A Categorical Semantics for The Parallel Lambda-Calculus

Germain Faure
U. Henri Poincaré & LORIA
PROTHEO, Bât. B Campus scientifique
54506 Vandoeuvre-lès-Nancy, France
germain.faure@loria.fr

Alexandre Miquel
U. Paris 7 Denis Diderot
PPS, Case 7014, 2 Place Jussieu
75251 Paris Cedex 05, France
alexandre.miquel@pps.jussieu.fr

Abstract

In this paper, we define a sound and complete categorical semantics for the parallel λ-calculus, based on a notion of aggregation monad which is modular w.r.t. associativity, commutativity and idempotence.

To prove completeness, we introduce a category of partial equivalence relations adapted to parallelism, in which any extension of the basic equational theory of the calculus is induced by some model.

We also present abstract methods to construct models of the parallel λ-calculus in categories where particular equations have solutions, such as the category of Scott domains and its variants, and check that G. Boudol's original semantics is a particular case of ours.

1. Introduction

The categorical semantics of the untyped λ-calculus [4], which is based on Cartesian closed categories (ccc) with reflexive objects [8], provides a rich framework to classify all the existing models of the λ-calculus [4]. One of the most striking properties of this semantics is that it is complete in a very strong sense [12, 13]: for each extension $\mathscr{T}$ of the basic equational theory of the λ-calculus, it is possible to find a reflexive object (in a syntactical ccc of partial equivalence relations) such that the equational theory induced by the interpretation of pure λ-terms in this object is exactly the theory $\mathscr{T}$. Consequently, there exists a reflexive object which captures β-conversion, so that two terms having the same denotation in all reflexive objects have the same denotation in this particular object, and thus are β-convertible.

The aim of this paper is to extend the categorical semantics of the λ-calculus to the parallel λ-calculus (introduced in [5]), while keeping a similar property of completeness.

Formally, the parallel λ-calculus is obtained by extending the pure λ-calculus with a binary operator $M \mathbin{/\!\!/} N$, that intuitively represents the parallel execution of M and N. The parallel λ-calculus adds to the equational theory of the pure λ-calculus the single equation

$$(\delta) \qquad (M_1 \mathbin{/\!\!/} M_2)N \;=\; M_1N \mathbin{/\!\!/} M_2N$$

expressing the distributivity of function application w.r.t. parallel aggregation. Considering this equation as a reduction rule (orienting it from left to right), this rule means that the execution of λ-terms is never blocked by the parallel construction, but that it simply 'forks' each time such a construction comes in head position.

The parallel λ-calculus was initially introduced as a tool to study full-abstraction of the interpretation of λ-terms in Scott domains. In this framework, Boudol extended the interpretation of pure λ-terms to the parallel construction using the join operation.

However, the already existing models of the parallel λ-calculus appear to be too limited if we want to model further extensions of it—such as Cirstea and Kirchner's rewriting calculus [7]—or simply if we are interested in the study of the parallel λ-calculus *per se*. Scott semantics is well-suited to achieve full-abstraction; but it is not sufficient to capture neither the basic equational theory of the calculus nor many interesting extensions of it—typically with one of the equations (ϵ) and (η) that we will present in section 2. In the same way, interpreting the parallel operator as the binary join automatically validates associativity, commutativity and idempotence (ACI), although on a purely syntactical level, these equations are clearly independent from the basic equations (β) and (δ). For these reasons, there is a need for a more general and more modular semantics.

The primary motivation of this work was the study of the rewriting calculus (a.k.a. the ρ-calculus), an extension of the λ-calculus that combines pattern-matching with a notion of structure (written $M \wr N$) used to build collections of terms. However, the semantical study of the ρ-calculus quickly revealed the similarity between the notion of structure and Boudol's parallel construct—up to the fact that the notion of structure of the ρ-calculus is not systematically required to be associative, commutative and/or idempotent.

Actually, it seems clear to the authors that up the ACI equations, the ρ-calculus is nothing but the parallel λ-calculus extended with a mechanism of pattern-matching.

The semantical study of the ρ-calculus also pointed out many interesting problems related to the interaction between a mechanism of pattern-matching and the parallel construct. These problems (that we briefly discuss in the end of the paper) helped us to grasp the importance of additive terms (defined in subsection 2.3) which play a central rôle in the proof of completeness. (Boudol's shift towards the λ-calculus with resources [6] seems to be motivated by similar reasons.) We hope that the categorical semantics presented in this paper will contribute in future works to a better understanding of this interaction, and thus will constitute a significant step towards a denotational semantics for the ρ-calculus.

Outline of the paper

The paper is organized as follows.

In section 2 we recall the syntax and equations of the parallel λ-calculus.

In section 3, we present a notion of aggregation monad from which we define (in section 4) the categorical notion of model for the parallel λ-calculus.

To prove completeness, we define a category of pers (in section 5) and study its properties. Then we present in section 6 two abstract methods for building models.

We conclude by showing some problems related to the interaction of ML-style pattern-matching with the parallel construct (in section 7).

2. The parallel λ-calculus

2.1. The core calculus

The parallel λ-calculus is obtained by extending the pure λ-calculus with a binary operator $M \mathbin{/\!\!/} N$ representing the parallel execution of M and N. Formally, the terms of the parallel λ-calculus are given by

$$M, N \ ::= \ x \mid \lambda x \,.\, M \mid MN \mid M \mathbin{/\!\!/} N$$

and the corresponding equational theory is defined from the two equations

$$\begin{array}{lrcl} (\beta) & (\lambda x \,.\, M)N & = & M\{x := N\} \\ (\delta) & (M_1 \mathbin{/\!\!/} M_2)N & = & M_1N \mathbin{/\!\!/} M_2N \end{array}$$

Throughout the paper, we will consider parallel λ-terms from the point of view of equational reasoning rather than from the point of view of reduction. However, both equations can also be presented as rewrite rules (orienting them from left to right), and it can be checked that the rewrite systems induced by β, δ and $\beta\delta$ are confluent.

2.2. Extensions of the equational theory

In many situations, it is desirable to extend the core calculus with one of the following equations:

$$\begin{array}{lrcll} (\epsilon) & \lambda x \,.\, (M_1 \mathbin{/\!\!/} M_2) & = & \lambda x \,.\, M_1 \mathbin{/\!\!/} \lambda x \,.\, M_2 & \\ (\eta) & \lambda x \,.\, Mx & = & M & (\text{if } x \notin FV(M)) \end{array}$$

Again, both equations can be presented as reduction rules, orienting them from left to right. Notice that in the presence of equation δ, the equation η subsumes ϵ, that is: $\epsilon \subset \delta\eta$ (equationally). From the point of view of the corresponding rewrite systems, both reduction rules δ and η make a critical pair which is closed with the ϵ-reduction rule.

Finally, the parallel λ-calculus can be extended with any combination of the three equations expressing associativity, commutativity and idempotence of the parallel operator:

$$\begin{array}{lrcl} (A) & (M_1 \mathbin{/\!\!/} M_2) \mathbin{/\!\!/} M_3 & = & M_1 \mathbin{/\!\!/} (M_2 \mathbin{/\!\!/} M_3) \\ (C) & M_1 \mathbin{/\!\!/} M_2 & = & M_2 \mathbin{/\!\!/} M_1 \\ (I) & M \mathbin{/\!\!/} M & = & M \end{array}$$

In what follows, most definitions will be modularized in order to handle the 24 variants of the calculus obtained by combining each of the 3 basic calculi $\beta\delta$, $\beta\delta\epsilon$ and $\beta\delta\eta$ with the $2^3 = 8$ possible combinations of A, C, and I.

2.3. Additivity

Let $\mathscr{T}$ be an equational theory of the parallel λ-calculus which contains at least β and δ. We say that a term M is *additive* (w.r.t. its first argument) in the theory $\mathscr{T}$ when

$$M(N_1 \mathbin{/\!\!/} N_2) =_{\mathscr{T}} MN_1 \mathbin{/\!\!/} MN_2$$

for all terms N_1, N_2. By substitutivity, it is equivalent to say that

$$M(x \mathbin{/\!\!/} y) =_{\mathscr{T}} Mx \mathbin{/\!\!/} My \,,$$

where x and y are fresh variables.

When a term M is additive in the theory $\beta\delta$, we simply say that M is additive. Examples of additive terms are the identity term $\lambda x \,.\, x$ and more generally all the terms of the form $\lambda x \,.\, x\vec{N}$ where $x \notin FV(\vec{N})$ (i.e. tuples).

It is important not to confound the notion of additivity with the more global notion of linear λ-term. (Remember that a λ-term is said to be linear when all its free and bound variables occur exactly once.) In particular, there is no inclusion between both notions:

- The term $\lambda x \,.\, x(\lambda y \,.\, yy)$ is additive but not linear,
- The term $\lambda xy \,.\, yx$ is linear but not additive.

3. Aggregation monads

We now present a notion of *aggregation monad* which is the categorical counterpart of the syntactical notion of parallel execution. We use here the terminology of 'aggregation' to emphasize the fact that this notion exists independently from the properties of associativity, commutativity and idempotence that are usually associated with the idea of parallelism. (However, we keep the name of parallel λ-calculus, for obvious historical reasons.)

3.1. Notion of aggregation

Let $\mathbf{T}$ be a monad [8] on a Cartesian category $\mathfrak{C}$ (with unit η and multiplication μ). A *notion of aggregation* for the monad $\mathbf{T}$ is a natural transformation

$$u_A : \mathbf{T}A \times \mathbf{T}A \to \mathbf{T}A$$

such that the following diagram commutes

$$\begin{array}{ccc} \mathbf{T}^2A \times \mathbf{T}^2A & \xrightarrow{u_{\mathbf{T}A}} & \mathbf{T}^2A \\ {\scriptstyle \mu_A \times \mu_A}\downarrow & & \downarrow{\scriptstyle \mu_A} \\ \mathbf{T}A \times \mathbf{T}A & \xrightarrow[u_A]{} & \mathbf{T}A \end{array} \qquad (1)$$

for all objects A. (As usual, we will frequently drop the subscript and write u for u_A.) An *aggregation monad* is simply a monad equipped with a notion of aggregation.

A consequence of diagram (1) which appears to be very useful in proofs is that the arrow u_A can always be defined in terms of the arrow $u_{\mathbf{T}A}$ via the commutative diagram

$$\mathbf{T}A \times \mathbf{T}A \xrightarrow[\eta_A \times \eta_A]{} \mathbf{T}^2A \times \mathbf{T}^2A \xrightarrow[u_{\mathbf{T}A}]{} \mathbf{T}^2A \xrightarrow[\mu_A]{} \mathbf{T}A, \quad \mathbf{T}A \times \mathbf{T}A \xrightarrow{u_A} \mathbf{T}A$$

Finally, we say that a notion of aggregation u is *associative* (A), *commutative* (C) or *idempotent* (I) depending on the corresponding diagram commutes in Fig. 1 (where α denotes the associativity isomorphism of $\times$).

Typical notions of aggregation monads are the following:

In the category of sets:

- The powerset monad with union (ACI)
- The multiset monad with multi-union (AC)
- The list monad with concatenation (A)
- The free group monad with composition (A)

In the category of Scott domains:

- The lower powerdomain monad with join (ACI)
- The upper powerdomain monad with meet (ACI)

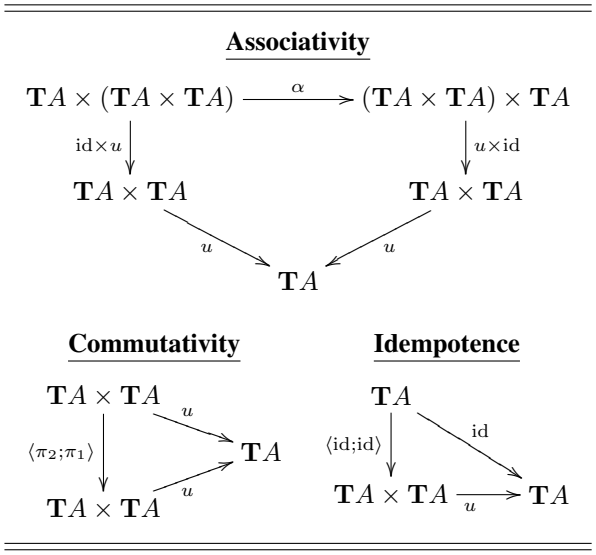


Figure 1. ACI diagrams of aggregation

In Ab-categories [8] (with finite products), the fundamental aggregation monad is the binary sum, given as the identity monad equipped with the arrow

$$\pi_1 + \pi_2 \ : \ A \times A \to A \qquad \text{(ACI)}$$

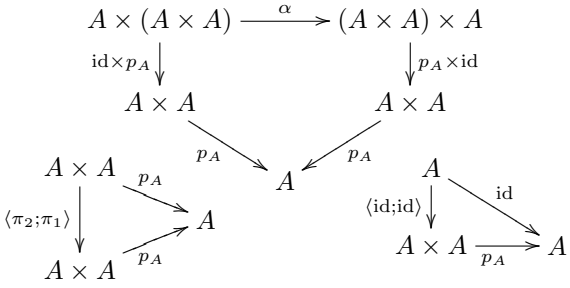
3.2. Algebras and additive morphisms

Let $\mathfrak{C}$ be a Cartesian category equipped with an aggregation monad $\langle \mathbf{T}, u \rangle$. Each $\mathbf{T}$-algebra $\langle A, h_A \rangle$ can be given an *aggregation operator* $p_A : A \times A \to A$ defined by

$$p_A \ = \ h_A \circ u_A \circ (\eta_A \times \eta_A) \, .$$

Of course, the aggregation operator p_A inherits the properties of associativity, commutativity and idempotence from the underlying aggregation monad $\langle \mathbf{T}, u \rangle$:

Lemma 1 — *If the aggregation monad $\langle \mathbf{T}, u \rangle$ is associative, commutative, and/or idempotent, then for all $\mathbf{T}$-algebras $\langle A, h_A \rangle$ the aggregation operator $p_A : A \times A \to A$ is associative, commutative, and/or idempotent in the sense of the diagrams:*



Proof. The property is straightforward for idempotence and commutativity. The diagram for associativity deeply relies on the characterization of u_A in terms of $u_{\mathbf{T}A}$ stated in subsection 3.1. □

A consequence of diagram (1) is that morphisms of algebras are additive in the sense that they commute with the aggregation operator:

Lemma 2 — *If $\langle A, h_A\rangle$ and $\langle B, h_B\rangle$ are $\mathbf{T}$-algebras, then for all morphisms of $\mathbf{T}$-algebras $f : \langle A, h_A\rangle \to \langle B, h_B\rangle$ the following diagram commutes:*

$$\begin{array}{ccc} A \times A & \xrightarrow{p_A} & A \\ {\scriptstyle f\times f}\downarrow & & \downarrow{\scriptstyle f} \\ B \times B & \xrightarrow[p_B]{} & B \end{array}$$

Remember that for all objects A, B and for all morphisms $f : A \to B$, the morphism $\mathbf{T}f$ is a morphism of algebras from $\langle \mathbf{T}A, \mu_A\rangle$ to $\langle \mathbf{T}B, \mu_B\rangle$.

3.3. Strong notion of aggregation

Let $\mathfrak{C}$ be a Cartesian category. A *strong monad* of $\mathfrak{C}$ [10] is a monad $\mathbf{T}$ of $\mathfrak{C}$ equipped with a binatural transformation

$$t_{A,B} : \mathbf{T}A \times B \to \mathbf{T}(A \times B)$$

such that the 3 diagrams of Fig. 2 commute, where α denotes the associativity isomorphism of $\times$ and r the right unit isomorphism. (Intuitively, the transformation t distributes the second component of its input to all the elements of the first component—thinking of $\mathbf{T}A$ as a type of lists or sets.)

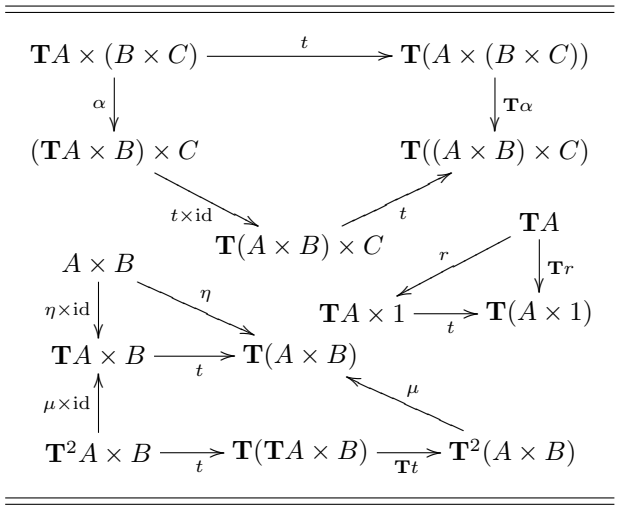


Figure 2. Diagrams of strong monadicity

In a strong monad $\mathbf{T}$, we say that a notion of aggregation u is *strong* when the following diagram commutes:

$$\begin{array}{ccc} (\mathbf{T}A \times \mathbf{T}A) \times B & \xrightarrow{u\times\mathrm{id}} & \mathbf{T}A \times B \\ {\scriptstyle \langle \pi_1\times\mathrm{id};\, \pi_2\times\mathrm{id}\rangle}\downarrow & & \\ (\mathbf{T}A \times B) \times (\mathbf{T}A \times B) & & \downarrow{\scriptstyle t} \\ {\scriptstyle t\times t}\downarrow & & \\ \mathbf{T}(A \times B) \times \mathbf{T}(A \times B) & \xrightarrow[u]{} & \mathbf{T}(A \times B) \end{array} \qquad (2)$$

A *strong aggregation monad* is simply a strong monad equipped with a strong notion of aggregation. In particular, all the examples of aggregation monads given in subsection 3.1 are strong aggregation monads.

4. Models of the parallel λ-calculus

4.1. Definition

A *model of the parallel λ-calculus* is a ccc $\mathfrak{C}$ equipped with a strong aggregation monad $\langle \mathbf{T}, u\rangle$ and a quadruple $\langle D, \mathsf{lam}, \mathsf{app}, \mathsf{flat}\rangle$ such that

- $\langle D, \mathsf{lam}, \mathsf{app}\rangle$ is a reflexive object of $\mathfrak{C}$,
- $\langle D, \mathsf{flat}\rangle$ is a $\mathbf{T}$-algebra;

and such that the following diagram commutes:

$$\begin{array}{ccc} \mathbf{T}D & \xrightarrow{\mathsf{flat}} & D \\ {\scriptstyle \mathbf{T}\mathsf{app}}\downarrow & & \downarrow{\scriptstyle \mathsf{app}} \\ \mathbf{T}(D^D) & \xrightarrow[\Lambda(\mathsf{flat}\circ\mathbf{T}\mathrm{ev}\circ t)]{} & D^D \end{array} \qquad (\delta)$$

where the bottom arrow is built by curryfying the following sequence of morphisms:

$$\mathbf{T}(D^D)\times D \xrightarrow[t]{} \mathbf{T}(D^D \times D) \xrightarrow[\mathbf{T}\mathrm{ev}]{} \mathbf{T}(D) \xrightarrow[\mathsf{flat}]{} D$$

Moreover, we say that such a model is:

- an *ϵ-model* when the following diagram commutes:

$$\begin{array}{ccc} \mathbf{T}D & \xrightarrow{\mathsf{flat}} & D \\ {\scriptstyle \mathbf{T}\mathsf{lam}}\uparrow & & \uparrow{\scriptstyle \mathsf{lam}} \\ \mathbf{T}(D^D) & \xrightarrow[\Lambda(\mathsf{flat}\circ\mathbf{T}\mathrm{ev}\circ t)]{} & D^D \end{array} \qquad (\epsilon)$$

- an *η-model* when $\mathsf{lam}\circ\mathsf{app} = \mathrm{id}$ (η), that is, when the arrows app and lam are converse isomorphisms;
- associative, commutative or idempotent when the underlying aggregation monad $\langle \mathbf{T}, u\rangle$ is.

It is straightforward to check that any η-model of the parallel λ-calculus is also an ϵ-model.

Finally, each model of the parallel λ-calculus comes with a *parallel operator* $\mathsf{par} : D \times D \to D$ given by

$$\mathsf{par} \;=\; p_D \;=\; \mathsf{flat} \circ u_D \circ (\eta_D \times \eta_D)\,.$$

4.2. Interpreting parallel λ-terms

Let $\mathfrak{C}$ be a ccc with a strong aggregation monad $\langle T, u\rangle$, and $\langle D, \mathsf{lam}, \mathsf{app}, \mathsf{flat}\rangle$ a model of the parallel λ-calculus in this category. Parallel λ-terms are interpreted in the model in the same way as pure λ-terms in any reflexive object.

Formally, the interpretation is parameterized by a list of variables (notation: ℓ, ℓ', etc.) which can be seen as a degenerate form of context where all variables are declared with type D. Given a list of variables ℓ, we write D^ℓ the object defined by $D^{[]} = 1$ and $D^{\ell,x} = D^\ell \times D$.

To each pair (ℓ, x) formed by a list of variables ℓ and a variable x that belongs to ℓ, we define the *projection* $\pi^x_\ell : D^\ell \to D$ by setting

$$\begin{array}{lcll} \pi^x_{\ell,x} & = & \pi_2 & \in \quad D^\ell \times D \to D \\ \pi^x_{\ell,y} & = & \pi^x_\ell \circ \pi_1 & \in \quad D^\ell \times D \to D \qquad (\text{if } y \neq x) \end{array}$$

Each parallel λ-term M whose free variables belong to a list ℓ is interpreted as an arrow $[\![M]\!]_\ell : D^\ell \to D$ given by

$$\begin{array}{lrcl} (\textsc{Var}) & [\![x]\!]_\ell & = & \pi^x_\ell \\ (\textsc{Lam}) & [\![\lambda x\,.\,M]\!]_\ell & = & \mathsf{lam} \circ \Lambda([\![M]\!]_{\ell,x}) \\ (\textsc{App}) & [\![MN]\!]_\ell & = & \mathrm{ev} \circ \langle \mathsf{app} \circ [\![M]\!]_\ell ; [\![N]\!]_\ell \rangle \\ (\textsc{Par}) & [\![M \mathbin{/\!/} N]\!]_\ell & = & \mathsf{par} \circ \langle [\![M]\!]_\ell ; [\![N]\!]_\ell \rangle \end{array}$$

Soundness of this interpretation relies on the lemma

Lemma 3 (Substitutivity) — *Given a list of variables ℓ and a variable x, then for all terms M and N such that $FV(M) \subset (\ell, x)$ and $FV(N) \subset \ell$, we have:*

$$[\![M\{x := N\}]\!]_\ell \quad = \quad [\![M]\!]_{\ell,x} \circ \langle \mathrm{id}; [\![N]\!]_\ell \rangle$$

Proof. By induction on M. □

Let $\mathscr{T}$ denote one of the 24 equational theories obtained by combining the three basic theories $\beta\delta$, $\beta\delta\epsilon$, $\beta\delta\eta$ with all possible combinations of A, C and I. We say that the model D is *adapted* to the theory $\mathscr{T}$ when

- if $\mathscr{T}$ contains the equation ϵ (resp. η), then D is an ϵ-model (resp. an η-model);
- if $\mathscr{T}$ contains the equation A (resp. C, I), then the underlying aggregation monad $\langle \mathbf{T}, u\rangle$ is associative (resp. commutative, idempotent).

Proposition 4 (Soundness) — *If the model is adapted to the theory $\mathscr{T}$, then for all lists of variables ℓ and for all terms M, M' whose free variables occur in ℓ, we have:*

$$M =_{\mathscr{T}} M' \quad \Rightarrow \quad [\![M]\!]_\ell = [\![M']\!]_\ell\,.$$

Proof. It suffices to check the equality for each equation of $\mathscr{T}$. The soundness of equation (δ) reduces to the corresponding diagram (δ) using the fact that the aggregation monad $\mathbf{T}$ is strong (diagram (2)). □

4.3. Examples in Scott domains

In the category of Scott domains [**?**, 3], the ACI-aggregation monad $\langle \mathcal{P}_l; \vee\rangle$ (where $\mathcal{P}_l$ denotes the lower powerdomain [11, 14]) is the source of a plethora of models for the parallel λ-calculus, due to the fact that:

Proposition 5 — *Any Scott domain D with a top element is a $\mathcal{P}_l$-algebra whose aggregation operator p_D is the binary join: $p_D(x, y) = x \vee y$ (for all $x, y \in D$).*

Proof. Remember that a Scott domain has a top element iff it has all its joins (the converse already holds for cpos). We define the morphism $h_D : \mathcal{P}_l(D) \to D$ from the map

$$\begin{array}{rcccl} h_0 & : & \mathcal{K}(\mathcal{P}_l(D)) = \mathfrak{P}^+_{\mathrm{fin}}(\mathcal{K}(D)) & \to & D \\ & & \{k_1; \ldots; k_n\} & \mapsto & k_1 \vee \cdots \vee k_n \end{array}$$

(writing $\mathcal{K}(D)$ the set of finite elements of D) by setting

$$h_D(x) = \sup_{k \ll x} h_0(k)$$

(where k ranges over all finite approximants of x) for all $x \in \mathcal{P}_l(D)$. From this construction, it is obvious that the corresponding aggregation operator is the binary join. □

Moreover, the notion of morphism of algebras (cf subsection 3.2) exactly corresponds to the notion of additive functions in Scott domains:

Proposition 6 — *Let D and E be two Scott domains with a top element. A continuous function $f : D \to E$ is a morphism of algebras iff it is additive, namely:*

$$f(\bot) = \bot \qquad \textit{and} \qquad f(x \vee y) = f(x) \vee f(y)$$

for all $x, y \in D$.

(Notice that we require that additive functions are strict.)

Proposition 7 — *If D is a Scott domain with a top element equipped with a reflexive structure $(\mathsf{lam}, \mathsf{app})$ where $\mathsf{app} : D \to D^D$ is an additive continuous function, then D is a model of the parallel λ-calculus w.r.t. the aggregation monad $\langle \mathcal{P}_l; \vee\rangle$.*

Proof. To check that the diagram (δ) commutes, it suffices to check that for all $\bar{k} = \{k_1; \ldots; k_n\} \in \mathcal{K}(\mathcal{P}_l(D))$ and for all $x \in D$ we have

$$\begin{array}{l}\mathsf{app}(k_1 \vee \cdots \vee k_n)(x) \;= \\ \quad \mathsf{app}(k_1)(x) \vee \cdots \vee \mathsf{app}(k_n)(x)\,,\end{array}$$

which follows from the hypothesis and the fact that function application is additive on its first argument. □

An obvious example of such a model is Scott's D_∞ domain, which is built from the domain $D_0 = \{\top\!\!\top_\perp\}$ by taking the colimit of the sequence $D_{i+1} = (D_i \to D_i)$ $(i \geq 0)$.

Application: Boudol's models In [5], Boudol presents two models D_* and D_s for λ-calculi with a parallel construct, as the initial solutions of both equations

$$D_* = (D_* \to D_*)_\perp \quad \text{and} \quad D_\mathsf{s} = (D_\mathsf{s} \to_\perp D_\mathsf{s})_\perp$$

(where $(\_)_\perp$ denotes lifting and $(\_ \to_\perp \_)$ the space of strict continuous functions). The first model D_* (due to Abramsky [1, 2]) is clearly a model of the parallel λ-calculus from Prop. 7 due to the existence of a retraction pair

$$(\mathrm{up}_*, \mathrm{down}_*) \;:\; (D_* \to D_*) \lhd D_*$$

whose second component (the projection) is additive. The second model D_s (which interprets a λ-calculus with call-by-value abstractions) can be decomposed as follows

$$\left\{\begin{array}{rcl} D_\mathsf{s} & = & V_{\mathsf{s}\perp} \\ V_\mathsf{s} & = & V_{\mathsf{s}\perp} \to_\perp D_\mathsf{s} \;=\; V_\mathsf{s} \to D_\mathsf{s} \end{array}\right.$$

where V_s is a space of values (as opposed to D_s, which is a space of computations). Here, the space of values V_s is again a model of the parallel λ-calculus from Prop. 7 due to the existence of a retraction pair

$$(\lambda f\,.\,\mathrm{up}_\mathsf{s} \circ f,\; \lambda f\,.\,\mathrm{down}_\mathsf{s} \circ f) \;:\; (V_\mathsf{s} \to V_\mathsf{s}) \lhd V_\mathsf{s}$$

whose second component is additive. (Here, $(\mathrm{up}_\mathsf{s}, \mathrm{down}_\mathsf{s})$ denotes the retraction $V_\mathsf{s} \lhd D_\mathsf{s}$.)

5. The PER-model

5.1. The notion of $\mathscr{T}$-per

Let $\mathscr{T}$ be one of the 24 equational theories of the parallel λ-calculus mentioned in section 2.

Definition 1 ($\mathscr{T}$-per) — *A $\mathscr{T}$-*partial equivalence relation *($\mathscr{T}$-*per*) is a partial equivalence relation (per) A on the set of parallel λ-terms such that $\mathscr{T} \circ A \subset A$, that is, a symmetric and transitive relation A such that*

$$(M, M') \in A \;\wedge\; M' =_\mathscr{T} M'' \;\Rightarrow\; (M, M'') \in A$$

for all terms M, M', M''.

Given a $\mathscr{T}$-per A, we call the *domain of* A the set

$$\mathrm{dom}(A) = \{M \mid (M, M) \in A\}\,.$$

$\mathscr{T}$-pers are naturally ordered by inclusion: the smallest $\mathscr{T}$-per is the empty per (of domain the empty set) and the largest $\mathscr{T}$-per is the full per (of domain the set of all terms). Moreover, $\mathscr{T}$-pers are closed under arbitrary intersection, and thus form a complete distributive lattice.

Two important constructions of $\mathscr{T}$-pers are:

- The *arrow* $A \to B$ of two $\mathscr{T}$-pers A and B, which is defined for all M, M' by

  $$\begin{array}{l}(M, M') \in (A \to B) \quad \text{iff} \\ \quad \forall N, N'\; ((N, N') \in A \Rightarrow (MN, M'N') \in B)\end{array}$$

  This construction is antimonotonic w.r.t. A and monotonic w.r.t. B.

- The *parallel closure* A^+ of a $\mathscr{T}$-per A, which is inductively defined by the two rules:

  $$\frac{(M, M') \in A}{(M, M') \in A^+} \qquad \frac{\begin{array}{c}(M_1, M_1') \in A^+ \quad M =_\mathscr{T} M_1 \mathbin{/\!\!/} M_2 \\ (M_2, M_2') \in A^+ \quad M' =_\mathscr{T} M_1' \mathbin{/\!\!/} M_2'\end{array}}{(M, M') \in A^+}$$

  This construction is a closure operator, in the sense that it is monotonic and fulfills $A \subset A^+$ and $A^{++} = A^+$.

Let us finally notice a few obvious but useful facts.

Fact 8 — *For all $\mathscr{T}$-pers A, B:* $(A \to B)^+ \subset A \to B^+$.

On the other hand, the inclusion $A \to B \subset A^+ \to B^+$ does not hold in general. However, we still have:

Fact 9 — *If two terms M_1 and M_2 are additive in the theory $\mathscr{T}$ (cf subsection 2.3), then $(M_1, M_2) \in (A \to B)$ implies $(M_1, M_2) \in (A^+ \to B^+)$.*

5.2. The ccc structure of $\mathscr{T}$-PER

Let $\mathscr{T}$-**PER** be the category whose objects are $\mathscr{T}$-pers and whose hom-sets are given by

$$\mathscr{T}\text{-}\mathbf{PER}[A; B] \;=\; \mathrm{dom}(A \to B)/{\sim_{(A \to B)}}\,,$$

where $\sim_{(A \to B)}$ denotes the (total) equivalence relation induced by the $\mathscr{T}$-per $(A \to B)$ on its domain.

The category $\mathscr{T}$-**PER** has the structure of a ccc:

- The terminal object is the full $\mathscr{T}$-per: $1 = \top$.
- The Cartesian product $A \times B$ is defined by

  $$\begin{array}{l}(M, M') \in (A \times B) \text{ iff} \\ \quad (\pi_1 M, \pi_1 M') \in A \text{ and } (\pi_2 M, \pi_2 M') \in B\,,\end{array}$$

  where $\pi_1 = \lambda p\,.\,p(\lambda xy\,.\,x)$, $\pi_2 = \lambda p\,.\,p(\lambda xy\,.\,y)$ and $\langle M_1; M_2\rangle = \lambda xp\,.\,p(M_1 x)(M_2 x)$.

- The exponent is given by $B^A = (A \to B)$, the evaluation arrow by $\mathrm{ev} = \lambda p\,.\,\pi_1 p\,(\pi_2 p)$, and the curryfied arrow by $\Lambda(M) = \lambda xy\,.\,M(\lambda p\,.\,pxy)$.

5.3. The aggregation monad of $\mathscr{T}$-PER

It would be tempting to define the aggregation monad $\mathbf{T}$ of $\mathscr{T}$-**PER** by setting $\mathbf{T}A = A^+$. Unfortunately, the parallel closure operator $A \mapsto A^+$ is not functorial (since $A \subset A^+$ but $(A \to B) \not\subseteq (A^+ \to B^+)$) and thus cannot be given the structure of a monad.

To achieve functoriality, we first need to introduce the following boxing mechanism:

The boxing monad For all M we set $[M] = \lambda x\,.\,xM$ (this construction can be understood as a 1uple). Unboxing is performed by applying $\mathbf{I} = \lambda x\,.\,x$, since $[M]\mathbf{I} =_\beta M$.

To each $\mathscr{T}$-per A we associate a $\mathscr{T}$-per $[A]$ defined by

$$(M, M') \in [A] \quad \text{iff} \quad \exists (M_0, M_0') \in A \;\; (M =_{\mathscr{T}} [M_0] \wedge M' =_{\mathscr{T}} [M_0'])\,.$$

Notice that both $\mathscr{T}$-pers $[A]$ and A are isomorphic via the converse isomorphisms:

$$\begin{array}{rcccl} \mathsf{box} & = & \lambda x\,.\,[x] & \in & \mathrm{dom}(A \to [A]) \\ \text{and} \quad \mathsf{unbox} & = & \lambda x\,.\,x\mathbf{I} & \in & \mathrm{dom}([A] \to A)\,. \end{array}$$

Moreover, we easily check that

$$[A \to B] \subset [A] \to B$$

for all $\mathscr{T}$-pers A and B.

The correspondence $A \mapsto [A]$ is turned into a strong monad as follows. First we make this correspondence functorial by setting $\uparrow M = [\lambda z\,.\,[Mz]]$ for all M, and by checking that $M \in \mathrm{dom}(A \to B)$ implies

$$\uparrow M \;\in\; \mathrm{dom}([A \to [B]]) \;\subset\; \mathrm{dom}([A] \to [B])\,.$$

Then we take $\eta = \mathsf{box}$ and $\mu = \mathsf{unbox}$, and set

$$t = \lambda x\,.\,\pi_1 x(\lambda y\,.\,[\langle y, \pi_2 x\rangle])\,.$$

The main property of the boxing monad is that boxed objects (including lifted arrows $\uparrow M$) are additive:

Fact 10 — *If a term $M \in \mathrm{dom}([A])$, then M is additive in the theory $\mathscr{T}$ (cf subsection 2.3).*

This property is crucial for the definition below.

The aggregation monad We can now define our aggregation monad $\mathbf{T}$ by setting

$$\mathbf{T}A \;=\; [A]^+$$

for all $\mathscr{T}$-pers A. The functorial map $\uparrow M$ and the natural transformations η, μ and t are defined the same way as for the boxing monad $[\_]$. Of course, one has to check that these constructions fit in their new types, which easily follows from the properties of additivity (Fact 10).

Finally, we set

$$u = \lambda p\,.\,(\pi_1 p \mathbin{/\!\!/} \pi_2 p)$$

and check that:

Proposition 11 — *$\langle \mathbf{T}, u\rangle$ is a strong aggregation monad on the category $\mathscr{T}$-**PER**.*

Moreover, it is straightforward to check that the aggregation monad $\langle \mathbf{T}, u\rangle$ is associative, commutative or idempotent as soon as $\mathscr{T}$ contains the corresponding equation.

5.4. Completeness

Every model D of the parallel λ-calculus induces a congruence written $=_D$ over the set of parallel λ-terms, which is defined for all terms M and M' by

$$M =_D M' \quad \text{iff} \qquad [\![M]\!]^D_\ell = [\![M']\!]^D_\ell$$

(where ℓ is such that $FV(MM') \subset \ell$). Of course, the congruence $=_D$ contains $\beta\delta$.

We now want to show that the converse holds, in the sense that for every congruence $\mathscr{T}$ containing $\beta\delta$, there exists a model D of the parallel λ-calculus that induces the congruence $\mathscr{T}$ exactly, namely, a model D such that $M =_D M'$ iff $M =_{\mathscr{T}} M'$ for all M, M'.

Theorem 1 (Completeness) — *Let $\mathscr{T}_0$ be one of the 24 equational theories of the parallel λ-calculus mentioned in section 2. For every congruence $\mathscr{T} \supseteq \mathscr{T}_0$, there exists a $\mathscr{T}_0$-per D such that:*

1. *D is a model of the parallel λ-calculus in $\mathscr{T}_0$-**PER**, which is adapted to the theory $\mathscr{T}_0$;*
2. *$M =_D M'$ iff $M =_{\mathscr{T}} M'$ (for all terms M, M')*

The theorem is proved as follows: consider a congruence $\mathscr{T} \supseteq \mathscr{T}_0 (\supseteq \beta\delta)$. Noticing that the congruence $\mathscr{T}$ is a $\mathscr{T}_0$-per whose domain is the set of all terms, we check that:

Proposition 12 — *The $\mathscr{T}_0$-per $\mathscr{T}$ can be equipped with all the structures of a model D of the parallel λ-calculus in the category $\mathscr{T}_0$-**PER**.*

Proof. We set $D = \mathscr{T}$, $\mathsf{lam} = \lambda xy\,.\,xy$, $\mathsf{app} = \lambda x\,.\,x$ (for the structure of reflexive object) and $\mathsf{flat} = \mathsf{unbox}$ (for the structure of algebra), and we check that the diagram (δ) of subsection 4.1 commutes. □

We easily check that the model D defined above is an ϵ-model (resp. an η-model) as soon as the equational theory $\mathscr{T}_0$ (and, actually $\mathscr{T}$) contains the equation ϵ (resp. the equation η). Moreover, we know that the underlying aggregation monad that comes with the category $\mathscr{T}_0$-$\mathbf{PER}$ is associative, commutative, idempotent as soon as $\mathscr{T}_0$ contains the corresponding equation, hence:

Proposition 13 — *The model D defined from the $\mathscr{T}_0$-per $\mathscr{T}$ is adapted to the theory $\mathscr{T}_0$.*

To conclude the proof of Theorem 1, we need to ensure that the congruence induced by the model D is precisely the congruence $\mathscr{T}$. This relies on the following lemma:

Lemma 14 — *Let $\ell = [x_1, \ldots, x_n]$ be a list of variables. For all terms M such that $FV(M) \subset \ell$, we have:*

$$[\![M]\!]^D_\ell \;=_{\beta\delta}\; \lambda z\,.\,M\{x_1 := \pi^{x_1}_\ell z; \ldots; x_n := \pi^{x_n}_\ell z\}$$

(where z is a fresh variable).

Proof. By induction on M. □

From this lemma, we easily conclude that $M =_D M'$ iff $M =_{\mathscr{T}} M'$, and the proof of Theorem 1 is done.

6. Building models

In this section, we present two methods to build a model of the parallel λ-calculus from a given ccc $\mathfrak{C}$ and a given strong aggregation monad $\langle T, u\rangle$. Both construction methods—which rely on the existence of objects satisfying particular equations—can be fruitfully used in the category of Scott domains (and its variants) where such equations have many interesting solutions.

6.1. First method

Theorem 2 — *If D is an object such that $(\mathbf{T}D)^D \simeq D$, then D can be given all the structures of a model of the parallel λ-calculus, model which is in general neither an ϵ-model nor an η-model.*

Proof. Consider an object D equipped with converse isomorphisms

$$(\mathbf{T}D)^D \underset{\mathsf{unfold}}{\overset{\mathsf{fold}}{\rightleftarrows}} D$$

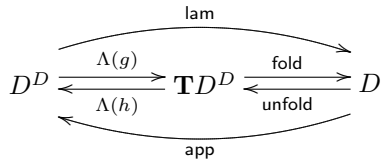
First, we equip D with a structure of $\mathbf{T}$-algebra by introducing the arrow $\mathsf{flat} : \mathbf{T}D \to D$ defined by

$$\mathbf{T}D \xrightarrow{\mathbf{T}\mathsf{unfold}} \mathbf{T}((\mathbf{T}D)^D) \xrightarrow{\Lambda(f)} (\mathbf{T}D)^D \xrightarrow{\mathsf{fold}} D$$

where $f : \mathbf{T}((\mathbf{T}D)^D) \times D \to \mathbf{T}D$ is given by

$$\mathbf{T}((\mathbf{T}D)^D)\times D \xrightarrow{t} \mathbf{T}((\mathbf{T}D)^D \times D) \xrightarrow{\mathbf{T}\mathrm{ev}} \mathbf{T}^2 D \xrightarrow{\mu} \mathbf{T}D$$

Then we define the reflexive structure by setting



where $g : D^D \times D \to \mathbf{T}D$ and $h : (\mathbf{T}D)^D \times D \to D$ are defined by

$$\begin{aligned} g &: D^D \times D \xrightarrow{\mathrm{ev}} D \xrightarrow{\eta} \mathbf{T}D \\ h &: (\mathbf{T}D)^D \times D \xrightarrow{\mathrm{ev}} \mathbf{T}D \xrightarrow{\mathsf{flat}} D \end{aligned}$$

We then check that $\mathsf{app} \circ \mathsf{lam} = \mathrm{id}$ (which comes from the fact that $\Lambda(h) \circ \Lambda(g) = \mathrm{id}$) and that diagram (δ) holds. □

The typical use of this theorem in the category of Scott domains is the following: assume that $\langle \mathbf{T}, u\rangle$ is a strong aggregation monad in the category of Scott domains whose underlying endofunctor $\mathbf{T}$ is ω^{op}-continuous (i.e. preserves limits on ω^{op}-chains). Then the correspondence

$$X \mapsto (\mathbf{T}X)^X$$

induces an ω-cocontinuous (covariant) endofunctor in the category $\mathbf{Scott}^{ip}$ of Scott domains equipped with injection-projection pairs. Starting from a domain D_0 equipped with an injection retraction pair $D_0 \rightarrowtail (\mathbf{T}D_0)^{D_0}$, it is easy to build a smallest fixpoint $D \simeq (\mathbf{T}D)^D$ containing D_0 (in the sense of injection-retraction pairs).

Notice that this way of constructing models in the category of Scott domains is not limited to the lower powerdomain monad $\mathcal{P}_l$, but that it can be also used with:

- The list monad, which defines an associative aggregation monad using the concatenation function;
- The free magma monad $\mathbf{T}X$, defined as the smallest fixpoint of the equation

  $$\mathbf{T}X \;=\; \big(X + (\mathbf{T}X \times \mathbf{T}X)\big)_\perp\,,$$

  that induces an aggregation monad which is neither associative, commutative nor idempotent.

6.2. Second method

The second construction method, which takes place in a ccc $\mathfrak{C}$ with denumerable Cartesian products, is inspired by a standard method to build models of the $\lambda\mu$-calculus in categories of domains.

Theorem 3 — *Let $\langle R, h_R\rangle$ be a $\mathbf{T}$-algebra. If D is an object such that $D \simeq R^{(D^\omega)}$ (where D^ω denotes a denumerable Cartesian product of D), then D can be given all the structures of an η-model of the parallel λ-calculus.*

Proof. Let D be an object equipped with converse isomorphisms

$$R^{(D^\omega)} \underset{\text{unfold}}{\overset{\text{fold}}{\rightleftarrows}} D$$

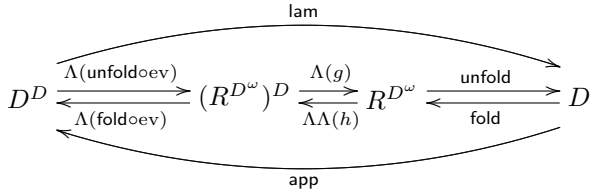
Consider the natural isomorphism $\mathsf{cons} : A \times A^\omega \xrightarrow{\sim} A^\omega$ and write $\mathsf{hd} = \pi_1 \circ \mathsf{cons}^{-1}$ and $\mathsf{tl} = \pi_2 \circ \mathsf{cons}^{-1}$. The algebra structure of D is defined by

$$\mathsf{flat} \;:\; \mathbf{T}D \xrightarrow{\mathbf{T}\mathsf{unfold}} \mathbf{T}(R^{D^\omega}) \xrightarrow{\Lambda(f)} R^{D^\omega} \xrightarrow{\quad\mathsf{fold}\quad} D$$

where $f : \mathbf{T}(R^{D^\omega}) \times D^\omega \to R$ is defined as the sequence

$$\mathbf{T}(R^{D^\omega}) \times D^\omega \xrightarrow{t} \mathbf{T}(R^{D^\omega} \times D^\omega) \xrightarrow{\mathbf{T}\mathrm{ev}} \mathbf{T}(R) \xrightarrow{h_R} R\,.$$

The reflexive structure $(\mathsf{lam}, \mathsf{app})$ is defined by



where both arrows g and h are defined by

$$\begin{aligned} g &= \mathrm{ev} \circ (\mathrm{ev} \times \mathrm{id}) \circ \alpha \circ (\mathrm{id} \times \mathsf{cons}^{-1}) \\ h &= \mathrm{ev} \circ (\mathrm{id} \times \mathsf{cons}) \circ \alpha \end{aligned}$$

We easily check that $\Lambda(\mathsf{unfold} \circ \mathrm{ev})$ and $\Lambda(\mathsf{fold} \circ \mathrm{ev})$ are converse isomorphisms, as well as $\Lambda(g)$ and $\Lambda\Lambda(h)$. We conclude by checking that the diagram (δ) of subsection 4.1 commutes, which is an exercise of diagram chasing. □

In Scott domains, the equation $D = R^{(D^\omega)}$ always has solutions, since the endofunctor $X \mapsto R^{(X^\omega)}$ is ω-cocontinuous in the category of injection-projection pairs. Notice that the least fixpoint D of this functor is not trivial as soon as the algebra R is not trivial. Intuitively, the smallest solution D can be understood as the smallest η-model of the parallel λ-calculus which contains R (in the sense of injection-projection pairs).

7. Future Work

As mentioned in the introduction, this work is initially motivated by the semantical study of the ρ-calculus [7], a formalism which combines ML-style pattern-matching with parallel aggregation. The next step is thus to find a satisfying way to integrate constructors and pattern-matching in our setting. However, combining pattern-matching with parallel aggregation naturally raises new problems related to additivity. To understand this point, let us consider the following example.

Assume that the parallel λ-calculus is enriched with two constant constructors a, b and a unary constructor $c(\_)$, plus a syntactic construct $[c(x) \ll N]M$ that matches the term N against the pattern $c(x)$, and binds all free occurrences of x in M to the argument of the destructed value. (We do not give any special meaning to this construction when N is not a constructed value.)

Now consider the term $M = [c(x) \ll c(a \mathbin{/\!\!/} b)]\, F\, x\, x$, where F is an arbitrary function. The naive way to reduce M is to substitute the term $(a \mathbin{/\!\!/} b)$ to x in the r.h.s. $F\, x\, x$, hence:

$$[c(x) \ll c(a \mathbin{/\!\!/} b)]\, F\, x\, x \;\;\to\;\; F\,(a \mathbin{/\!\!/} b)\,(a \mathbin{/\!\!/} b)\,.$$

(This strategy is the one which is actually implemented by the standard encodings of constructed values and pattern-matching in the λ-calculus.)

However, it is also legitimate to consider that a and b represent two possible choices for the argument of the constructed value $c(a \mathbin{/\!\!/} b)$. Following this intuition, a completely different reduction strategy is to distribute a and b w.r.t. the matching construct, which yields:

$$[c(x) \ll c(a \mathbin{/\!\!/} b)]F\, x\, x \;\;\to\;\; F\, a\, a \;\mathbin{/\!\!/}\; F\, b\, b\,.$$

Of course, both design choices are clearly incompatible, which is easy to see by taking $F = \lambda xy\,.\,xy$. This second strategy—which seems to be impossible to simulate in the core parallel λ-calculus—is much more interesting, since it suggests that both operations of construction and destruction are additive:

$$\begin{aligned} c(N_1 \mathbin{/\!\!/} N_2) &= c(N_1) \mathbin{/\!\!/} c(N_2) \\ [c(x) \ll (N_1 \mathbin{/\!\!/} N_2)]M &= [c(x) \ll N_1]M \mathbin{/\!\!/} [c(x) \ll N_2]M \end{aligned}$$

This example naturally raises the exciting challenge of constructing a model of the ρ-calculus that implements the second reduction strategy, while being rich enough to reflect all the expressivity of ML-style pattern-matching, such as the existence of infinitely many constructors of all arities (with pairwise disjoint images), the existence of variadic constructors, etc.

Acknowledgements

We would like to thank François Lamarche and Paul-André Melliès who helped us to find our way through Category theory.

References

[1] S. Abramsky. Domain theory in logical form. *Annals of Pure and Applied Logic*, 51(1–2):1–77, 1991.

[2] S. Abramsky and C.-H. L. Ong. Full abstraction in the lazy lambda calculus. *Information and Computation*, 105(2):159–267, 1993.

[3] R. M. Amadio and P.-L. Curien. *Domains and Lambda-Calculi*, volume 46 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 1998.

[4] H. P. Barendregt. *The Lambda-Calculus, its syntax and semantics*. Studies in Logic and the Foundation of Mathematics. North Holland, 1984. Second edition.

[5] G. Boudol. Lambda-calculi for (strict) parallel functions. *Information and Computation*, 108(1):51–127, 1994.

[6] G. Boudol, P.-L. Curien, and C. Lavatelli. A semantics for lambda calculi with resources. *Mathematical Structures in Computer Science*, 9(4):437–482, 1999.

[7] H. Cirstea and C. Kirchner. The rewriting calculus — Part I *and* II. *Logic Journal of the Interest Group in Pure and Applied Logics*, 9(3):427–498, 2001.

[8] S. Mac Lane. *Categories for the working mathematician*. Graduate Texts in Mathematics. Springer, New York / Berlin, 2nd. edition edition, 1998.

[9] E. Moggi. Computational lambda-calculus and monads. In *Proc. of Logic in Compute Science, LICS'89*, pages 14–23. IEEE Computer Society Press, Washington, DC, 1989.

[10] E. Moggi. Notions of computation and monads. *Information and Computation*, 93:55–92, 1991.

[11] G. D. Plotkin. A powerdomain construction. *SIAM Journal on Computing*, 5(3):452–487, 1976.

[12] D. S. Scott. Lambda calculus: Some models, some philosophy. In J. Barwise, H. J. Keisler, and K. Kunen, editors, *The Kleene Symposium*, pages 223–265. North Holland, Amsterdam, 1980.

[13] D. S. Scott. Relating theories of the λ-calculus. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 403–450. Academic Press, 1980.

[14] M. B. Smyth. Power domains. *Journal of Computer and System Sciences*, 16(1):23–36, 1978.