

Models of a Non-Associative Composition

Guillaume Munch-Maccagnoni



LIPN, Université Paris 13

*17th International Conference on Foundations of
Software Science and **Computation Structures** (FoSSaCS)
April 11th 2014*

When composition is not associative

Origins

- Part of my thesis defended in December 2013.
- In 2009-2011: explanation of CPS translations for delimited control operators. Uniform reconstruction of many variants of delimited control operators. (State of the art: Curién-Herbelin's μ and $\tilde{\mu}$ binders, polarisation and focalisation from proof theory.)
- What was new? Composition was not associative!
- Here I show that non-associativity gives a **direct characterisation of polarisation** in correspondence with adjunction-based models of computation.



When composition is not associative

In computer science

$$(h \circ g) \cdot f \neq h \circ (g \cdot f)$$

- : composition in call by value
- : composition in call by name

ML `let y = f x in h (fun () -> g y) ≠ h (fun () -> g (f x))`

Haskell `(\y->h (g y)) $! (f x) ≠ h (g $! (f x))`



When composition is not associative

In computer science

Evidence of polarisation

- Implementing call-by-name in call-by-value (Hatcliff and Danvy) or call-by-value in call-by-name (?)
- Value restriction for polymorphism, context restriction for existential types...

Idea

Distinction between *strict* and *lazy* types inside the same programming language.

(Indirect: Levy, Zeilberger. Direct: Murthy, M.-M.)



When composition is not associative

In Game semantics: the Blass problem

Blass problem Failure of the associativity of composition when the middle morphism is of type $P \rightarrow N$

Samson Abramsky. **Sequentiality vs. concurrency in games and logic**. *Math. Struct. Comput. Sci.*, 13(4):531–565, 2003

Paul-André Melliès. **Asynchronous Games 3 An Innocent Model of Linear Logic**. *Electr. Notes Theor. Comput. Sci.*, 122:171–192, 2005

following:

Andreas Blass. **A game semantics for linear logic**. *Ann. Pure Appl. Logic*, 56(1-3):183–220, 1992



When composition is not associative

In logic

Polarisation = Making the distinction between positive ($\exists, \vee \dots$) and negative ($\forall, \rightarrow \dots$) connectives formal.

- Focalisation in proof search (Andreoli)
- Type isomorphism $A \simeq \neg\neg A$ in classical logic? (Girard)
(See my companion CSL-LICS paper, *On the constructive interpretation of an involutive negation*)
- Disjunction in intuitionistic logic? (Girard)
- Categorical structure of game semantics? (Melliès)

The interpretation in a **category** is not immediate.



Direct models

Direct denotational model: Exact correspondence between operations of the model and constructions of the language.

Example

CCCs for simply-typed λ -calculus

- Moggi's λ_C -models of call by value (strong monad + Kleisli exponentials) are indirect, but:

Example

Thunks (+ pre-monoidal structure & exponents) model call-by-value directly.



Direct models

- **Thunks** implement call by name in call by value.
- A *thunk* L is a **co-monad** such that every morphism $f : A \rightarrow B$ has a co-extension $*f : A \rightarrow LB$. (Usually only true for $f : LA \rightarrow B$.)
Think $LA = \text{unit} \rightarrow A$.
- Correspondence between direct models of call-by-value and Moggi's monad-based models:

Carsten Führmann. **Direct Models for the Computational Lambda Calculus**. *Electr. Notes Theor. Comput. Sci.*, 20:245–292, 1999





Direct models

- **Duploids** generalise thunks.
- They mix strict types and lazy types.
- They generalise call-by-value and call-by-name.
- They are in correspondence with **adjunctions**.



Duploid construction

Let $\uparrow \dashv \downarrow : n \rightarrow \mathcal{P}$ be an adjunction:

$$\frac{P \rightarrow \downarrow N}{\uparrow P \rightarrow N} (\simeq)$$

$(\downarrow : n \rightarrow \mathcal{P})$

P, Q objects of \mathcal{P} and N, M objects of n

Example Structure of CPS: Adjunction of negation with itself:

$$\frac{P \rightarrow \neg Q}{\neg P \leftarrow Q} (\simeq)$$

Duploid construction Recipe for defining a notion of morphism $A \rightarrow B$ with A, B from *either* category \mathcal{P} or n .



Duploid construction

Definition An *oblique morphism* $f : P \rightarrow_{\mathcal{D}} N$
is (equivalently) either $P \rightarrow \downarrow N$ or $\uparrow P \rightarrow N$

Negative composition

$$\frac{\frac{f : P \rightarrow_{\mathcal{D}} N}{f : P \rightarrow \downarrow N} (\simeq)}{\frac{g : \downarrow N \rightarrow_{\mathcal{D}} M}{g : \downarrow N \rightarrow \downarrow M} (\simeq)} (\circ)$$

$$\frac{g \circ f : P \rightarrow \downarrow M}{g \circ f : P \rightarrow_{\mathcal{D}} M} (\simeq)$$

Positive composition

$$\frac{\frac{f : P \rightarrow_{\mathcal{D}} \uparrow Q}{f : \uparrow P \rightarrow \uparrow Q} (\simeq)}{\frac{g : Q \rightarrow_{\mathcal{D}} N}{g : \uparrow Q \rightarrow N} (\simeq)} (\bullet)$$

$$\frac{g \bullet f : \uparrow P \rightarrow N}{g \bullet f : P \rightarrow_{\mathcal{D}} N} (\simeq)$$



Duploid construction

We define:

$$\begin{aligned}
 P \rightarrow_{\mathcal{D}} Q &\stackrel{\text{def}}{=} P \rightarrow_{\mathcal{D}} \uparrow Q \\
 N \rightarrow_{\mathcal{D}} M &\stackrel{\text{def}}{=} \downarrow N \rightarrow_{\mathcal{D}} M \\
 N \rightarrow_{\mathcal{D}} P &\stackrel{\text{def}}{=} \downarrow N \rightarrow_{\mathcal{D}} \uparrow P
 \end{aligned}$$

Thus:

$g \circ f$ composition of $A \xrightarrow{f}_{\mathcal{D}} P \xrightarrow{g}_{\mathcal{D}} B$

$g \circ f$ composition of $A \xrightarrow{f}_{\mathcal{D}} N \xrightarrow{g}_{\mathcal{D}} B$

Hint Generalises the Kleisli constructions of the monad $\downarrow\uparrow$ and of the co-monad $\uparrow\downarrow$.



Duploid construction

Generalises the *polarised translation* of classical logic.

Jean-Yves Girard. *A new constructive logic: Classical logic*.
Math. Struct. Comp. Sci., 1(3):255–296, 1991

Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx.
A New Deconstructive Logic: Linear Logic. *Journal of
Symbolic Logic*, 62 (3):755–807, 1997

Olivier Laurent. *Etude de la polarisation en logique*. Thèse
de doctorat, Université Aix-Marseille II, mar 2002



Intuitions

$f \circ g$ first computes f while $f \bullet g$ first computes g

Hence associativity:

$$(h \bullet g) \bullet f = h \bullet (g \bullet f)$$

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

$$(h \bullet g) \circ f = h \bullet (g \circ f)$$

But:

$$(h \circ g) \bullet f \neq h \circ (g \bullet f) \text{ in general}$$

See Loday's **duplicial algebras**:

Jean-Louis Loday. **Generalized bialgebras and triples of operads.**

arXiv preprint math/0611885, 2006



Intuitions

In practice, omit parentheses in any sequence of the form:

$$f_1 \bullet \cdots \bullet f_i \circ \cdots \circ f_n$$

Remaining parentheses denote sequencing
(think **boxes** in proof nets)





Need for cleanliness

$$A \xrightarrow{f} \mathcal{D} B \xrightarrow{g} \mathcal{D} C$$

2 polarities for each of A, B, C

= 8 cases ?



Comparative table

Evaluation order	By value	By name	Polarised
Indirect model	Monad T	Co-monad L	Adjunction $F \dashv G$
Direct model	Thunk (Führmann)	Runnable monad (e.g. $\neg\neg$ with $C : \neg\neg A \rightarrow A$)	Duploid
Programs	Kleisli maps $P \rightarrow TQ$	co-Kleisli maps $LN \rightarrow M$	Oblique maps $\uparrow P \rightarrow N$ $\simeq P \rightarrow \downarrow N$
Syntactic data	Values	Stacks	Both
Completion into	Thunkable expressions	Linear evalua- tion contexts	Both



Magmoids

 \mathcal{D}

Objects A, B

Morphisms $f : A \rightarrow B, g : B \rightarrow C \dots$

Composition $g \circ f : A \rightarrow C$

Identity id_A neutral for \circ

(category = above + associativity)

“unital magmoid”



Magmoids

Linear and thunkable morphisms

Definition

Linear morphism f associates to its right

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

Thunkable morphism f associates to its left

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

\mathcal{D}_l category of linear morphisms

\mathcal{D}_t category of thunkable morphisms

Proposition

Hom-functor

$$\mathcal{D}(-, =) : \mathcal{D}_t^{\text{op}} \times \mathcal{D}_l \rightarrow \mathbf{Set}$$



Pre-duploids

Definition

Pre-duploid \mathcal{D}

Objects, morphisms, composition, identity

Polarities Mapping $\pi : \text{Obj}(\mathcal{D}) \rightarrow \{+, \ominus\}$ such that:

Every $f : A \rightarrow N$ is thunkable.

“f is called by name”

Every $g : P \rightarrow A$ is linear.

“g calls by value”

No need to reason by cases on polarities :

Morphisms are treated uniformly via the hom-functor $\mathcal{D}(-, =)$.



Duploids

characterise a “Blass phenomenon”

Definition

Duploid \mathcal{D}

Pre-duploid $\mathcal{D} + \text{Shifts}$

- For every P a negative object $\uparrow P$,
- For every N a positive object $\downarrow N$,
- Thinkable morphisms $\text{wrap}_N : N \rightarrow \downarrow N$,
- Linear morphisms $\text{force}_P : \uparrow P \rightarrow P$,
- Inverses:

$$\text{unwrap}_N = \text{wrap}_N^{-1} : \uparrow N \rightarrow N$$

$$\text{delay}_P = \text{force}_P^{-1} : P \rightarrow \uparrow P.$$



Duploids

Main lemma

Thunkability and linearity are characterised locally:

Proposition

$f \in \mathcal{D}(A, P)$ is *thunkable* iff:

$$(\text{wrap}_{\uparrow P} \circ \text{delay}_P) \bullet f = \text{wrap}_{\uparrow P} \circ (\text{delay}_P \bullet f)$$

$f \in \mathcal{D}(N, B)$ is *linear* iff:

$$f \circ (\text{unwrap}_N \bullet \text{force}_{\downarrow N}) = (f \circ \text{unwrap}_N) \bullet \text{force}_{\downarrow N}$$





Structure of Shifts

\mathcal{N} sub-category of morphisms $N \rightarrow M$.

\mathcal{P} sub-category of morphisms $P \rightarrow Q$.

Proposition

- \Uparrow extends into an equivalence of categories $\mathcal{D}_1 \xrightarrow{\cong} \mathcal{N}_1$
- \Downarrow extends into an equivalence of categories $\mathcal{D}_t \xrightarrow{\cong} \mathcal{P}_t$

This characterises duploids.



Structure of Shifts

Corollary

1. *Adjunction* $\Downarrow \dashv \Uparrow$
2. *Adjunction* $\Uparrow \dashv \Downarrow$ *when restricted to linear and thunkable morphisms.*

For those who are familiar: Distinguishes our approach from continuation-passing style or focusing (as in Laurent or Zeilberger)
They are based on adjunctions of the form $\uparrow \dashv \downarrow$.



Duploid functors

Definition

Functor $\mathcal{D} \rightarrow \mathcal{D}'$

- $F : \text{Obj}(\mathcal{D}) \rightarrow \text{Obj}(\mathcal{D}')$ that preserves polarities.
- $f : A \rightarrow B \Rightarrow Ff : FA \rightarrow FB$
- $\text{Fid}_A = \text{id}_{FA}$
- $F(f \circ g) = Ff \circ Fg$
- $F\text{force}_P$ is linear and $F\text{wrap}_N$ is thunkable

Remark No ad hoc strictness condition unlike Führmann's functors of Think-force categories.



Duploid functors

(Functor $F : \mathcal{C} \rightarrow \mathcal{C}'$ between categories: natural transformation $\mathcal{C}(-, =) \rightarrow \mathcal{C}'(F-, F=)$)

Proposition

Let F be a function on objects and on morphisms. F is a duploid functor $\mathcal{D} \rightarrow \mathcal{D}$ if and only if:

1. F preserves linearity
2. F preserves thunkability
3. F is a natural transformation:

$$F_l : \mathcal{D}_l \rightarrow \mathcal{D}'_l$$

$$F_t : \mathcal{D}_t \rightarrow \mathcal{D}'_t$$

$$F : \mathcal{D}(-, =) \rightarrow \mathcal{D}'(F_t-, F_l=)$$



Main result

Dupl: Category of duploids and duploid functors

Adj: Category of adjunctions and *pseudo-maps of adjunctions*

Theorem

There is a reflection:

$$\mathbf{Dupl} \triangleleft \mathbf{Adj}$$

i.e., the duploid construction extends into a functor $j : \mathbf{Adj} \rightarrow \mathbf{Dupl}$ that admits a full and faithful right adjoint $i : \mathbf{Dupl} \rightarrow \mathbf{Adj}$.



Main result

In more details:

1. The duploid construction extends into a functor $j : \mathbf{Adj} \rightarrow \mathbf{Dupl}$;
2. Every duploid arises in this way (functor $i : \mathbf{Dupl} \rightarrow \mathbf{Mon}$ such that $ji\mathcal{D} \simeq \mathcal{D}$);
3. There is an adjunction $j \dashv i$.
The unit maps an adjunction $\uparrow \dashv \downarrow$ to a completed adjunction $ij(\uparrow \dashv \downarrow)$.
4. We characterise the completion. Duploids correspond to adjunctions satisfying an *equalising requirement*.

$$\mathbf{Dupl} \simeq \mathbf{Adj}_{eq}$$

An adjunction in \mathbf{Adj}_{eq} has all the linear and thunkable morphisms in the sense of duploids.



Main result

Additional results

1. *Depolarisation* condition: the duploid is a category if and only if the adjunction is **idempotent**.
2. Kleisli categories are exactly duploids where \uparrow (for monads) or \downarrow (for co-monads) are bijective on objects. The structure dual to thunks are *runnable monads* which implement call-by-value in call-by-name.
3. Internal language based on Curien-Herbelin's μ and $\tilde{\mu}$, *polarisation* and *focalisation*. A core language for abstract machines and sequent calculus. Applied to study **focalisation**, **CPS translations** and **classical logic** in my Ph.D. thesis. (Also related: Paul Downen's talk at ESOP on last Wednesday)



Conclusion

Polarisation everywhere Indirect, Call-by-name, Call-by-value...
The various biases of denotational semantics are a way of **hiding** the fact that composition is not always associative *a priori*.

Internal language inspired from Curien-Herbelin's $\bar{\lambda}\mu\tilde{\mu}$ enriched with polarities (M.-M., CSL'09).
Curien and Herbelin's $\bar{\lambda}\mu\tilde{\mu}$ **scales** gracefully towards richer models of computation, better than the λ calculus — makes direct term languages a potent approach.



And more !

Includes work in progress with Marcelo Fiore and Pierre-Louis Curien

Very simple syntax for connectives (only β and η rules)

Suggests an elegant characterisation in terms of **universal properties** over duploids.

Equational reasoning with thunkable and linear morphisms The completion of values and stacks has good syntactic properties. (Clean semantic notion of *stoup*.)

Direct models of polarised intuitionistic logic / lambda-calculus with extensional sums / call-by-push-value.

No reason for composition to be associative without strong normalisation.



Thank you

Pseudo-morphisms of adjunctions

Bart Jacobs. *Comprehension categories and the semantics of type dependency*. *Theor. Comput. Sci.*, 107(2):169–207, 1993

Definition

Let $F \dashv_{(\eta, \varepsilon)} G : \mathcal{C}_1 \rightarrow \mathcal{C}_2$ and $F' \dashv_{(\eta', \varepsilon')} G' : \mathcal{C}'_1 \rightarrow \mathcal{C}'_2$ be two adjunctions. A *pseudo-morphism of adjunctions*:

$$(H_1, H_2, \phi, \psi) : (F \dashv_{(\eta, \varepsilon)} G) \rightarrow (F' \dashv_{(\eta', \varepsilon')} G')$$

is given by a pair of functors $H_1 : \mathcal{C}_1 \rightarrow \mathcal{C}'_1$ and $H_2 : \mathcal{C}_2 \rightarrow \mathcal{C}'_2$ and a pair of natural isomorphisms $\phi : F'H_2 \xrightarrow{\cong} H_1F$ and $\psi : G'H_1 \xrightarrow{\cong} H_2G$, such that H_1 and H_2 preserve η and ε up to isomorphism:

$$H_2\eta = \psi_F \circ G'\phi \circ \eta'_{H_2} \qquad H_1\varepsilon = \varepsilon'_{H_1} \circ F'\psi^{-1} \circ \phi_G^{-1}.$$

The composition of (H_1, H_2, ϕ, ψ) with $(H'_1, H'_2, \phi', \psi')$ is defined as:

$$(H'_1, H'_2, \phi', \psi') \circ (H_1, H_2, \phi, \psi) = (H'_1H_1, H'_2H_2, H'_1\phi \circ \phi'_{H_2}, H'_2\psi \circ \psi'_{H_1})$$

Exercise

Let (E, \circ, e^\ominus) and (E, \bullet, e^+) be two monoids on the same set E , that satisfy the following mixed associativity rule:

$$\forall x, y, z \in E, x \bullet (y \circ z) = (x \bullet y) \circ z$$

Let $x \in E$. Show that the following two properties are equivalent:

$$x \bullet e^\ominus = x \circ (e^\ominus \bullet e^\ominus) \tag{1}$$

$$\forall y, z \in E, (x \circ y) \bullet z = x \circ (y \bullet z) \tag{2}$$

x is linear

Exercise

Symmetrically the following two propositions are equivalent:

$$(e^+ \circ e^+) \bullet x = e^+ \circ x$$

$$\forall y, z \in E, (z \circ y) \bullet x = z \circ (y \bullet x)$$

x is thinkable : without side-effect

Answer to the Exercise

$$x \bullet e^- = x \circ (e^- \bullet e^-) \quad (1)$$

$$\forall y, z \in E, (x \circ y) \bullet z = x \circ (y \bullet z) \quad (2)$$

Quite trivially one has (2) \Rightarrow (1). We first prove that we have:

$$(1) \Rightarrow \forall y \in E, x \circ (e^- \bullet y) = x \bullet y$$

Proof.

Assume (1) and let $y \in E$. We have:

$$\begin{aligned} x \circ (e^- \bullet y) &= x \circ \underline{(e^- \bullet (e^- \circ y))} \\ &= \underline{x \circ ((e^- \bullet e^-) \circ y)} \\ &= \underline{(x \circ (e^- \bullet e^-)) \circ y} \\ &= \underline{(x \bullet e^-) \circ y} \\ &= x \bullet \underline{(e^- \circ y)} \\ &= x \bullet y \end{aligned}$$



Answer to the Exercise

Proof of (1) \Rightarrow (2).

Let $y, z \in E$. We have:

$$\begin{aligned}x \circ (\underline{y} \bullet z) &= x \circ ((\underline{e^-} \circ y) \bullet z) \\&= x \circ (((\underline{e^- \cdot e^+}) \circ y) \bullet z) \\&= x \circ ((\underline{e^- \cdot (e^+ \circ y)}) \bullet z) \\&= \underline{x \circ (e^- \cdot ((e^+ \circ y) \bullet z))} \\&= \underline{x \cdot ((e^+ \circ y) \bullet z)} && \text{as previously} \\&= (\underline{x \cdot (e^+ \circ y)}) \bullet z \\&= (x \circ (\underline{e^- \cdot (e^+ \circ y)})) \bullet z && \text{as previously} \\&= (x \circ ((\underline{e^- \cdot e^+}) \circ y)) \bullet z \\&= (x \circ (\underline{e^- \circ y})) \bullet z \\&= (x \circ y) \bullet z\end{aligned}$$



The syntactic unital magmoid

Terms, contexts, commands:

$$t ::= x \mid \mu\alpha.c \mid \dots$$

$$e ::= \alpha \mid \tilde{\mu}x.c \mid \dots$$

$$c ::= \langle t \parallel e \rangle$$

$$\frac{\frac{\frac{\overline{x : A \vdash x : A} \quad \overline{\alpha : A \vdash \alpha : A}}{c : (x : A \vdash \Delta)} \quad \overline{\Gamma \vdash \mu\alpha.c : A}}{\Gamma \vdash t : A} \quad \overline{\Gamma \vdash \tilde{\mu}x.c : A \vdash \Delta}}{\langle t \parallel e \rangle : (\Gamma \vdash \Delta)}$$

(Reads as a type system, from top to bottom!)

The syntactic unital magmoid

Composition:

$$\text{let } x \text{ be } t \text{ in } u \stackrel{\text{def}}{=} \mu\alpha.\langle t \parallel \tilde{\mu}x.\langle u \parallel \alpha \rangle \rangle$$

Variables are values:

$$V ::= x \mid \dots$$

$$\pi ::= \alpha \mid \dots$$

Reductions and expansions:

$$\langle V \parallel \tilde{\mu}x.c \rangle \triangleright c[V/x]$$

$$e \triangleright \tilde{\mu}x.\langle x \parallel e \rangle$$

$$\langle \mu\alpha.c \parallel \pi \rangle \triangleright c[\pi/\alpha]$$

$$t \triangleright \mu\alpha.\langle t \parallel \alpha \rangle$$

One has:

$$\text{let } x \text{ be } y \text{ in } t \simeq t[y/x]$$

$$\text{let } y \text{ be } t \text{ in } y \simeq t$$

The syntactic pre-duploid

Now variables are either positive or negative:

$$\frac{}{x^+ : P \vdash x^+ : P} \quad \frac{}{x^\ominus : N \vdash x^\ominus : N}$$
$$\frac{}{\alpha^+ : P \vdash \alpha^+ : P} \quad \frac{}{\alpha^\ominus : N \vdash \alpha^\ominus : N}$$

Terms and contexts are either positive or negative:

$$t_+ ::= x^+ \mid \mu\alpha^+.c \mid \dots \quad t_\ominus ::= x^\ominus \mid \mu\alpha^\ominus.c \mid \dots$$
$$e_+ ::= \alpha^+ \mid \tilde{\mu}x^+.c \mid \dots \quad e_\ominus ::= \alpha^\ominus \mid \tilde{\mu}x^\ominus.c \mid \dots$$
$$c ::= \langle t_+ \parallel e_+ \rangle \mid \langle t_\ominus \parallel e_\ominus \rangle$$

The syntactic pre-duploid

Negative terms are values, positive contexts are stacks:

$$V ::= x^+ \mid t_\ominus \mid \dots$$

$$\pi ::= \alpha^\ominus \mid e_+ \mid \dots$$

In particular:

$$\langle \mu\alpha^\ominus.c \parallel \tilde{\mu}x^\ominus.c' \rangle \triangleright c'[\mu\alpha^\ominus.c/x^\ominus] \quad x^\ominus \text{ is called by name}$$

$$\langle \mu\alpha^+.c \parallel \tilde{\mu}x^+.c' \rangle \triangleright c[\tilde{\mu}x^+.c'/\alpha^+] \quad x^+ \text{ is called by value}$$

Proposition

Associativity of composition:

let y be (let x be t in u) in $v \simeq$ let x be t in let y be u in v

unless t is positive and u is negative.

The syntactic duploid

Coercions: $V_+ ::= \dots | \{t_\ominus\} | \dots$ $t_\ominus ::= \dots | \mu\{\alpha^+\}.c | \dots$
 $e_+ ::= \dots | \tilde{\mu}\{x^\ominus\}.c | \dots$ $\pi_\ominus ::= \dots | \{e_+\} | \dots$

$$\frac{\Gamma \vdash t_\ominus : N}{\Gamma \vdash \{t_\ominus\} : \Downarrow N}$$

$$\frac{e_+ : P \vdash \Delta}{\{e_+\} : \Uparrow P \vdash \Delta}$$

$$\frac{c : (x^\ominus : N \vdash \Delta)}{|\tilde{\mu}\{x^\ominus\}.c : \Downarrow N \vdash \Delta|}$$

$$\frac{c : (\Gamma \vdash \alpha^+ : P)}{\Gamma \vdash \mu\{\alpha^+\}.c : \Uparrow P}$$

New reductions and expansions:

$$\langle \{t_\ominus\} \parallel \tilde{\mu}\{x^\ominus\}.c \rangle \triangleright c[t_\ominus/x^\ominus]$$

$$\langle \mu\{\alpha^+\}.c \parallel \{e_+\} \rangle \triangleright c[e_+/\alpha^+]$$

$$e_+ \triangleright \tilde{\mu}\{x^\ominus\}.\langle \{x^\ominus\} \parallel e_+ \rangle$$

$$t_\ominus \triangleright \mu\{\alpha^+\}.\langle t_\ominus \parallel \{\alpha^+\} \rangle$$

The syntactic duploid

A term t is thunkable if and only if either:

1. for all c, e, q, q' , $\langle \mu q'. \langle t \parallel \tilde{\mu} q. c \rangle \parallel e \rangle \simeq_{\text{RE}_p} \langle t \parallel \tilde{\mu} q. \langle \mu q'. c \parallel e \rangle \rangle$
where q denotes an arbitrary pattern-matching
($\tilde{\mu}x, \mu\alpha, \tilde{\mu}\{x\}, \mu\{\alpha\} \dots$);
2. for all c, x , $\langle t \parallel \tilde{\mu}x. c \rangle \simeq_{\text{RE}_p} c[t/x]$.

Symmetrically, a context e is linear if and only if either:

1. for all c, t, q, q' , $\langle t \parallel \tilde{\mu} q'. \langle \mu q. c \parallel e \rangle \rangle \simeq_{\text{RE}_p} \langle \mu q. \langle t \parallel \tilde{\mu} q'. c \rangle \parallel e \rangle$; or
2. for all c, α , $\langle \mu\alpha. c \parallel e \rangle \simeq_{\text{RE}_p} c[e/\alpha]$.

We can easily prove many properties of linear contexts and thunkable terms thanks to having both a global and a local characterisation.

Führmann's result

Dualised

Carsten Führmann. *Direct Models for the Computational Lambda Calculus*. *Electr. Notes Theor. Comput. Sci.*, 20:245–292, 1999

Runnable monads implement call-by-value in call-by-name.

Definition

(T, η, ρ) runnable monad on a category \mathcal{C} : $T : \mathcal{C} \rightarrow \mathcal{C}$ functor,
 $\eta : 1 \rightarrow T$ natural transformation, $\rho : T \rightarrow 1$ transformation such
that $\rho_T : T^2 \rightarrow T$ is natural, such that $\rho \circ \eta = \text{id}$; $\rho_T \circ T\eta = \text{id}_T$ and
 $\rho \circ T\rho = \rho \circ \rho_T$. ((T, η, ρ_T) is a monad)

Syntactic idea λ calculus + a constant η + a term constructor $-^*$.
 t^* evaluates its argument until the latter is of the form ηu .
Then it continues with $t u$.

Führmann's result

Dualised

Comon: category of co-monads

RunMon: category of runnable monads

Theorem

Reflection $\mathbf{RunMon} \triangleleft \mathbf{Comon}$; in other words:

The Kleisli construction determines a functor $\mathbf{Comon} \rightarrow \mathbf{RunMon}$ that has a full and faithful right adjoint ($\mathbf{RunMon} \rightarrow \mathbf{Comon}$)

1. Every co-monad determines a runnable monad in the Kleisli
2. Every runnable monad arises in this way
3. The co-monad we retrieve from a runnable monad has special properties:
The set of stacks is completed into the set of all *linear* contexts
(+quotient of undistinguishable stacks)
4. ... and compositionally so.

Example

Ref! has a runnable monad (T, η, ρ) defined with:

- $TA \stackrel{\text{def}}{=} M_{\text{fin}}(A)$;
- $([m_1 + \dots + m_n], [a_1, \dots, a_n]) \in Tf$ whenever $n \in \mathbb{N}$ and $(m_i, a_i) \in f$ for all $i \leq n$;
- $\forall a \in A, ([[a]]), a) \in \rho_A$;
- $\forall m \in !A, (m, m) \in \text{wrap}_A$.

(Underlies Girard's boring translation:

$$A \rightarrow B = !(A \multimap B)$$

which is a model of commutative call-by-value.)

Example

continued

Definition

$$f \bullet g \stackrel{\text{def}}{=} \rho \circ T f \circ g$$

Proposition

$f \in \mathbf{Ref}_1(A, B)$ is linear:

$$\forall g, h \in E, (f \circ g) \bullet h = f \circ (g \bullet h)$$

if and only if:

$$\forall m \in M_{\text{fin}}(A), ((m, b) \in f \implies \#m = 1)$$