Balanced polymorphism and linear lambda calculus

Noam Zeilberger

MSR-Inria Joint Centre

TYPES 2015
Tallinn, 18 May 2015

a pearl theorem

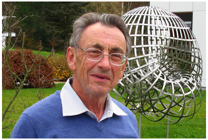
Linear lambda calculus as an extremal case of parametricity:

$$
\begin{array}{ccc}
\lambda x.x(\lambda y.y) & : & ((\alpha \multimap \alpha) \multimap \beta) \multimap \beta \\
\lambda x.\lambda y.x(y) & : & (\alpha \multimap \beta) \multimap (\alpha \multimap \beta) \\
\lambda x.\lambda y.y(x(\lambda z.z)) & : & ((\alpha \multimap \alpha) \multimap \beta) \multimap ((\beta \multimap \gamma) \multimap \gamma) \\
\lambda x.\lambda y.x(\lambda z.z(y)) & : & (((\alpha \multimap \beta) \multimap \beta) \multimap \gamma) \multimap (\alpha \multimap \gamma) \\
\vdots & : & \vdots
\end{array}
$$

Every linear lambda term is (simply-)**typable, and its** ($\beta\eta$-)**normal form is uniquely identified by its principal type.**

Mairson asserts this as a “pearl theorem”.
I believe that the first proof is due to Mints.

- Harry G. Mairson.
  Linear lambda calculus and PTIME-completeness. JFP, 14:6 (2004).
- Grigorii E. Mints. Closed categories and the theory of proofs. Zapiski Nauchnykh Seminarov LOMI im. V.A. Steklova AN SSSR, 68 (1977). Translation in *Journal of Soviet Mathematics*, 15 (1981), republished in Mints, *Selected Papers in Proof Theory*, Bibliopolis (1992).



Grigori Mints
7 June 1939 – 29 May 2014

Mints' key ideas:

1. The principal type of a linear lambda term is *balanced:*

$$
\begin{array}{rcc}
\lambda x.x(\lambda y.y) & : & ((\alpha \multimap {}_\bullet\alpha) \multimap \beta) \multimap {}_\bullet\beta \\
\lambda x.\lambda y.x(y) & : & ({}_\bullet\alpha \multimap \beta) \multimap (\alpha \multimap {}_\bullet\beta) \\
\lambda x.\lambda y.y(x(\lambda z.z)) & : & ((\alpha \multimap {}_\bullet\alpha) \multimap \beta) \multimap (({}_\bullet\beta \multimap \gamma) \multimap {}_\bullet\gamma) \\
\lambda x.\lambda y.x(\lambda z.z(y)) & : & ((({}_\bullet\alpha \multimap \beta) \multimap {}_\bullet\beta) \multimap \gamma) \multimap (\alpha \multimap {}_\bullet\gamma)
\end{array}
$$

2. Any balanced type (more generally, any balanced typing sequent) has at most one inhabitant up to $\beta\eta$.

Proof by induction on length of terms.

Mints' proof is not that complicated, but a pearl theorem deserves a "pearl proof", and **balanced polymorphism** is a recurring pattern...

Polymorphic **CPS typing**:

$$\lambda k.k(t) : \forall R.(A \to R) \to {\bullet}R$$

Semantics of **Separation Logic**:

$$\begin{array}{rcl} w \models \phi * \psi & \text{iff} & \exists w_1, w_2.\,(w = {\bullet}w_1 \circledast {\bullet}w_2) \wedge (w_1 \models \phi) \wedge (w_2 \models \psi) \\ w \models \phi \mathrel{-\!\!*} \tau & \text{iff} & \forall w'.\,({\bullet}w' \models \phi) \supset (w' \circledast w \models \tau) \end{array}$$

Ends and coends in category theory.

I will describe two ways of understanding the pearl theorem:

1. as a simple bijection between *string diagrams* for linear normal forms and provable balanced sequents, and
2. as a simple bidirectional type inference algorithm.

But first some background...

a graphical language for (neutral/normal) linear lambda terms

Described in a recent paper:

- Noam Zeilberger and Alain Giorgetti. A correspondence between rooted planar maps and normal planar lambda terms. To appear in *Logical Methods in Computer Science.*

A rational reconstruction of “lambda-graphs with back-pointers”, and a coloring protocol for *neutral* and *normal* terms.

From reflexive objects to lambda-graphs.

Dana Scott (1980): pure lambda calculus can be modelled by a **reflexive object** in a ccc: an object u and morphisms

$$u \underset{L}{\overset{A}{\rightleftarrows}} u^u$$

such that the $L; A = \mathrm{id}_{u^u}$.

Question: what is a model of pure linear lambda calculus?

- A **monoidal category** is a category $\mathcal{C}$ equipped with a tensor product and unit operation

$$\bullet : \mathcal{C} \times \mathcal{C} \to \mathcal{C} \qquad I : 1 \to \mathcal{C}$$

  associative and unital up to coherent isomorphism.
- It is **closed** if it is also equipped with operations $\backslash : \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathcal{C}$ and $/ : \mathcal{C} \times \mathcal{C}^{\mathrm{op}} \to \mathcal{C}$ right adjoint to the tensor product in each component:

$$\mathcal{C}(y, x \backslash z) \cong \mathcal{C}(x \bullet y, z) \cong \mathcal{C}(x, z / y)$$

- It is **symmetric** if there is a family of isomorphisms

$$\gamma_{x,y} : x \bullet y \xrightarrow{\sim} y \bullet x$$

  involutive in the sense that $(\gamma_{x,y}; \gamma_{y,x}) = \mathrm{id}_{x \bullet y}$ for all $x, y \in \mathcal{C}$, and which satisfy a few additional equations.

In a smcc, left and right residuals are isomorphic, but let us nonetheless distinguish them and give an explicit name

$$\sigma_{x,y} : x \setminus y \xrightarrow{\sim} y \,/\, x$$

to the isomorphism.

Definition

A **linear reflexive object** in a smcc $\mathcal{C}$ is an object $u \in \mathcal{C}$ equipped with a pair of morphisms

$$u \setminus u \xrightarrow{L} u \xrightarrow{A} u \,/\, u$$

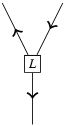
such that $L\,;A = \sigma_{u,u}$.

Idea: recover lambda-graphs by considering a linear reflexive object in a *compact closed category* and applying the machinery of *string diagrams*.

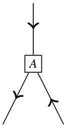
Recall that any compact closed category has left and right residuals defined by $x \setminus y \stackrel{\text{def}}{=} {}^{*}x \bullet y$ and $y \,/\, x \stackrel{\text{def}}{=} y \bullet x^{*}$.

The definition of lro translates into the following components in the graphical language of compact closed categories:

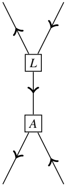
$${}^{*}u \bullet u \xrightarrow{L} u \quad \rightsquigarrow$$



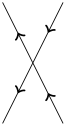
$$u \xrightarrow{A} u \bullet u^{*} \quad \rightsquigarrow$$



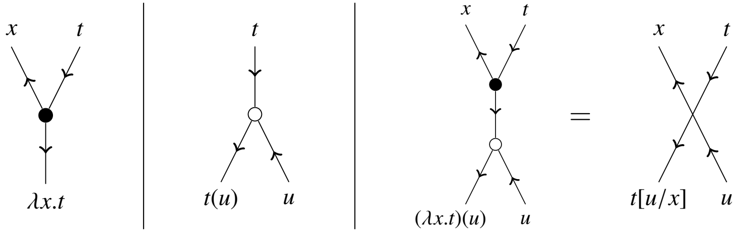
$$L\,;A = \sigma_{u,u} \quad \rightsquigarrow$$



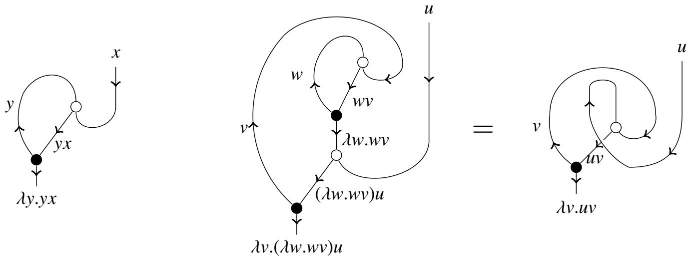
$$=$$



Annotating wires with input/output terms:



Some examples:



A coloring protocol for neutral and normal terms

Recall the standard definition of *neutral* and (β-)*normal* terms:

- Any variable x is neutral.
- If t is neutral and u is normal then $t(u)$ is neutral.
- If t is neutral then t is normal.
- If t is normal then $\lambda x.t$ is normal.

Frank Pfenning (TYPES 1993) gave an elegant reformulation of neutral and normal terms as a *refinement type signature*.

- Frank Pfenning. Refinement Types for Logical Frameworks. In *Informal Proceedings of the Workshop on Types for Proofs and Programs* (ed. Herman Geuvers), 285–299, Nijmegen, The Netherlands, May 1993.

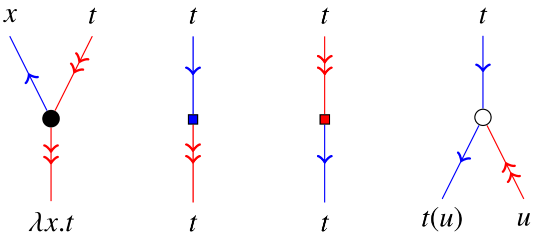
Reformulating Pfenning's reformulation, we introduce the following refinement of the notion of linear reflexive object:

Definition

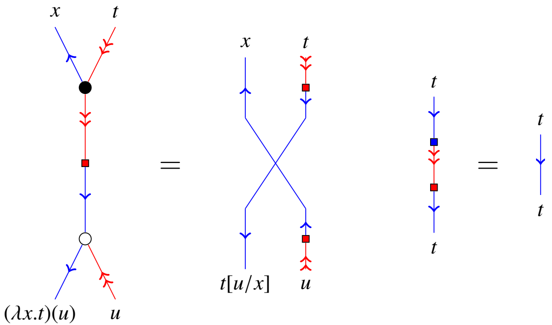
A **linear reflexive pair** in a smcc $\mathcal{D}$ is a pair of objects $B, R \in \mathcal{D}$ equipped with a quadruple of morphisms

$$B \setminus R \xrightarrow{\ell} R \underset{s}{\overset{c}{\rightleftarrows}} B \xrightarrow{a} B \,/\, R$$

such that $s; c = \mathrm{id}_B$ and $\ell; c; a = (\mathrm{id}_B \setminus c); \sigma_{b,b}; (\mathrm{id}_B \,/\, c)$.

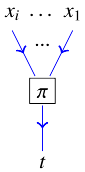


x
t
$\lambda x.t$
t
t
t
t
t
$t(u)$
u

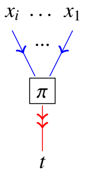


x
t
$(\lambda x.t)(u)$
u
$=$
x
t
$t[u/x]$
u
t
t
$=$
t
t

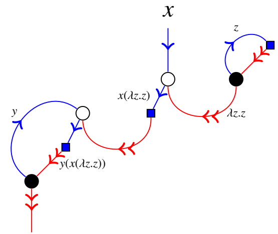
Any neutral or normal linear term (with *i* free variables) can be given a colored string diagram of the form



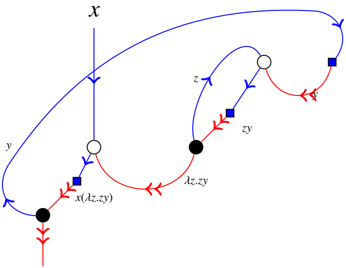
or



which moreover is free of *c*-nodes (= no red boxes).



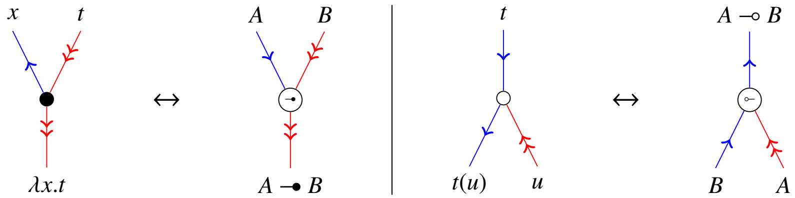
$\lambda y.y(x(\lambda z.z))$



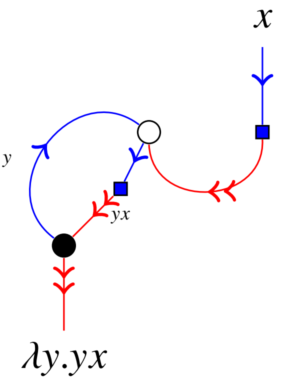
$\lambda y.x(\lambda z.zy)$

relating normal linear terms and balanced principal types

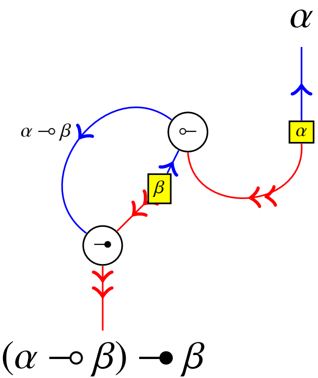
Reverse the orientation of blue wires

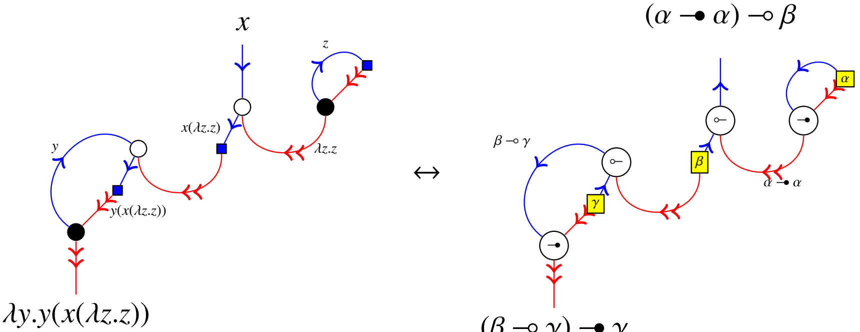


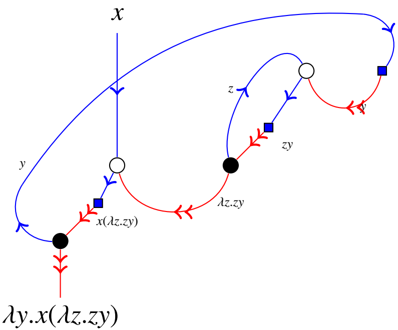
and replace each blue box (*s*-node) by a distinct type variable...



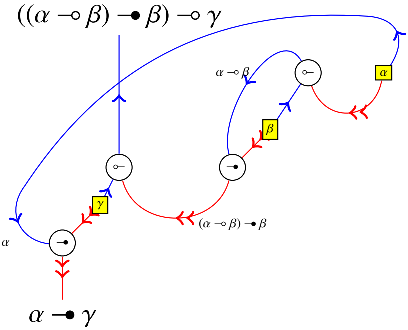
$\leftrightarrow$



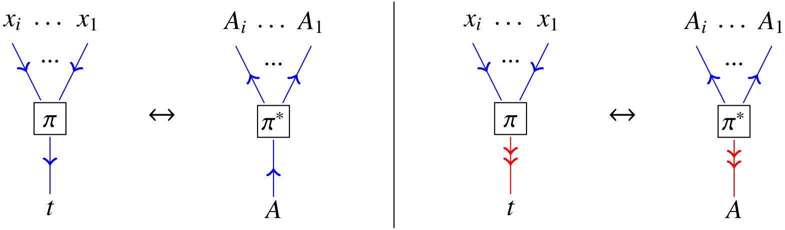

x
z
x(λz.z)
λz.z
y
y(x(λz.z))
λy.y(x(λz.z))
↔
(α −• α) −∘ β
α
β
α −• α
β −∘ γ
γ
(β −∘ γ) −• γ


x
z
zy
y
λz.zy
x(λz.zy)
λy.x(λz.zy)

↔


((α −∘ β) −• β) −∘ γ
α −∘ β
α
β
γ
(α −∘ β) −• β
α
α −• γ

bidirectional type inference


$x_i \ \ldots \ x_1$
π
t
$\leftrightarrow$
$A_i \ \ldots \ A_1$
π^*
A
$x_i \ \ldots \ x_1$
π
t
$\leftrightarrow$
$A_i \ \ldots \ A_1$
π^*
A

Two moded typing judgments:

$\Gamma \Leftarrow R \Leftarrow A$ checking against A, R synthesizes context Γ
$\Gamma \Leftarrow N \Rightarrow A$ N synthesizes type A and context Γ

Well-moded inference rules:

$$\frac{}{x : A \Leftarrow x \Leftarrow A} \qquad \frac{\Gamma \Leftarrow R \Leftarrow A \multimap B \quad \Delta \Leftarrow N \Rightarrow A}{\Gamma, \Delta \Leftarrow R(N) \Leftarrow B}$$

$$\frac{\Gamma \Leftarrow R \Leftarrow \alpha \quad \alpha \text{ fresh}}{\Gamma \Leftarrow R \Rightarrow \alpha} \qquad \frac{x : A, \Gamma \Leftarrow N \Rightarrow B}{\Gamma \Leftarrow \lambda x.N \Rightarrow A \multimap B}$$

This is just dual to standard bidirectional type checking!

$$\Gamma \Leftarrow R \Leftarrow A \quad \leftrightarrow \quad \Gamma \Rightarrow R \Rightarrow A$$

$$\Gamma \Leftarrow N \Rightarrow A \quad \leftrightarrow \quad \Gamma \Rightarrow N \Leftarrow A$$

todo list

- Formal meaning of “reverse the blue arrows”
- Understanding normalization and type annotations
- Pure lambda calculus and intersection types