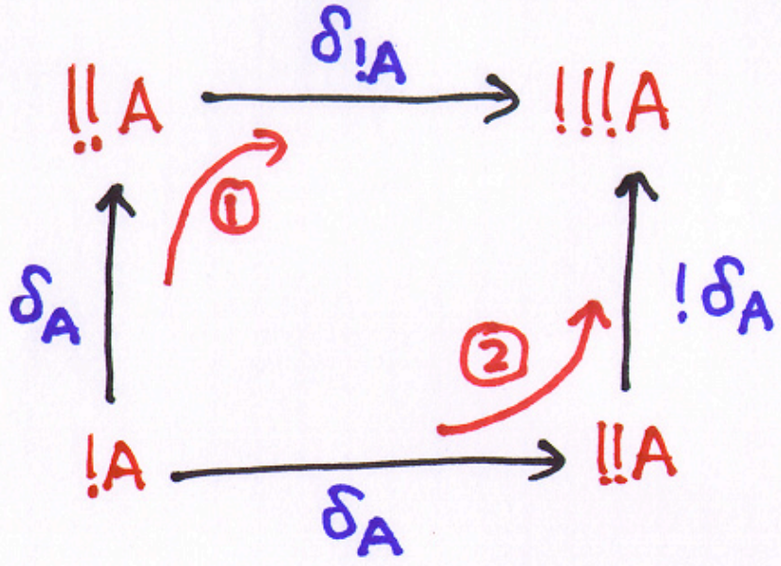


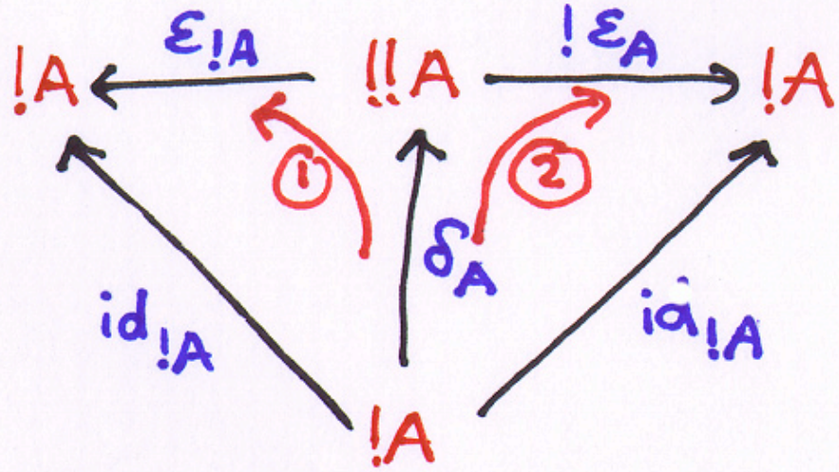




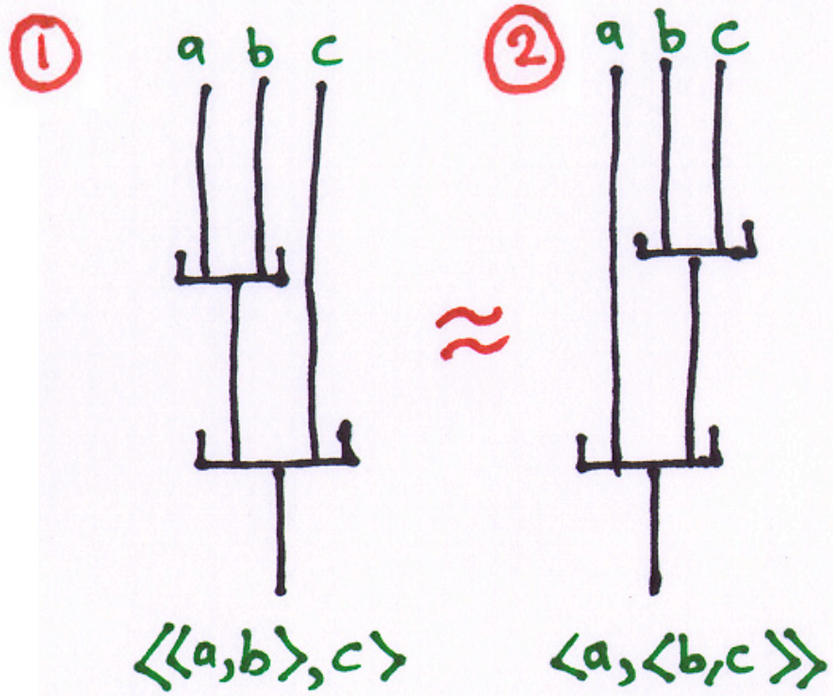
What's all this got to do with GRAPHS?

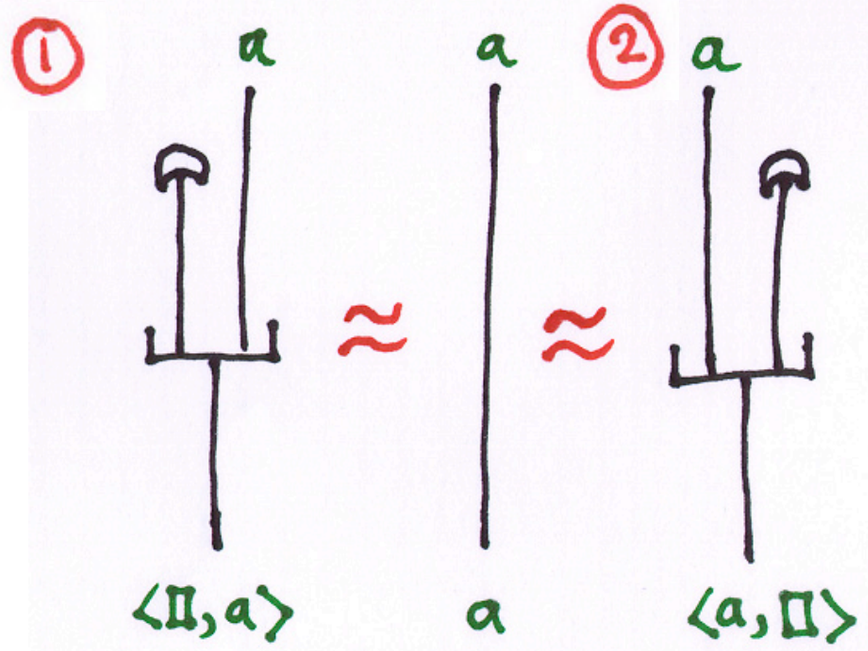
... comonad equations ...





... look at the bus (context semantics) ...



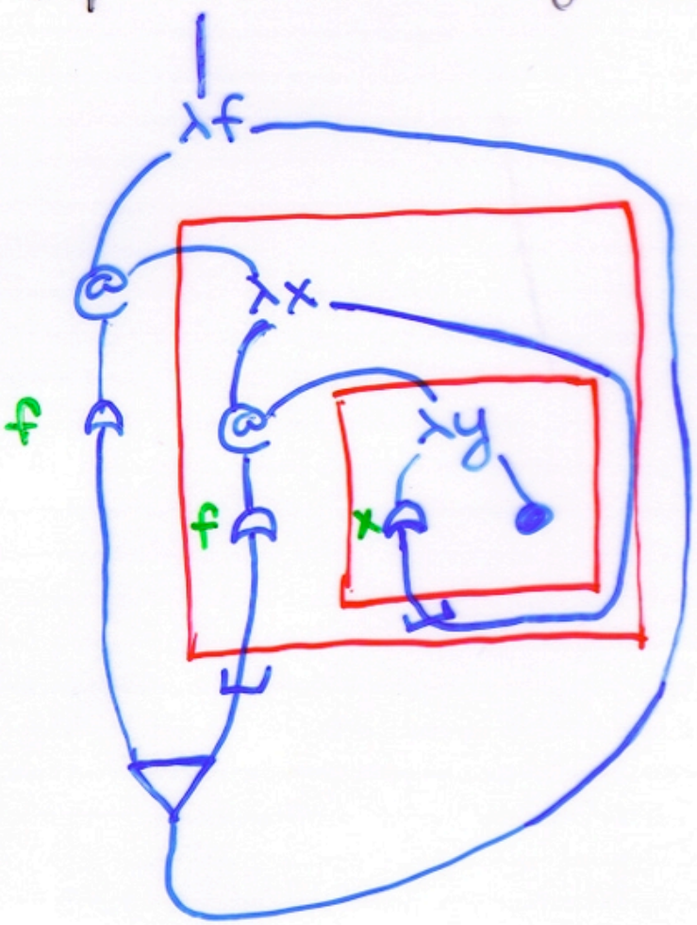


Conclusions, open problems

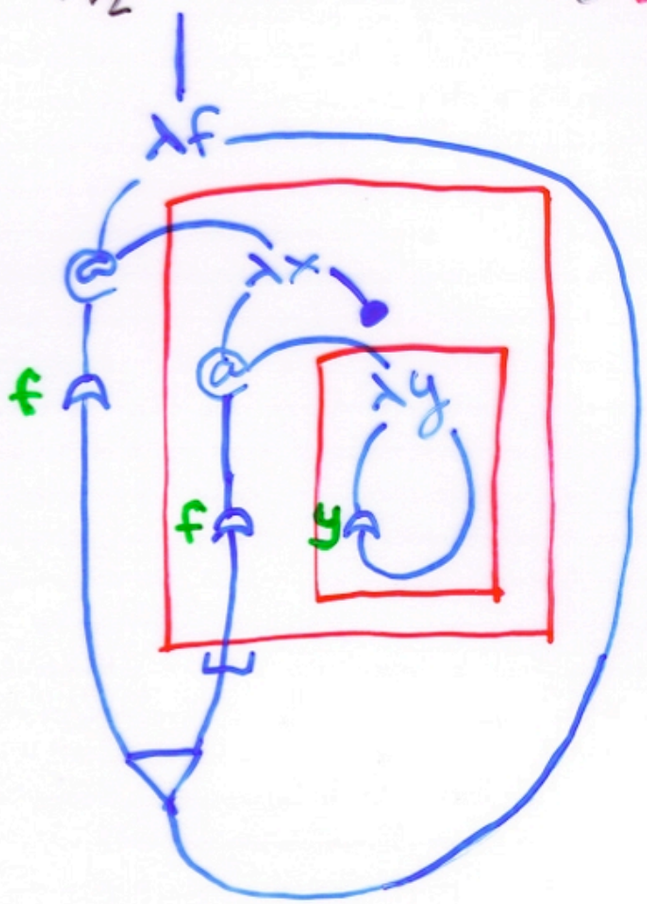
- Context semantics: a complete static analysis (as are labelling schemes) — or what can't it do?
- Where do known flow analyses live within (conservative) or beyond (liberal).
  "false −" "false +"
  Slop factor: differentiating variable occurrences (e.g. 0CFA).
- Computing intersection types from semantics.
- Full abstraction theorems for sharing, e.g.,
  $(\lambda x.xx)(yz) \neq yz(yz)$.
- Give a context semantics of state, I/O, exceptions, &c. realized by graph reduction and reinterpretation of $(?, \Phi, \Psi)$.
- Closer relations to games (in a technical sense).

Kierstad terms: context semantics

$M_1 = \lambda f.\, f\,(\lambda x.\, f\,(\lambda y.\, x))$



$M_2 = \lambda f.\, f\,(\lambda x.\, f\,(\lambda y.\, y))$



$Op_1 = \circ$

$Pl_1 = \bullet Lf$

$Op_2 = \bullet Lf, \bullet\alpha, \circ$

$Pl_2 = \bullet R\langle\alpha, f\rangle$

$Op_3 = \bullet R\langle\alpha, f\rangle, \bullet\beta, \circ$

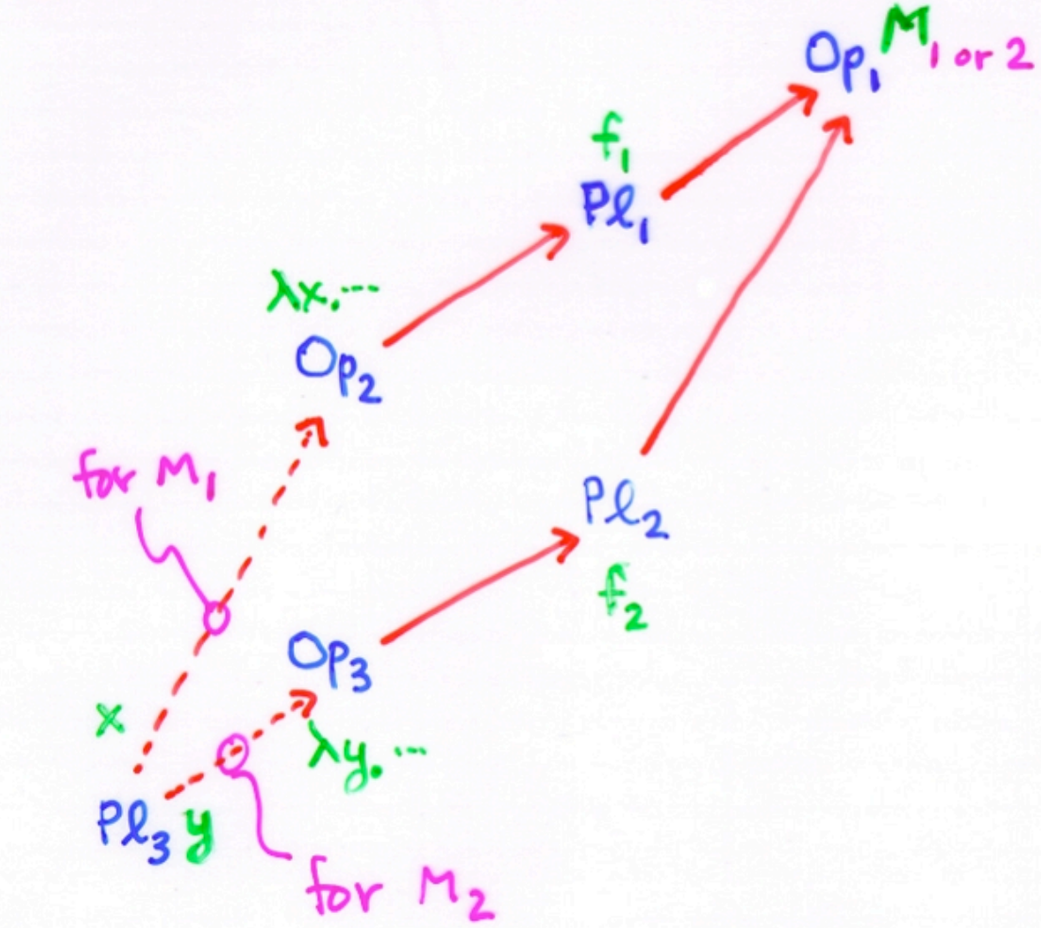
$(M_1)\ Pl_3 = \bullet Lf, \bullet\alpha, \bullet\langle\beta, \underline{x}\rangle$

$(M_2)\ Pl_3 = \bullet R\langle\alpha, f\rangle, \bullet\beta, \bullet\,\underline{y}$

Kierstad terms : games and readback

$$M_1 = \lambda f.\ f_1(\lambda x.\ f_2(\lambda y.\ x))$$
$$M_2 = \lambda f.\ f_1(\lambda x.\ f_2(\lambda y.\ y))$$

$((\text{int} \to \text{int}) \to \text{int}) \to \text{int}$



All moves are ?

("what's the integer?")

Player view ≈ lexical scope
(compile time environment)

Move + pointer chain ≈ context

Conclusions; open problems

- Context semantics is everything — a complete static analysis (as are labelling schemes).
  Where do known flow analyses live within?
  What fragments are tractable?
  (Slop factor: variable occurrences?)
- Context semantics computes Böhm trees (via readback).
  Can it also compute intersection types?
- Context semantics (via cousin, game semantics) is a foundation for <u>full abstraction theorems</u> — what about such a theorem <u>for sharing</u>?

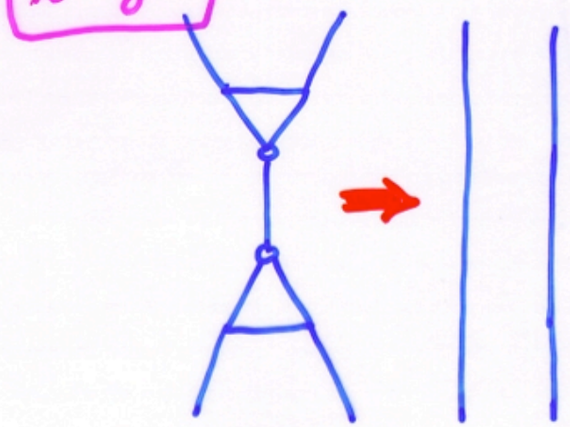
  $(\lambda x.xx)(yz)$ $\neq$ $yz(yz)$

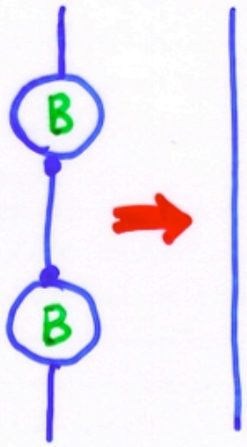
  Applications of y can be shared — not here.
- Context semantics is a semantics of <u>comonads</u>
  $(F, \eta, \mu) = (!, \oplus, \dagger)$ — why not <u>monads</u>, i.e
  $(F, \eta, \mu) = (?, \ominus, \dagger)$ for state, I/O, exceptions, &c,
  realized via graph reduction.
- Closer relation to games (in technical sense)

Optimal reduction (Lamping)

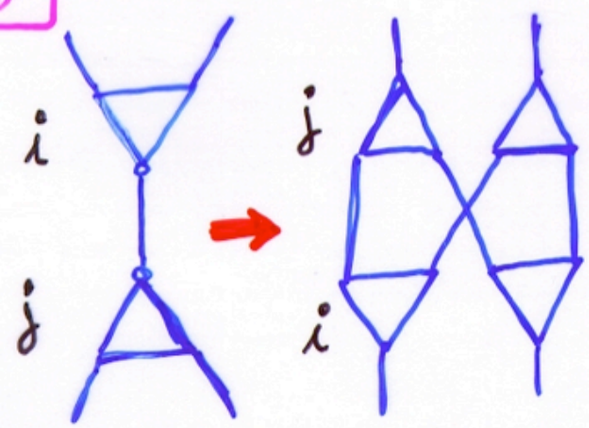
Local, maximal sharing, semantics preserving

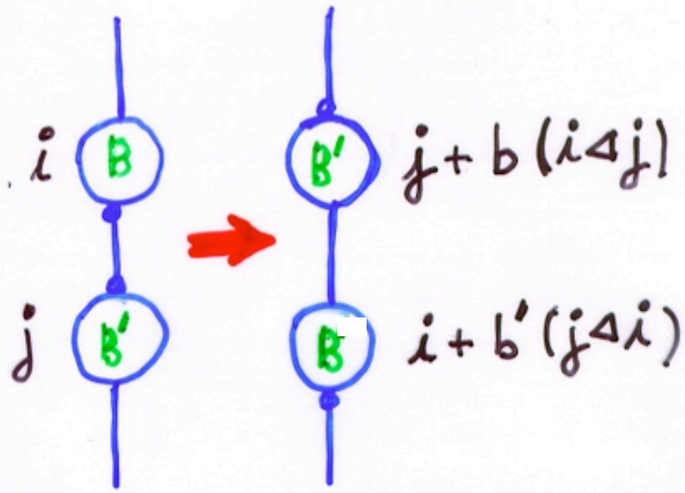
$i = j$:

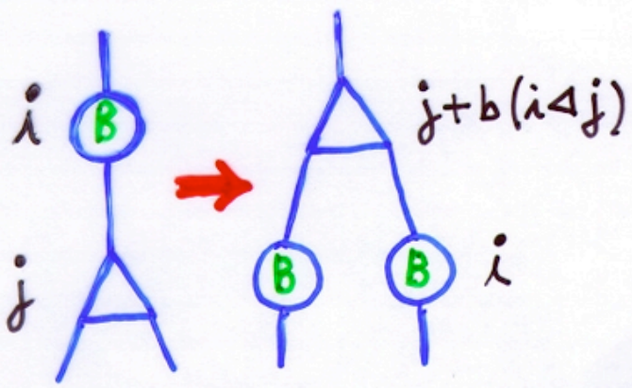




$i \neq j$







$$i \triangleleft j = \begin{cases} 1 & \text{if } i < j \\ 0 & \text{otherwise} \end{cases}$$

$$b = \begin{cases} +1 & \text{if } B = \Psi \\ -1 & \text{if } B = \Phi \end{cases}$$