Towards direct models

# of classical logic

Locali meeting (Beijing, 4-6/11/2013)

Pierre-Louis Curien (CNRS, Paris 7, and INRIA)

(Pictures made using Mimram's string diagram generator strid `http://strid.sourceforge.net`)

Gentzen's sequent calculus

$$A_1, \ldots, A_m \vdash B_1, \ldots, B_n$$

to be read as "the conjunction of the A's implies the disjunction of the B's".

Inference rules :

$$\frac{\Gamma_1 \vdash B_1, \Delta_1 \qquad \Gamma_2 \vdash B_2, \Delta_2}{\Gamma_1, \Gamma_2 \vdash B_1 \otimes B_2, \Delta_1, \Delta_2}$$

(read $\otimes$ as conjunction)

etc...

Link with algebra (operad theory, props, etc. . .)

- view a proof as an operation with many inputs and outputs

- view cut elimination as a composition of these operations

PLAN

I) Linear setting

$$(x \cdot y) \cdot z = x \cdot (y \cdot z)$$

$$\{\{x, y\}, z\} - \{x, \{y, z\}\} = \{\{x, z\}, y\} - \{x, \{z, y\}\}$$

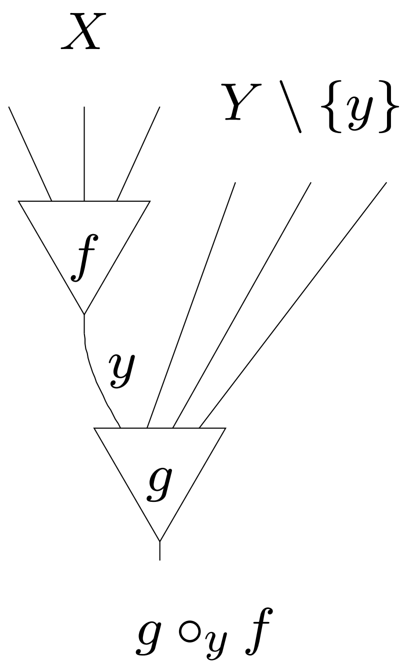
II) Adding merging and dangling wires

$$(\mathtt{k} \cdot x) \cdot y = x \qquad ((\mathtt{s} \cdot x) \cdot y) \cdot z = (x \cdot z) \cdot (y \cdot z)$$

→ a non associative world

Based on works and ideas of Hugo Herbelin, Guillaume Munch, Marcelo Fiore, Paul Downen, Zena Ariola, and myself

Intuitionistic cut elimination, graphically



In operad theory this is known as partial composition. This can be seen as explicit substitution $g[f/y]$.

For the moment, we work at the core level of wiring : no connectives, no types.

The equations of operadic partial composition

sequential and parallel composition, identity

$$
\begin{array}{ll}
(h \circ_z g) \circ_u f = h \circ_z (g \circ_u f) & (h \in P(Z), g \in P(Y), u \in Y) \\
(h \circ_z g) \circ_u f = (h \circ_u f) \circ_z g & (h \in P(Z), g \in P(Y), u \in Z \setminus z)
\end{array}
$$

$$
\begin{array}{l}
g \circ_x id_x = g \\
id_x \circ_x g = g
\end{array}
$$

Implicit in the first equation : if $g \in P(Y)$, the codomain of σ must be disjoint from $Y \setminus y$.

Notation : $P(X)$ is the set of operations whose inputs are named by the elements of X

(there are symmetries involved here – actually P is a species, i.e., a functor from the category of bijections to the category of sets (details omitted))

Free operads

Given a (contravariant) species M, the free operad $F(M)$ is built as follows.

- One constructs formal terms ($y \in Y, X \cap (Y \setminus y) = \emptyset$ in the third rule) :

$$\frac{f \in M(X)}{\underline{f} : X \to \cdot} \qquad \frac{}{id_x : \{x\} \to \cdot} \qquad \frac{u : X \to \cdot \quad t : Y \to \cdot}{t \circ_x u : X \cup (Y \setminus y) \to \cdot}$$

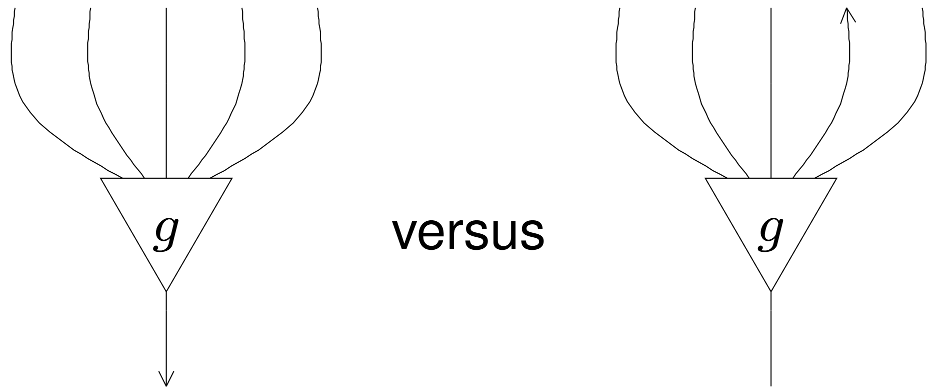
(the notation is meant to reflect the graphical representations as a box with input wires named by X and a single output wire $\cdot$)

One quotients the set of terms by the equations of the previous slide.

We refer to these formal terms as (operadic) combinators (cf. categorical combinators, calculi of explicit substitutions)

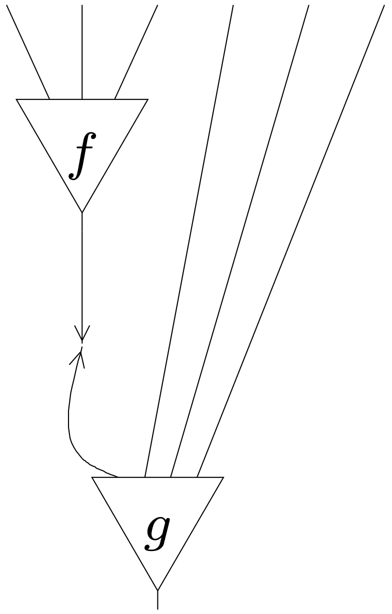
We now give another style of syntax, which is more like the λ-calculus.

Key idea : decompose partial composition in two steps



term context

and then :



$\langle$term | context$\rangle$

Another syntax, with a binder

Two kinds of expressions and typing judgements

- the terms v (which produce an output),

$$v : (X \vdash \cdot)$$

- the contexts e (which expect an input at a designated place)

$$X \mid e \vdash \cdot$$

$$\begin{array}{l} v ::= x \parallel \underline{f}\{v_x \mid x \in X\} \parallel \langle v \mid e \rangle \\ e ::= \tilde{\mu} x.v \end{array}$$

The typing rules

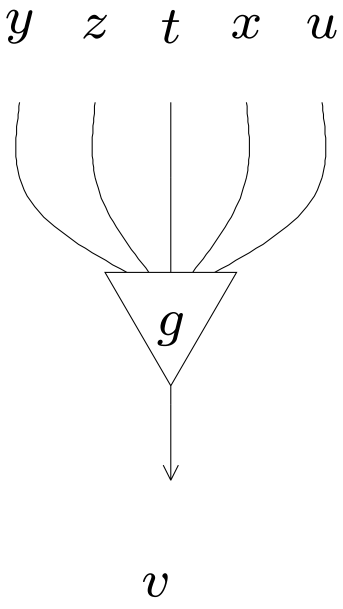
$$\frac{}{x : (\{x\} \vdash \cdot)} \qquad \frac{f \in M(X) \quad \ldots \; v_x : (Y_x \vdash \cdot) \; \ldots}{\underline{f}\{v_x \mid x \in X\} : (\bigcup Y_x \vdash \cdot)} \qquad \frac{v : (X \vdash \cdot) \quad Y \,|\, e \vdash \cdot}{\langle v \mid e \rangle : (X \cup Y \vdash \cdot)}$$

$$\frac{v : (X \vdash \cdot)}{(X \setminus x) \,|\, \tilde{\mu}x.v \vdash \cdot}$$

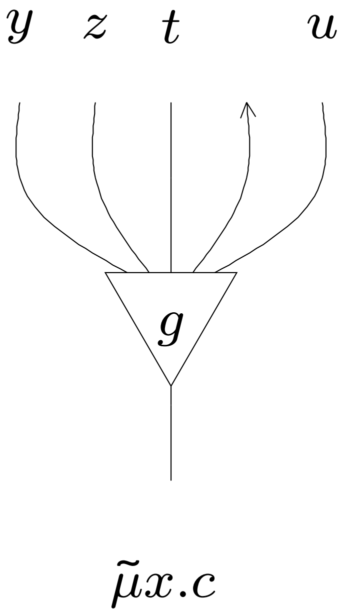
(in the first rule, the Y_x are indexed by X and pairwise disjoint, as are X and Y in the third rule, $x \in X$ in the fourth rule)

(The $\tilde{\mu}$-binder is a syntactic version of the operation ∂ on species : $\partial(M)(X) = M(X + 1)$.)

Pictorially



versus



$v : (\{y, z, t, x, u\} \vdash \cdot)$ $\{y, z, t, u\} \mid \tilde{\mu}x.c \vdash \cdot$

Only one equation !

$$\langle v_1 \mid \tilde{\mu} x . v_2 \rangle = v_2[v_1/x]$$

(plus an equations for symmetries : $\underline{f^\sigma}\{t_x \mid x \in X\} = \underline{f}\{t_{\sigma^{-1}(y)} \mid y \in Y\}$)

The two presentations are equivalent

(Again, cf. Lambek's correspondence CCCs / λ-calculus)

To prove the equivalence of the two presentations, and hence the freeness of the one based on $\tilde{\mu}$, one defines inverse translations ($f \in M(X)$ in the first rule) :

$$\underline{f}_\star = \underline{f}\{x \mid x \in X\} \qquad (id_x)_\star = x \qquad (t \circ_x u)_\star = \langle u_\star \mid \tilde{\mu}x.(t_\star)\rangle$$

and (total composition = sequence of partial compositions in the second rule, translation of contexts indexed by a fresh variable) :

$$[\![x]\!] = id_x \qquad [\![\underline{f}\{v_x \mid x \in X\}]\!] = \underline{f} \circ_{\vec{x}} \overrightarrow{[\![v_x]\!]} \qquad [\![\langle v \mid e\rangle]\!] = [\![e]\!]_y \circ_y [\![v]\!]$$

$$[\![\tilde{\mu}x.v]\!]_y = [\![v[x/y]]\!]$$

A brief look at the proof of equivalence

Verifying the sequential and parallel composition laws is instructive :

$$((h \circ_z g) \circ_u f)_\star = \langle f_\star \mid \tilde{\mu} u.\langle g_\star \mid \tilde{\mu} z.h_\star \rangle\rangle$$

One should then read the two equations as a case statement : graft f on g or on h, depending on where u lies.

(cf. Chapoton-Livernet's construction of the free pre-Lie algebra !).

Free operads (or algebras) via rewriting

Another way to convince ourselves that our syntax “does the job” is to view the second equation as a rewriting rule :

$$\langle v_1 \mid \tilde{\mu}x.v_2 \rangle \to v_2[v_1/x]$$

This defines a confluent and terminating rewriting system (modulo the equality), whose normal forms are the terms produced only by the rules

$$v ::= x \parallel \underline{f}\{v_x \mid x \in X\}$$

i.e. the trees that one can build from the operators (and names at the leaves)

The point here is that we do not need to show that trees form an operad : the construction of the quotient does that for us, and we then *synthesize* the presentation of elements (or elements of the basis of) the free algebra as trees.

Dioperads

We are now interested in operations with multiple inputs and outputs.

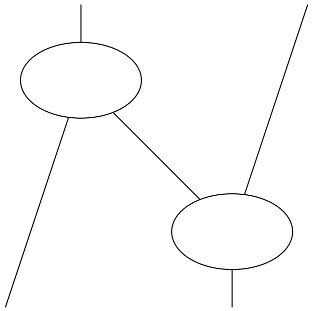
But unlike in more general situations considered by algebraists, like PROPs (the monoidal analogue of Lawvere's algebraic theories), or Vallette's properads, we insist that composition remains definable partially (one output wire plugs into one input wire) = dioperad (Gan).

The corresponding drawings, also known as string diagrams, are simply connected (at most one path between two boxes).

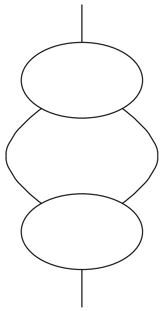
This is because the cut rule has this restricted shape (the one called "allowed" in the next slide).

Pictorially

allowed



not allowed



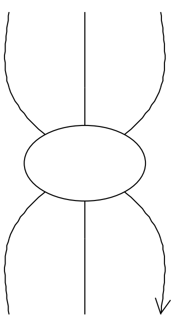
A $\mu - \tilde{\mu}$ syntax for dioperads

(μ plays a role dual to $\tilde{\mu}$ for output wires). There are now three sorts of “operations” :

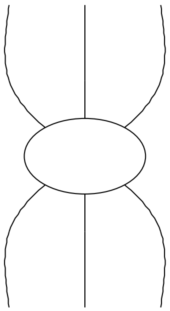
- The operations themselves (commands c)
- the operations one output of which is selected (terms v)
- the operations one input of which is selected (contexts e)

(In the operadic case, we could confuse c and v since there was no choice for the (unique) output wire.)

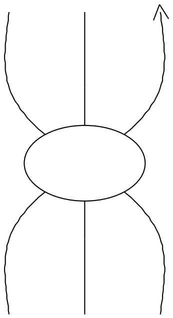
Pictorially



versus



versus



term
$X \vdash v \,|\, A$

command
$c : (X \vdash A)$

context
$X \,|\, e \vdash A$

The $\mu - \tilde{\mu}$ kit

Names for input (resp. output) wires are $x, y, \ldots$ (resp. $\alpha, \beta, \ldots$).

$$c ::= \langle v \,|\, e \rangle \qquad v ::= x \parallel \mu\alpha.c \qquad e = \alpha \parallel \tilde{\mu}x.c$$

($X \cap Y = \emptyset$ and $A \cap B = \emptyset$ in the first rule)

$$\frac{X \vdash v \,|\, A \quad Y \,|\, e \vdash B}{\langle v \,|\, e \rangle : (X \cup Y \vdash A \cup B)} \qquad \frac{}{\{x\} \vdash x \,|} \qquad \frac{}{|\, \alpha \vdash \{\alpha\}}$$

$$\frac{c : (X, x \vdash A)}{X \,|\, \tilde{\mu}x.c \vdash A} \qquad \frac{c : (X \vdash \alpha, A)}{X \vdash \mu\alpha.c \,|\, A}$$

The generating operations come now from $M(X; A)$ and give rise to corresponding commands $\underline{f}\{\ldots, v_x, \ldots, e_\alpha, \ldots\}$. Equations : equivariance +

$$\begin{array}{ll} \langle \mu\alpha.c \,|\, e \rangle = c[e/\alpha] & \quad v = \mu\alpha.\langle v \,|\, \alpha \rangle \\ \langle v \,|\, \tilde{\mu}x.c \rangle = c[v/x] & \quad e = \tilde{\mu}x.\langle x \,|\, e \rangle \end{array}$$

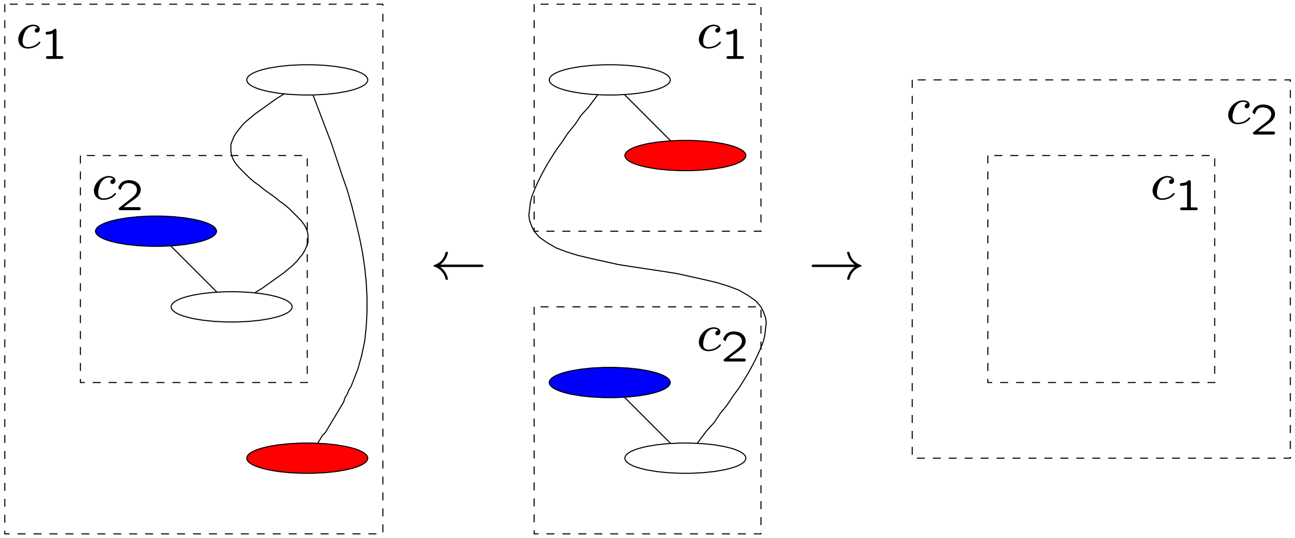
(the equations in the right column say that $\tilde{\mu}$ and $\tilde{\mu}$ have an inverse)

The $\mu - \tilde{\mu}$ critical pair (angelic side)

We have (thinking in terms of rewriting) :

$$c_1[\tilde{\mu}x.c_2/\alpha] \leftarrow \langle \mu\alpha.c_1 \mid \tilde{\mu}x.c_2 \rangle \rightarrow c_2[\mu\alpha.c_1/x]$$

Rewriting is not confluent anymore, but the three expressions describe, i.e. are sequentialisations of, the same underlying string diagram :



Coloured (symmetric) operads (or multicategories)

One replaces finite sets X by C^X (for C a set of colours, or objects). An element of C^X is written $\Gamma = \ldots, x : c, \ldots$ (c is the colour of the input wire named x – different wires can have the same color).

The unique output wire of an operation also has a colour. Thus $P(X)$ is replaced by $P(\Gamma; c)$, and

- If $f \in P(\ldots, x : c, \ldots; d)$ and $g \in P(\Delta; c')$, partial composition must be well-typed, i.e., $c = c'$.

The colored case of *dioperads* goes back to Szabo's polycategories. The operations are indexed by left contexts $\Gamma = \ldots, x : c, \ldots$ and right contexts $\Delta = \ldots, \alpha : d, \ldots$.

Tensor products in multicategories

We say that a (symmetric) multicategory has (non unital) tensor products if there is an operation $(c_1, c_2) \mapsto c_1 \otimes c_2$ on colours together with the following other data :

• for each $\Gamma = \{y : a_y \mid y \in X\}$, a_1, a_2, and $x_1, x_2, x \notin X$ (and $x_1 \neq x_2$), a mapping

$$f \mapsto f_x^{x_1,x_2} : P(\Gamma, x_1 : a_1, x_2 : a_2; c) \to P(\Gamma, x : a_1 \otimes a_2; c)$$

• an operation $\chi_{x_1,x_2} \in P(x_1 : a_1, x_2 : a_2; a_1 \otimes a_2)$ (for distinct x_1, x_2)
• satisfying (kind of adjunction !) :

$$f_x^{x_1,x_2} \circ_x \chi_{x_1,x_2} = f \qquad (g \circ_y \chi_{x_1,x_2})_y^{x_1,x_2} = g$$

(Equivalently, one requires the mappings $f \mapsto f_x^{x_1,x_2}$ to form natural bijections.)

Free multicategories with tensor products

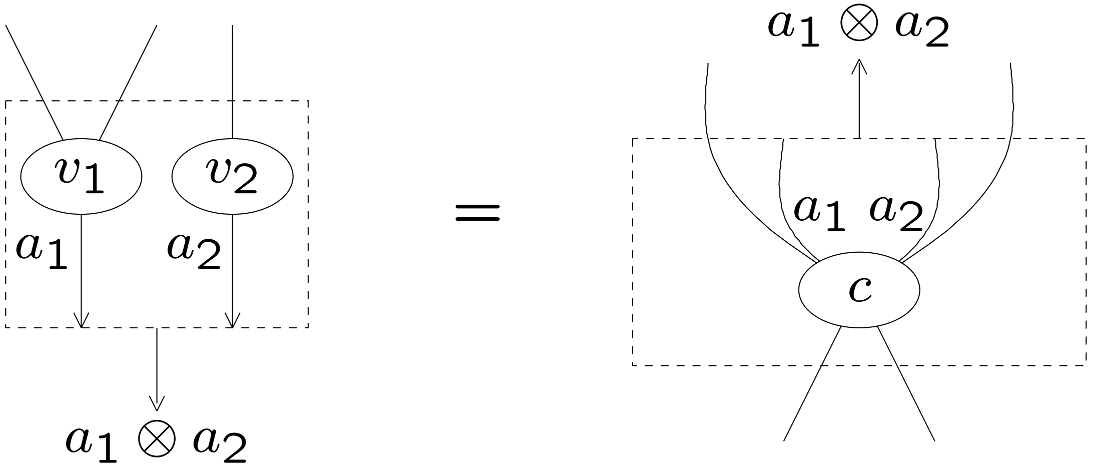
We omit the extended operadic combinator syntax/equations (mimicked from the previous slide) and give here directly an extended $\tilde{\mu}$ style presentation using a *structured* form of binding :

$$v ::= \cdots \parallel (v_1, v_2) \qquad e ::= \cdots \parallel \tilde{\mu}(x_1, x_2).v$$

with the following rules (Γ_1, Γ_2 have disjoint domains in the first rule)

$$\frac{v_1 : (\Gamma_1 \vdash a_1) \quad v_2 : (\Gamma_2 \vdash a_2)}{(v_1, v_2) : (\Gamma_1, \Gamma_2 \vdash a_1 \otimes a_2)} \qquad \frac{v : (\Gamma, x_1 : a_1, x_2 : a_2 \vdash c)}{\Gamma \,|\, \tilde{\mu}(x_1, x_2).v : a1 \otimes a_2 \vdash c}$$

Pictorially



(v_1, v_2) $\tilde{\mu}(x_1, x_2).c$

Cut elimination

We impose the following equations (as usual, when both sides are defined) :

$$\langle (v_1, v_2) \mid \tilde{\mu}(x_1, x_2).v \rangle = \langle v_1 \mid \tilde{\mu}x_1.\langle v_2 \mid \tilde{\mu}x_2.v \rangle \rangle$$

$$e = \tilde{\mu}(x_1, x_2).\langle (x_1, x_2) \mid e \rangle$$

The first equation performs pattern-matching !

One can similarly define coproducts, and then, in the polycategorical setting, the dual constructions “par” (or cotensor) and “with” (or product). (it is known that in a linear setting, there exist two conjunctions, and two disjunctions)

II) General (non linear) case. . .

Now terms are allowed to have repeated variables, or forgotten variables like in the right hand sides of

$$(\mathtt{k} \cdot x) \cdot y = x$$
$$((\mathtt{s} \cdot x) \cdot y) \cdot z = (x \cdot z) \cdot (y \cdot z)$$

The $\mu - \tilde{\mu}$ critical pair (evil side)

Take the $\mu - \tilde{\mu}$ pair with dummy variables on both sides

$$c_1 \leftarrow \langle \mu\_.c_1 \mid \tilde{\mu}\_.c_2 \rangle \rightarrow c_2$$

All operations (proofs, programs) are equal .... This is known as Lafont's critical pair, or ambiguity .
We have to impose some discipline. One is led to introduce a subclass of terms V, called values, and a subclass E of contexts, called covalues, and to restrict the equations as follows :

$$\begin{array}{l}
\langle \mu\alpha.c \mid E \rangle \rightarrow c[E/x] \\
\langle V \mid \tilde{\mu}x.c \rangle \rightarrow c[V/x] \\
E = \tilde{\mu}(x_1, x_2).\langle (x_1, x_2) \mid E \rangle \quad \ldots
\end{array}$$

One requires of course E and V to be designed in such a way that

$$\langle \mu\alpha_1.c_1 \mid \tilde{\mu}x.c_2 \rangle$$

is not a critical pair anymore. E and V should also be stable by substitutions of (co)variables by (co)values.

Polarised approach

One way to do this is through the use of the polarities $(+,-)$ of the formulas / colors, these *determining* in turn what values and covalues are.

These restrictions are called focalising (terminology coming from Andreoli).

The polarities are defined as follows : tensor and coproduct are positive (or eager) , and cotensor and product are negative (or lazy).

Typically, a value of type $A \otimes B$ is a pair (V_1, V_2) of values – not a term like $(5, (\lambda x.x)4))$ = eager pairs. In contrast, a lazy pair is just a pair of two things, each of which might be a very complex non evaluated program.

What is core focalizing system L the internal language of ?

We limit ourselves to one color, one input wire and one output wire, no connectives. Generators are sorted by the polarity of their wires : $f \in M(\delta;\epsilon)$, where $\delta, \epsilon \in \{+,-\}$.

$$\begin{array}{ll} c ::= \langle v^+ \mid e^+ \rangle \parallel \langle v^- \mid e^- \rangle \parallel \underline{f}(v^\delta, e^\epsilon) & \\ v^+ ::= x^+ \parallel \mu\alpha^+.c & v^- ::= x^- \parallel \mu\alpha^-.c \\ e^+ ::= \alpha^+ \parallel \tilde{\mu}x^+.c & e^- ::= \alpha^- \parallel \tilde{\mu}x^-.c \end{array}$$

$$V ::= x^+ \parallel v^- \qquad E ::= \alpha^- \parallel e^+$$

$$\frac{}{x \vdash x \,|} \qquad \frac{c : (x \vdash \alpha)}{x \vdash \mu\alpha.c \,|} \qquad \frac{}{|\, \alpha \vdash \alpha} \qquad \frac{c : (x \vdash \alpha)}{|\, \tilde{\mu}x.c \vdash \alpha}$$

$$\frac{x \vdash v^\delta \,| \qquad |\, e^\delta \vdash \alpha}{\langle v \mid e \rangle : (x \vdash \alpha)} \qquad \frac{x \vdash v^\delta \,| \qquad |\, e^\epsilon \vdash \alpha \qquad f \in M(\delta;\epsilon)}{\underline{f}(v^\delta, e^\epsilon) : (x \vdash \alpha)}$$

Preduploids (Munch)

Only sequential composition $\odot$ is available. We write $\odot = \bullet$ (resp. $\odot = \circ$) when the connecting color is positive (resp. negative). Is it associative ? Consider

$$\begin{array}{r}(c_3 \odot_{z,\beta} c_2) \odot_{y,\alpha} c_1 = \langle \mu\alpha.c_1 \mid \tilde{\mu}y.\langle \mu\beta.c_2 \mid \tilde{\mu}z.c_3\rangle\rangle \\ c_3 \odot_{z,\beta} (c_2 \odot_{y,\alpha} c_1) = \langle \mu\beta.\langle \mu\alpha.c_1 \mid \tilde{\mu}y.c_2\rangle \mid \tilde{\mu}z.c_3\rangle\end{array}$$

One can check that we have

$$(c_3 \bullet c_2) \odot c_1 = c_3 \bullet (c_2 \odot c_1) \quad \text{and} \quad (c_3 \odot c_2) \circ c_1 = c_3 \odot (c_2 \circ c_1)$$

but (cf. Blass' problem in game semantics)

$$(c_3 \circ c_2) \bullet c_1 \quad \neq \quad c_3 \circ (c_2 \bullet c_1)$$

These are known as duplicial algebras (Loday, Loday-Ronco)

With multiple inputs and outputs

When multiple input and output wires are allowed, sequential composition is constrained in the same way as above, and in addition parallel composition is also constrained ($c_3 : (\ldots, x_1, x_2 \vdash \ldots)$, $c_1 : (\ldots \vdash \alpha_1, \alpha_2, \ldots)$, respectively) :

$$(c_3 \circ_{x_1,\alpha_1} c_1) \odot_{x_2,\alpha_2} c_2 = (c_3 \odot_{x_2,\alpha_2} c_2) \circ_{x_1,\alpha_1} c_1$$
$$c_1 \bullet_{x_1,\alpha_1} (c_2 \odot_{x_2,\alpha_2} c_3) = c_1 \odot_{x_2,\alpha_2} (c_2 \bullet_{x_1,\alpha_1} c_3)$$

but

$$(c_3 \bullet_{x_1,\alpha_1} c_1) \bullet_{x_2,\alpha_2} c_2 \neq (c_3 \bullet_{x_2,\alpha_2} c_2) \bullet_{x_1,\alpha_1} c_1$$
$$c_1 \circ_{x_1,\alpha_1} (c_2 \circ_{x_2,\alpha_2} c_3) \neq c_1 \circ_{x_2,\alpha_2} (c_2 \circ_{x_1,\alpha_1} c_3)$$

The resulting structure is a polarised version of polycagegories.

With connectives

They are defined using the *same* universal constructions as in the polycategory case. This works because the restrictions on the equations (values and covalues) are concentrated in the two "control" (non logical) rules

$$\begin{array}{l}\langle \mu\alpha.c \mid E\rangle \to c[E/x] \\ \langle V \mid \tilde{\mu}x.c\rangle \to c[V/x] \\ E = \tilde{\mu}(x_1,x_2).\langle (x_1,x_2) \mid E\rangle \quad \ldots\end{array}$$

As a matter of fact, the equation

$$\langle (v_1,v_2) \mid \tilde{\mu}(x_1,x_2).v\rangle = \langle v_1 \mid \tilde{\mu}x_1.\langle v_2 \mid \tilde{\mu}x_2.v\rangle\rangle$$

for tensors does not refer to values or covalues.

Values and covalues are thus relevant only for the bare unstructured wires, and “control is orthogonal to logic”.

Shifts

Implicit in the polarised connectives are operations $\Uparrow$ and $\Downarrow$ (the *shifts*) such that $\Downarrow A$ is always positive, and $\Uparrow B$ is always negative. We have always

$$A \otimes B = (\Downarrow A) \otimes (\Downarrow B) \qquad \text{(idem } \oplus\text{)}$$

$$A \parr B = (\Uparrow A) \parr (\Uparrow B) \qquad \text{(idem } \&\text{)}$$

With shifts we can encode CBN implication and CBV implication, as well as typed languages allowing for mixed CBN and CBV.

$$M \to_n N = \uparrow (\neg M) \parr N \qquad\qquad P \to_v Q = \downarrow ((\neg P) \parr \uparrow Q)$$

This leads to the following definition (next slide).

Duploids (Munch)

A (poly)duploid is a pre(poly)duploid D equipped with the following additional structure :

- two mappings $\Downarrow$ and $\Uparrow$ from negative and positive colours to positive and negative colours, respectively
- for every positive colour P, two operations $\mathtt{delay} \in D(P; \Uparrow P)$ and $\mathtt{force} \in D(\Uparrow P; P)$
- for every negative colour N, two operations $\mathtt{wrap} \in D(N; \Downarrow N)$ and $\mathtt{unwrap} \in D(\Downarrow N; N)$

satisfying :

$$\begin{array}{l} \mathtt{force} \circ (\mathtt{delay} \bullet_\alpha f) = f \\ (f \circ_x \mathtt{unwrap}) \bullet \mathtt{wrap} = f \\ \mathtt{delay} \bullet \mathtt{force} = id \\ \mathtt{wrap} \circ \mathtt{unwrap} = id \end{array}$$

(we should have written $\mathtt{delay}_{x,\beta} \in D(\{x : P\}; \{\beta : \Downarrow P\})$, and $\bullet_{x,\alpha}$ instead of $\bullet_\alpha$, etc. . .).

Duploids and adjunctions

In his thesis, Guillaume Munch shows that the category of duploids is equivalent to the category of adjunctions (the shifts are adjoint).

Computer science relevance (direct versus indirect style) :

It is well-known that an adjunction gives rise to a monad and a comonad. The semantics of programming languages with effects is typically described with the help of a monad : programs $\Gamma \vdash M : A$ are interpreted as morphisms from Γ to TA, i.e., in the Kleisli category of a monad.

Duploids instead axiomatise directly the relevant properties of these Kleisli (and coKleisli) categories.

Future work

Computer science : Compare with numerous other propositions for giving meaning to programs (or proofs) with effects.

Category theory : relate polyduploids to more standard categorical notions based on (polarised) categories rather than multi or polycategories (cf. work of Hermida relating multicategories and monoidal categories through an adjunction).

Operads : explore other types of “things like operas” (“combinads”