

Introduction

This is *part* of the material that I've prepared for a very introductory course on λ -calculus, types, intuitionistic propositional logic, Curry-Howard, Categories, Lisp and Lua... the course is 2hs/week, has no prerequisites at all, has no homework, and is usually attended by 2nd/3rd semester CompSci students; they spend most of their time in class discussing and doing exercises together in groups on a whiteboard.

I still need to: a) typeset lots of exercises that I wrote directly on the whiteboard, and some of their solutions — see:

<http://angg.twu.net/2017.2-LA/2017.2-LA.pdf> (whiteboards, PDFized)

<http://angg.twu.net/2017.2-LA.html> (course page)

and b) write a decent introduction, and c) make this self-contained.

Outline of the course

Evaluation

Directed graphs

Basic mathematical objects

Variables (and simultaneous substitution)

Functions as their graphs (i.e., as sets of pairs)

Operations like ' Σ ' (including ' λ ')

(and lots more)

Btw:

<http://angg.twu.net/LATEX/idarct-preprint.pdf>

<http://angg.twu.net/LATEX/2017planar-has-1.pdf>

<http://angg.twu.net/math-b.html#idarct>

<http://angg.twu.net/math-b.html#zhas-for-children-2>

<http://angg.twu.net/logic-for-children-2018.html>

Eduardo Ochs, 2017dez20

Expressions (and reductions)

The usual way to calculate an expression, one step at a time, with '='s:

$$\begin{aligned} 2 \cdot 3 + 4 \cdot 5 &= 2 \cdot 3 + 20 \\ &= 6 + 20 \\ &= 26 \end{aligned}$$

$$\begin{aligned} 2 \cdot 3 + 4 \cdot 5 &= 6 + 4 \cdot 5 \\ &= 6 + 20 \\ &= 26 \end{aligned}$$

Each '=' corresponds to a ' \rightarrow ' in the reduction diagram below.

A notation for calculating the value of an expression by calculating the values of all its subexpressions:

$$\underbrace{\underbrace{2 \cdot 3}_6 + \underbrace{4 \cdot 5}_{20}}_{26}$$

Each '=' in the previous diagram corresponds to applying one ' $\underbrace{\quad}$ '.

A *reduction diagram* for $2 \cdot 3 + 4 \cdot 5$:
(See Hindley/Seldin, pages 14 and 17)

$$\begin{array}{ccccc} 2 \cdot 3 + 4 \cdot 5 & \longrightarrow & 2 \cdot 3 + 20 & & \\ \downarrow & & \downarrow & & \\ 6 + 4 \cdot 5 & \longrightarrow & 6 + 20 & \longrightarrow & 26 \end{array}$$

Note that when we can choose two subexpressions to calculate the ' \downarrow ' evaluates the leftmost one, and the ' \rightarrow ' evaluates the rightmost one.

The *subexpressions* of $2 \cdot 3 + 4 \cdot 5$:

$$\underbrace{\underbrace{2 \cdot 3}_6 + \underbrace{4 \cdot 5}_{20}}_{26}$$

Exercise:

Do the same as above for these expressions:

a) $2 \cdot (3 + 4) + 5 \cdot 6$

b) $2 + 3 + 4$

c) $2 + 3 + 4 + 5$

(Improvise when needed)

Expressions with variablesIf $a = 5$ and $b = 2$, then:

$$\underbrace{\underbrace{\underbrace{a}_{5} + \underbrace{b}_{2}}_{7} \cdot \underbrace{\underbrace{a}_{5} - \underbrace{b}_{2}}_{3}}_{21}$$

If $a = 10$ and $b = 1$, then:

$$\underbrace{\underbrace{\underbrace{a}_{10} + \underbrace{b}_{1}}_{11} \cdot \underbrace{\underbrace{a}_{10} - \underbrace{b}_{1}}_{9}}_{99}$$

We know – by algebra, which is not for (tiny) children – that $(a + b) \cdot (a - b) = a \cdot a - b \cdot b$ is true for all $a, b \in \mathbb{R}$

We know – without algebra – how to test

“($a + b$) · ($a - b$) = $a \cdot a - b \cdot b$ ”

for specific values of a and b ...

If $a = 5$ and $b = 2$, then:

$$\underbrace{\underbrace{\underbrace{a}_{5} + \underbrace{b}_{2}}_{7} \cdot \underbrace{\underbrace{a}_{5} - \underbrace{b}_{2}}_{3}}_{21} = \underbrace{\underbrace{a}_{5} \cdot \underbrace{a}_{5}}_{25} - \underbrace{\underbrace{b}_{2} \cdot \underbrace{b}_{2}}_{4}}_{21}$$

true

If $a = 10$ and $b = 1$, then:

$$\underbrace{\underbrace{\underbrace{a}_{10} + \underbrace{b}_{1}}_{11} \cdot \underbrace{\underbrace{a}_{10} - \underbrace{b}_{1}}_{9}}_{99} = \underbrace{\underbrace{a}_{10} \cdot \underbrace{a}_{10}}_{100} - \underbrace{\underbrace{b}_{1} \cdot \underbrace{b}_{1}}_{1}}_{99}$$

true

A notation for (simultaneous) substitution:

$$((x + y) \cdot z) \left[\begin{array}{l} x:=a+y \\ y:=b+z \\ z:=c+x \end{array} \right] = ((a + y) + (b + z)) \cdot (c + x).$$

Note that $((a + b) \cdot (a - b)) \left[\begin{array}{l} a:=5 \\ b:=2 \end{array} \right] = (5 + 2) \cdot (5 - 2)$.

Operations with substitution and copying

We know that $\sum_{i=2}^5 i^3 = 2^3 + 3^3 + 4^3 + 5^3$.

If we introduce some intermediate steps we get:

$$\begin{aligned} & \sum_{i=2}^5 i^3 \\ \rightsquigarrow & \sum_{i \text{ in } (2,3,4,5)} i^3 \\ \rightsquigarrow & (i^3)[i := 2] + (i^3)[i := 3] + (i^3)[i := 4] + (i^3)[i := 5] \\ \rightsquigarrow & 2^3 + 3^3 + 4^3 + 5^3 \end{aligned}$$

$$\begin{aligned} & \forall a \in \{2, 3, 5\}. a < 4 \\ \rightsquigarrow & \forall a \text{ in } (2, 3, 5). a < 4 \\ \rightsquigarrow & (a < 4)[a := 2] \& (a < 4)[a := 3] \& (a < 4)[a := 5] \\ \rightsquigarrow & (2 < 4) \& (3 < 4) \& (5 < 4) \\ \rightsquigarrow & \text{false} \end{aligned}$$

$$\begin{aligned} & \forall a \in \{2, 3, 3, 5\}. a < 4 \\ \rightsquigarrow & \forall a \text{ in } (2, 3, 3, 5). a < 4 \\ \rightsquigarrow & (a < 4)[a := 2] \& (a < 4)[a := 3] \& (a < 4)[a := 3] \& (a < 4)[a := 5] \\ \rightsquigarrow & (2 < 4) \& (3 < 4) \& (3 < 4) \& (5 < 4) \\ \rightsquigarrow & \text{false} \end{aligned}$$

$$\begin{aligned} & \{a^3 \mid a \in \{2, 3, 5\}\} \\ \rightsquigarrow & \{(a^3)[a := 2], (a^3)[a := 3], (a^3)[a := 5]\} \\ \rightsquigarrow & \{2^3, 3^3, 5^3\} \end{aligned}$$

$$\begin{aligned} & \{(a, a^3) \mid a \in \{2, 3, 5\}\} \\ \rightsquigarrow & \{(a, a^3)[a := 2], (a, a^3)[a := 3], (a, a^3)[a := 5]\} \\ \rightsquigarrow & \{(2, 2^3), (3, 3^3), (5, 5^3)\} \\ \rightsquigarrow & \{(2, 8), (3, 27), (5, 125)\} \end{aligned}$$

One way to understand the ‘ λ ’ operator is using the idea — from Calculus 1 and Discrete Mathematics — that a function is a set of pairs (its “graph”)...

$$\begin{aligned} & \lambda a:\{2, 3, 5\}. a^3 \\ \rightsquigarrow & \{(a, a^3) \mid a \in \{2, 3, 5\}\} \\ \rightsquigarrow & \{(a, a^3)[a := 2], (a, a^3)[a := 3], (a, a^3)[a := 5]\} \\ \rightsquigarrow & \{(2, 2^3), (3, 3^3), (5, 5^3)\} \\ \rightsquigarrow & \{(2, 8), (3, 27), (5, 125)\} \end{aligned}$$

Note that

$$\begin{aligned} & (\lambda a:\{2, 3, 5\}. a^3)(5) \\ \rightsquigarrow & (\{(2, 2^3), (3, 3^3), (5, 5^3)\})(5) \\ \rightsquigarrow & 5^3 \\ \rightsquigarrow & 125 \end{aligned}$$

$$\begin{aligned} & (\lambda a:\{2, 3, 5\}. a^3)(4) \\ \rightsquigarrow & (\{(2, 2^3), (3, 3^3), (5, 5^3)\})(4) \\ \rightsquigarrow & \text{error} \end{aligned}$$

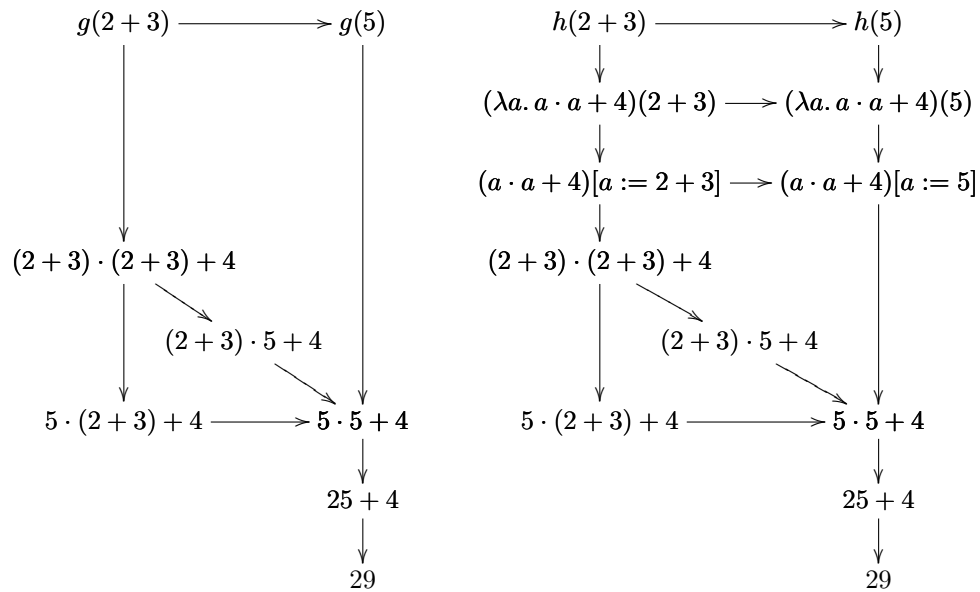
Lambda

A named function: $g(a) = a \cdot a + 4$

An unnamed function: $\lambda a. a \cdot a + 4$

Let $h = \lambda a. a \cdot a + 4$.

Then:



The usual notation for defining functions is like this:

$$f : \mathbb{N} \rightarrow \mathbb{R}$$

$$n \mapsto 2 + \sqrt{n}$$

$$\begin{array}{l} \text{(name)} : \text{(domain)} \rightarrow \text{(codomain)} \\ \text{(variable)} \mapsto \text{(expression)} \end{array}$$

It creates *named* functions
(with domains and codomains).

The usual notation for creating named functions
without specifying their domains and codomains
is just $f(n) = 2 + \sqrt{n}$.

Note that this is:

$$f \quad (n) \quad = \quad 2 + \sqrt{n}$$

$$\text{(name)} \quad \text{((variable))} \quad = \quad \text{(expression)}$$

Lambda notation: exercises

- a) $(\lambda a.10a)(2+3)$
 b) $(\lambda a.10a)((\lambda b.b+4)(3))$

Hint: use the speed you're most comfortable with. For example:

$$\begin{array}{ccc}
 (\lambda a.10a)((\lambda b.b+4)(3)) & (\lambda a.10a)((\lambda b.b+4)(3)) & (\lambda a.10a)((\lambda b.b+4)(3)) \\
 \underbrace{\underbrace{\underbrace{(b+4)[b:=3]}_{3+4}}_7}_{(10a)[a:=7]} & \underbrace{\underbrace{\underbrace{3+4}_7}_{10 \cdot 7}}_{70} & \underbrace{\underbrace{\underbrace{7}_7}_{70}}_{70} \\
 \underbrace{\underbrace{10 \cdot 7}_{70}}_{70} & &
 \end{array}$$

- c) $((\lambda a.(\lambda b.10a+b))(3))(4)$
 d) $((\lambda f.(\lambda a.f(f(a))))(\lambda x.10x))(7)$

Hint 2: give names to subexpressions.

$$\begin{array}{l}
 \underbrace{(\lambda a.10a)}_{\alpha} \underbrace{((\lambda b.b+4)(3))}_{\underbrace{\beta}_{\gamma}} \quad \alpha(\gamma) = \alpha(\beta(3)) \\
 = \alpha((\lambda b.b+4)(3)) \\
 = \alpha((b+4)[b:=3]) \\
 = \alpha(3+4) \\
 = \alpha(7) \\
 = (\lambda a.10a)(7) \\
 = 70
 \end{array}$$

Functions as their graphs

The *graph* of

$$h : \begin{array}{ccc} \{-2, -1, 0, 1, 2\} & \rightarrow & \{0, 1, 2, 3, 4\} \\ k & \mapsto & k^2 \end{array}$$

is $\{(-2, 4), (-1, 1), (0, 0), (1, 1), (2, 4)\}$.

We can think that a function *is* its graph,

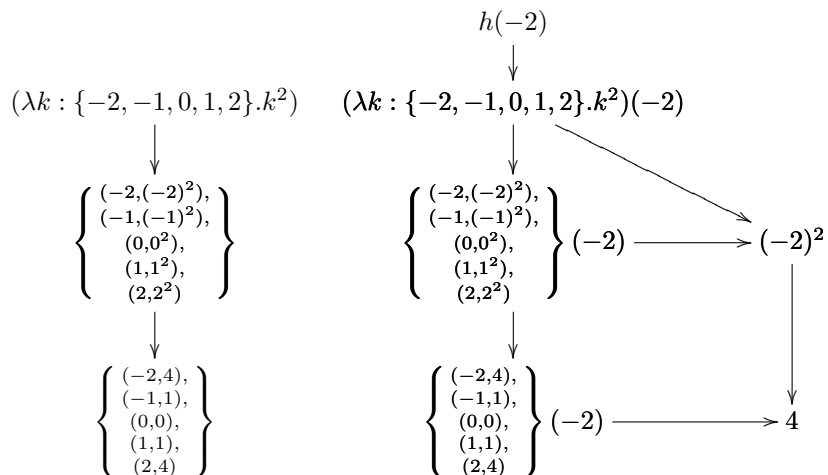
and that a lambda-expression (with domain) reduces to a graph.

Then $h = \{(-2, 4), (-1, 1), (0, 0), (1, 1), (2, 4)\}$

and $h(-2) = \{(-2, 4), (-1, 1), (0, 0), (1, 1), (2, 4)\}(-2) = 4$.

Let $h := (\lambda k : \{-2, -1, 0, 1, 2\}.k^2)$.

We have:



Note:

the graph of $(\lambda n : \mathbb{N}.n^2)$ has infinite points,

the graph of $(\lambda n : \mathbb{N}.n^2)$ is an infinite set,

the graph of $(\lambda n : \mathbb{N}.n^2)$ can't be written down *explicitly* without '...'s...

Mathematicians love infinite sets.

Computers hate infinite sets.

For mathematicians a function *is* its graph

(↑ remember Discrete Mathematics!)

For computer scientists a function *is* a finite program.

Computer scientists love 'λ's!

I love things like this: $\left\{ \begin{array}{l} (3, 30), \\ (4, 40) \end{array} \right\} (3) = 30$

Types (introduction)

Let:

$$A = \{1, 2\}$$

$$B = \{30, 40\}.$$

If $f : A \rightarrow B$, then f is one of these

four functions:

$$\begin{array}{cccc} 1 \mapsto 30 & 1 \mapsto 30 & 1 \mapsto 40 & 1 \mapsto 40 \\ 2 \mapsto 30 & 2 \mapsto 40 & 2 \mapsto 30 & 2 \mapsto 40 \end{array}$$

or, in other notation,

$$\left\{ \begin{array}{l} (1,30) \\ (2,30) \end{array} \right\}, \left\{ \begin{array}{l} (1,30) \\ (2,40) \end{array} \right\}, \left\{ \begin{array}{l} (1,40) \\ (2,30) \end{array} \right\}, \left\{ \begin{array}{l} (1,40) \\ (2,40) \end{array} \right\}$$

which means that:

$$f \in \left\{ \left\{ \begin{array}{l} (1,30) \\ (2,30) \end{array} \right\}, \left\{ \begin{array}{l} (1,30) \\ (2,40) \end{array} \right\}, \left\{ \begin{array}{l} (1,40) \\ (2,30) \end{array} \right\}, \left\{ \begin{array}{l} (1,40) \\ (2,40) \end{array} \right\} \right\}$$

Let's use the notation " $A \rightarrow B$ " for

"the set of all functions from A to B ".

$$\text{Then } (A \rightarrow B) = \left\{ \left\{ \begin{array}{l} (1,30) \\ (2,30) \end{array} \right\}, \left\{ \begin{array}{l} (1,30) \\ (2,40) \end{array} \right\}, \left\{ \begin{array}{l} (1,40) \\ (2,30) \end{array} \right\}, \left\{ \begin{array}{l} (1,40) \\ (2,40) \end{array} \right\} \right\}$$

and $f : A \rightarrow B$

means $f \in (A \rightarrow B)$.

In Type Theory and λ -calculus " $a : A$ "

is pronounced " a is of type A ", and the meaning

of this is *roughly* " $a \in B$ ".

(We'll see the differences between ' \in ' and ':' (much) later).

Note that:

1. if $f : A \rightarrow B$ and $a : A$ then $f(a) : B$

2. if $a : A$ and $b : B$ then $(a, b) : A \times B$

3. if $p : A \times B$ then $\pi p : A$ and $\pi' p : B$, where

' π ' means 'first projection' and

' π' ' means 'second projection';

if $p = (2, 30)$ then $\pi p = 2$, $\pi' p = 30$.

If $p : A \times B$ and $g : B \rightarrow C$, then:

$$\underbrace{\left(\underbrace{\underbrace{(\pi \ p)}_{:A \times B}}_{:A}, \underbrace{g}_{:B \rightarrow C} \left(\underbrace{(\pi' \ p)}_{:A \times B} \right) \right)}_{:C} : A \times C$$

Typed λ -calculus: trees

$$A = \{1, 2\}$$

$$B = \{3, 4\}$$

$$C = \{30, 40\}$$

$$D = \{10, 20\}$$

$$A \times B = \left\{ \begin{array}{l} (1, 3), (1, 4), \\ (2, 3), (2, 4) \end{array} \right\}$$

$$B \rightarrow C = \left\{ \left\{ \begin{array}{l} (3,30), \\ (4,30) \end{array} \right\}, \left\{ \begin{array}{l} (3,30), \\ (4,40) \end{array} \right\}, \left\{ \begin{array}{l} (3,40), \\ (4,30) \end{array} \right\}, \left\{ \begin{array}{l} (3,40), \\ (4,40) \end{array} \right\} \right\}$$

If we know [the values of] a, b, f
then we know [the value of] $(a, f(b))$.

If $(a, b) = (2, 3)$ and $f = \left\{ \begin{array}{l} (3,30), \\ (4,40) \end{array} \right\}$

then $(a, f(b)) = (2, 30)$.

$$\frac{\frac{(a, b)}{a} \pi \quad \frac{\frac{b}{f(b)} \pi' \quad f}{\text{pair}} \text{app}}{(a, f(b))} \quad \frac{\frac{(2, 3)}{2} \pi \quad \frac{\frac{3}{30} \pi' \quad \{(3, 30), (4, 40)\}}{\text{pair}} \text{app}}{(2, 30)}$$

If we know the types of a, b, f
we know the type of $(a, f(b))$.

If we know the types of p, f
we know the type of $(\pi p, f(\pi' p))$.

If we know the types of p, f
we know the type of $(\lambda p : A \times B. (\pi p, f(\pi' p)))$.

$$\frac{\frac{(a, b) : A \times B}{a : A} \pi \quad \frac{\frac{(a, b) : A \times B}{b : B} \pi' \quad f : B \rightarrow C}{f(b) : C} \text{pair}}{(a, f(b)) : A \times C} \text{app}$$

$$\frac{\frac{\frac{p : A \times B}{\pi p : A} \pi \quad \frac{\frac{p : A \times B}{\pi' p : B} \pi' \quad f : B \rightarrow C}{f(\pi' p) : C} \text{pair}}{(\pi p, f(\pi' p)) : A \times C} \text{app}}{(\lambda p : A \times B. (\pi p, f(\pi' p))) : A \times B \rightarrow A \times C} \lambda$$

Types: exercises

Let:

$$A = \{1, 2\}$$

$$B = \{3, 4\}$$

$$C = \{30, 40\}$$

$$D = \{10, 20\}$$

$$f = \left\{ \begin{array}{l} (3,30), \\ (4,40) \end{array} \right\}$$

$$g = \left\{ \begin{array}{l} (1,10), \\ (2,20) \end{array} \right\}$$

Note that $f : B \rightarrow C$ and $g : A \rightarrow D$.

- Evaluate $A \times B$.
- Evaluate $A \rightarrow D$.
- Evaluate $(\pi p, f(\pi' p))$ for each of the four possible values of $p : A \times B$.
- Evaluate $\lambda p:A \times B.(\pi p, f(\pi' p))$.
- Is this true?

$$(\lambda p:A \times B.(\pi p, f(\pi' p))) = \left\{ \begin{array}{l} ((1,3),(1,30)), \\ ((1,4),(1,40)), \\ ((2,3),(2,30)), \\ ((2,4),(2,40)) \end{array} \right\}$$

- Let $p = (2, 3)$. Evaluate $(g(\pi p), f(\pi' p))$.
- Check that if $p : A \times B$ then $(g(\pi p), f(\pi' p)) : D \times C$.
- Check that

$$(\lambda p:A \times B.(g(\pi p), f(\pi' p))) : A \times B \rightarrow D \times C.$$

- Evaluate $(\lambda p:A \times B.(g(\pi p), f(\pi' p)))$.

Type inference

Here is another notation for checking types:

$$\underbrace{\underbrace{(\lambda p : A \times B. (\underbrace{\pi p}_{:A \times B}, \underbrace{f(\underbrace{\pi' p}_{:A \times B})}_{:B \rightarrow C})}_{:A \times C})}_{:A \times B \rightarrow A \times C}}$$

Compare it with:

$$\frac{\frac{\frac{p : A \times B}{\pi p : A} \pi \quad \frac{\frac{p : A \times B}{\pi' p : B} \pi' \quad f : B \rightarrow C}{f(\pi' p) : C} \text{app}}{(\pi p, f(\pi' p)) : A \times C} \text{pair}}{(\lambda p : A \times B. (\pi p, f(\pi' p))) : A \times B \rightarrow A \times C} \lambda$$

Exercise:

Infer the type of each of the terms below (at the right of the ‘:=’).

Use the two notations above.

The types of f, g, h, k are shown in the diagram below.

- $(\times C)f := \lambda p : A \times C. (f(\pi p), \pi' p)$
- $h^b := \lambda q : B \times C. (h(\pi q), (\pi' q))$
- $g^\sharp := \lambda b : B. \lambda c : C. g(b, c)$
- $(C \rightarrow)k := \lambda \varphi : C \rightarrow D. \lambda c : C. k(\varphi c)$

$$\begin{array}{ccc}
 A \times C & \longleftarrow & A \\
 \downarrow (\times C)f & \longleftarrow & \downarrow f \\
 B \times C & \longleftarrow & B \\
 \downarrow \begin{matrix} h^b \\ g \end{matrix} & \longleftarrow & \downarrow \begin{matrix} h \\ g^\sharp \end{matrix} \\
 D & \longrightarrow & C \rightarrow D \\
 \downarrow k & \longleftarrow & \downarrow (C \rightarrow)k \\
 E & \longrightarrow & C \rightarrow E
 \end{array}$$

Term inference

Exercises:

$$\frac{\frac{\frac{p : A \times C}{: A} \pi \quad f : A \rightarrow B}{: B} \text{app} \quad \frac{p : A \times C}{: C} \pi'}{: B \times C} \text{pair}}{: A \times C \rightarrow B \times C} \lambda$$

$$\frac{\frac{\frac{q : B \times C}{: C} \pi' \quad \frac{\frac{q : B \times C}{: B} \pi \quad h : B \rightarrow (C \rightarrow D)}{: C \rightarrow D} \text{app}}{: D} \text{app}}{: B \times C \rightarrow D} \lambda} \text{app}$$

$$\frac{\frac{\frac{b : B \quad c : C}{: B \times C} \text{pair} \quad g : B \times C \rightarrow D}{: D} \text{app}}{: C \rightarrow D} \lambda} \lambda$$

$$\frac{\frac{\frac{c : C \quad \varphi : C \rightarrow D}{: D} \text{app} \quad k : D \rightarrow E}{: E} \text{app}}{: (C \rightarrow E)} \lambda} \lambda$$

Term inference: answers

$$\frac{\frac{\frac{p : A \times C}{\pi p : A} \pi \quad f : A \rightarrow B}{f(\pi p) : B} \text{app} \quad \frac{p : A \times C}{\pi' p : C} \pi'}{(f(\pi p), \pi' p) : B \times C} \text{pair}}{\lambda p : A \times C. (f(\pi p), \pi' p) : A \times C \rightarrow B \times C} \lambda$$

$$\frac{\frac{\frac{q : B \times C}{\pi' q : C} \pi' \quad \frac{\frac{q : B \times C}{\pi q : B} \pi \quad h : B \rightarrow (C \rightarrow D)}{h(\pi q) : C \rightarrow D} \text{app}}{h(\pi q)(\pi' q) : D} \text{app}}{\lambda q : B \times C. h(\pi q)(\pi' q) : B \times C \rightarrow D} \lambda$$

$$\frac{\frac{\frac{b : B \quad c : C}{(b, c) : B \times C} \text{pair} \quad g : B \times C \rightarrow D}{g(b, c) : D} \text{app}}{\lambda c : C. g(b, c) : C \rightarrow D} \lambda}{\lambda b : B. \lambda c : C. g(b, c) : B \rightarrow (C \rightarrow D)} \lambda$$

$$\frac{\frac{\frac{c : C \quad \varphi : C \rightarrow D}{\varphi c : D} \text{app} \quad k : D \rightarrow E}{k(\varphi c) : E} \text{app}}{\lambda c : C. k(\varphi c) : (C \rightarrow E)} \lambda}{\varphi : C \rightarrow D. \lambda c : C. k(\varphi c) : (C \rightarrow D) \rightarrow (C \rightarrow E)} \lambda$$

Contexts and ‘+’

Suppose that A, B, C are known, and are sets.

(Jargon: “fix sets A, B, C ”.)

Then this

$$\underbrace{p : A \times B, f : B \rightarrow C}_{\substack{\text{“context”: a series of} \\ \text{declarations like} \\ \text{var:type}}} \vdash \underbrace{f(\pi'p) : C}_{\text{expr:type}}$$

Means:

“In this context the expression $expr$ makes sense, is not error, and its result is of type $type$.”

Note that calculating $f(\pi'p)$ yields error if we do not know the values of f or p .

What happens if we add contexts to each $term : type$ in a tree?
The two bottom nodes in

$$\frac{\frac{\frac{p : A \times B}{\pi p : A} \pi \quad \frac{\frac{p : A \times B}{\pi'p : B} \pi' \quad f : B \rightarrow C}{f(\pi'p) : C} \text{pair}}{(\pi p, f(\pi'p)) : A \times C} \text{pair}}{(\lambda p : A \times B. (\pi p, f(\pi'p))) : A \times B \rightarrow A \times C} \lambda$$

would become:

$$\frac{f : B \rightarrow C, p : A \times B \vdash (\pi p, f(\pi'p)) : A \times C}{f : B \rightarrow C \vdash (\lambda p : A \times B. (\pi p, f(\pi'p))) : A \times B \rightarrow A \times C}$$

After the rule ‘ λ ’ the ‘ p ’ is no longer needed!

If we add the contexts and omit the types, the tree becomes:

$$\frac{\frac{\frac{p \vdash p}{p \vdash \pi p} \pi \quad \frac{\frac{p \vdash p}{p \vdash \pi'p} \pi' \quad f \vdash f}{f, p \vdash f(\pi'p)} \text{pair}}{f, p \vdash (\pi p, f(\pi'p))} \text{pair}}{f \vdash (\lambda p : A \times B. (\pi p, f(\pi'p)))} \lambda \quad \rightsquigarrow \quad \frac{\frac{\frac{[p]^1}{[p]^1} \pi \quad \frac{[p]^1}{\pi'p} \pi' \quad f}{f(\pi'p)} \text{pair}}{(\pi p, f(\pi'p))} \text{pair}}{(\lambda p : A \times B. (\pi p, f(\pi'p)))} \lambda; 1$$

Notational trick:

below the bar ‘ $\lambda; 1$ ’ the value of p is no longer needed;
we say that the p is “discharged” (from the list of hypotheses)
and we mark the ‘ p ’ on the leaves of the tree with ‘ $[\cdot]^1$ ’;
a ‘ $[\cdot]^1$ ’ on a hypothesis means: “below the bar ‘ $\lambda; 1$ ’ I am no longer a hypothesis”.

Curry-Howard: introduction

We are learning a system called
 “the simply-typed λ -calculus (with binary products)” —
 system $\lambda 1$, for short.

In $\lambda 1$ in its fullest form,
 its objects are trees of ‘ $\dots \vdash term : type$ ’s,
 but we saw (evidence) that we can:

- reconstruct the full tree from just the ‘ $term : type$ ’s,
- write just ‘ $: type$ ’s (except on the leaves, to get the var names),
- reconstruct the full tree from just the bottom ‘ $term : type$ ’...

For example, we can reconstruct the whole tree,
 with contexts, from:

$$\frac{\frac{\frac{[p : A \times B]^1}{: A} \pi \quad \frac{\frac{[p : A \times B]^1}{: B} \pi' \quad f : B \rightarrow C}{: C} \text{app}}{: A \times C} \text{pair}}{: A \times B \rightarrow A \times C} \lambda}$$

If we erase the terms and the ‘:’s and leave only the types,
 we get something that is strikingly similar to a tree in
 Natural Deduction,

$$\frac{\frac{\frac{[A \times B]^1}{A} \pi \quad \frac{\frac{[A \times B]^1}{B} \pi' \quad B \rightarrow C}{C} \text{app}}{A \times C} \text{pair}}{A \times B \rightarrow A \times C} \lambda}{\frac{\frac{[P \wedge Q]^1}{P} \wedge E_1 \quad \frac{\frac{[P \wedge Q]^1}{Q} \wedge E_2 \quad Q \rightarrow R}{R} \rightarrow E}{P \wedge R} \wedge I}{P \wedge R \rightarrow P \wedge Q} \rightarrow I; 1} \rightsquigarrow$$

which talks about *logic*.

Curry-Howard: Natural Deduction

The tree

$$\frac{\frac{\frac{[P \wedge Q]^1}{P} \wedge E_1 \quad \frac{\frac{[P \wedge Q]^1}{Q} \wedge E_2 \quad Q \rightarrow R}{R} \rightarrow E}{P \wedge R} \wedge I}{P \wedge R \rightarrow P \wedge Q} \rightarrow I; 1$$

is in $\text{ND}_{\wedge \rightarrow}$ (or in $\text{IPL}_{\wedge \rightarrow}$), the fragment of Natural Deduction (or intuitionistic predicate logic) that only has the connectives \wedge and \rightarrow .

Its rules are:

$$\frac{P \quad Q}{P \wedge Q} \wedge I \quad \frac{P \wedge Q}{P} \wedge E_1 \quad \frac{P \wedge Q}{Q} \wedge E_2$$

$$\frac{P \quad [Q]^1 \quad \vdots \quad R}{Q \rightarrow R} \rightarrow I; 1 \quad \frac{P \quad P \rightarrow Q}{Q} \rightarrow E$$

New rules (for \top , \perp , \vee):

(not yet — see the whiteboard for 20170418)

Planar Heyting Algebras

Read sections 2–8 of:

<http://angg.twu.net/LATEX/2017planar-has.pdf>

$$\text{Let } B = \begin{array}{ccc} & & 32 \\ & & 22 \\ 21 & 12 & \\ 20 & 11 & 02 \\ 10 & 01 & \\ & & 00 \end{array} .$$

Exercises:

Calculate, and represent in positional notation when possible:

- a) $\lambda lr:B.l$
- b) $\lambda lr:B.r$
- c) $\lambda lr:B.(l \leq 1)$
- d) $\lambda lr:B.(r \geq 1)$
- e) $\lambda lr:B.lr \leq 11$
- f) $\lambda lr:B.lr \wedge 12$
- g) $\lambda lr:B.\text{valid}(\langle l+1, r \rangle)$
- h) $\lambda lr:B.lr \text{ leftof } 11$
- i) $\lambda lr:B.lr \text{ leftof } 12$
- j) $\lambda lr:B.lr \text{ above } 11$
- k) $\lambda lr:B.\text{ne}(lr)$
- l) $\lambda lr:B.\text{nw}(lr)$
- m) $20 \rightarrow 11$
- n) $02 \rightarrow 11$
- o) $22 \rightarrow 11$
- p) $00 \rightarrow 11$
- q) $\lambda lr:B.\neg lr$
- r) $\lambda lr:B.\neg\neg lr$
- s) $\lambda lr:B.(lr = \neg\neg lr)$

Algebraic structures

A *ring* is a 6-uple

$$(R, 0_R, 1_R, +_R, -_R, \cdot_R)$$

where $R, 0_R, \dots, \cdot_R$ have the following types,

$$\begin{aligned} R &\text{ is a set,} \\ 0_R &\in R, \\ 1_R &\in R, \\ +_R &: R \times R \rightarrow R, \\ -_R &: R \rightarrow R \text{ (unary minus),} \\ \cdot_R &: R \times R \rightarrow R, \end{aligned}$$

and where the components obey these equations ($\forall a, b, c \in R$):

$$\begin{aligned} a + 0_R &= 0_R + a = a, & a + b &= b + a, & a + (b + c) &= (a + b) + c, & a + (-a) &= 0, \\ a \cdot 1_R &= 1_R \cdot a = a, & a \cdot b &= b \cdot a, & a \cdot (b \cdot c) &= (a \cdot b) \cdot c, \\ a \cdot (b + c) &= a \cdot b + a \cdot c. \end{aligned}$$

A *proto-ring* is a 6-uple $(R, 0_R, 1_R, +_R, -_R, \cdot_R)$

that obeys the typing conditions of a ring.

A *ring* is a proto-ring plus the assurance that it obeys the ring equations.

A *proto-Heyting Algebra* is a 7-uple

$$H = (\Omega, \leq_H, \top_H, \perp_H, \wedge_H, \vee_H, \rightarrow_H)$$

in which:

$$\begin{aligned} \Omega &\text{ is a set (the "set of truth values"),} \\ \leq_H &\subset \Omega \times \Omega \text{ (partial order),} \\ \top_H &\in \Omega, \\ \perp_H &\in \Omega, \\ \wedge_H &: \Omega \times \Omega \rightarrow \Omega \\ \vee_H &: \Omega \times \Omega \rightarrow \Omega \\ \rightarrow_H &: \Omega \times \Omega \rightarrow \Omega \end{aligned}$$

Sometimes we add operations ‘ \neg ’ and ‘ \leftrightarrow ’ to a (proto-)HA H ,

$$H = (\Omega, \leq_H, \top_H, \perp_H, \wedge_H, \vee_H, \rightarrow_H, \neg_H, \leftrightarrow_H)$$

by defining them as $\neg P := P \rightarrow \perp$ and $P \leftrightarrow Q := (P \rightarrow Q) \wedge (Q \rightarrow P)$

(i.e., $\neg_H P := P \rightarrow_H \perp_H$

and $P \leftrightarrow_H Q := (P \rightarrow_H Q) \wedge_H (Q \rightarrow_H P)$).

This abuse of language is very common:

R “=” $(R, 0_R, 1_R, +_R, -_R, \cdot_R)$.

Protocategories

A *protocategory* is a 4-uple

$$\mathbf{C} = (\mathbf{C}_0, \text{Hom}_{\mathbf{C}}, \text{id}_{\mathbf{C}}, \circ_{\mathbf{C}})$$

where

\mathbf{C}_0 is a set (more precisely a “class”),

$\text{Hom}_{\mathbf{C}} : \mathbf{C}_0 \times \mathbf{C}_0 \rightarrow \mathbf{Sets}$,

$\text{id}_{\mathbf{C}}(A) \in \text{Hom}_{\mathbf{C}}(A, A)$,

$(\circ_{\mathbf{C}})_{ABC} : \text{Hom}_{\mathbf{C}}(B, C) \times \text{Hom}_{\mathbf{C}}(A, B) \rightarrow \text{Hom}_{\mathbf{C}}(A, C)$.

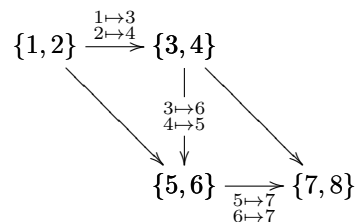
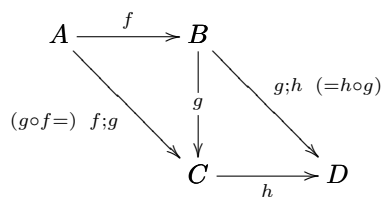
A *category* is a protocategory plus the assurance that identities behave as expected and composition is associative.

Sometimes we add an operation ‘;’ to a category,

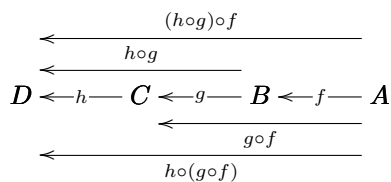
$$\mathbf{C} = (\mathbf{C}_0, \text{Hom}_{\mathbf{C}}, \text{id}_{\mathbf{C}}, \circ_{\mathbf{C}}, ;_{\mathbf{C}})$$

where ‘;’ is the composition in other order: $f \circ g = g; f$.

Composition
(is associative)

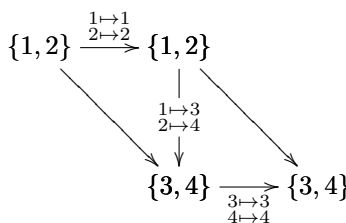
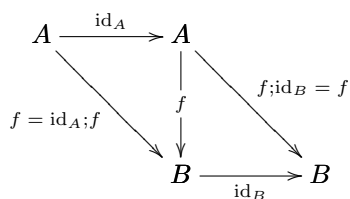


$$\begin{aligned}
 ((h \circ g) \circ f)(a) &= (h \circ g)(f(a)) \\
 &= h(g(f(a))) \\
 &= h((g \circ f)(a)) \\
 &= (h \circ (g \circ f))(a) \\
 ((h \circ g) \circ f)(a) &= (h \circ (g \circ f))(a) \\
 (h \circ g) \circ f &= h \circ (g \circ f)
 \end{aligned}$$



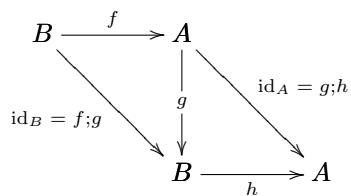
Identities:

If $f : A \rightarrow B$ then $\text{id}_A; f = f = f; \text{id}_B$



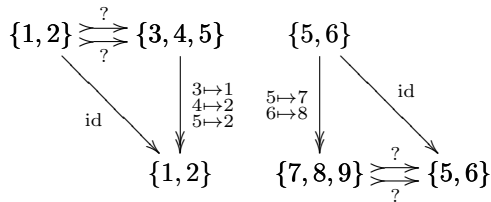
A theorem about lateral inverses:

If $f; g = \text{id}$ and $g; h = \text{id}$ then $f = h$

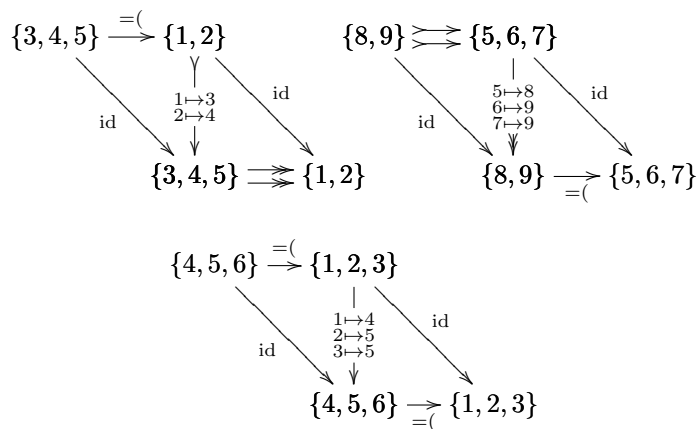


$$\begin{aligned}
 (f; g); h &= \text{id}_B; h = h \\
 f; (g; h) &= f; \text{id}_A = f \\
 f &= f; \text{id}_A \\
 &= f; (g; h) \\
 &= (f; g); h \\
 &= \text{id}_B; h \\
 &= h
 \end{aligned}$$

Multiple inverses



No inverses



Products

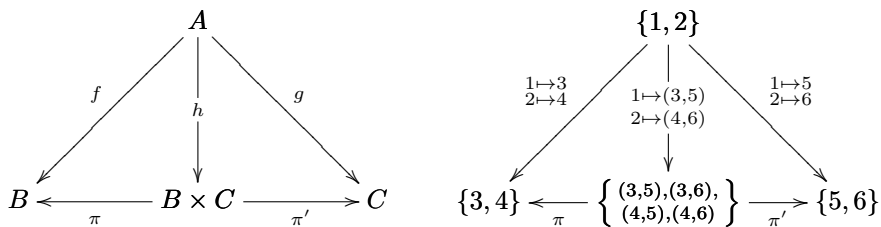
Property: $\forall f, g, \exists! h. (f = h; \pi \ \& \ g = h; \pi')$

Solution: $h = \lambda a : A. (f(a), g(a))$

Bijection: $(f, g) \leftrightarrow h$

$(\rightarrow): \lambda(f, g). (\lambda a : A. (f(a), g(a)))$

$(\leftarrow): \lambda h. ((h; \pi), (h; \pi'))$

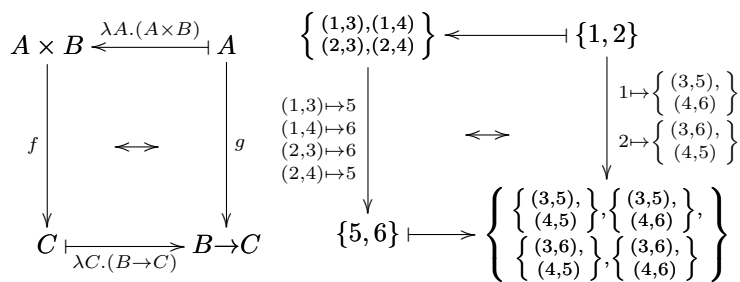


Exponentials

Bijection: $f \leftrightarrow g$

(\rightarrow) (“currying”): $g := \text{cur } f := \lambda a : A. \lambda b : B. f(a, b)$

(\leftarrow) (“uncurrying”): $f := \text{uncur } g := \lambda(a, b) : A \times B. ((g(a))(b))$



Properties: $\text{cur } \text{uncur } f = f$, $\text{uncur } \text{cur } g = g$

where: $f \times f' := \langle \pi; f, \pi'; f' \rangle$, $\text{uncur } g := (g \times \text{id}); \text{ev}$

solving type equations:

$$\frac{\frac{\pi \quad f}{\pi; f} \quad \frac{\pi' \quad f'}{\pi'; f'}}{\langle \pi; f, \pi'; f' \rangle} \text{ren} \quad \frac{\frac{\pi : A \times ? \rightarrow A \quad \pi' : ? \times A' \rightarrow A' \quad f' : A' \rightarrow B'}{\pi; f : A \times ? \rightarrow B} \quad \frac{\pi'; f' : ? \times A' \rightarrow B'}{\pi'; f' : ? \times A' \rightarrow B'}}{\langle \pi; f, \pi'; f' \rangle : A \times A' \rightarrow B \times B'} \text{ren} \\
 \frac{\quad}{f \times f' : A \times A' \rightarrow B \times B'} \text{ren}$$

$$\frac{\frac{g \quad \text{id}}{g \times \text{id}} \quad \text{ev}}{(g \times \text{id}); \text{ev}} \text{ren} \quad \frac{g : A \rightarrow (B \rightarrow C) \quad \text{id} : ? \rightarrow ?}{g \times \text{id} : A \times ? \rightarrow (B \rightarrow C) \times ?} \quad \frac{\text{ev} : (B \rightarrow C) \times B \rightarrow C}{(g \times \text{id}); \text{ev} : A \times ? \rightarrow C}}{\text{uncur } g : A \times ? \rightarrow C}$$