On the denotational semantics of the untyped lambda-mu calculus

Olivier LAURENT

Preuves Programmes Systèmes

CNRS – Université Paris VII

UMR 7126 – Case 7014

75205 Paris Cedex 13 – FRANCE

175, rue du Chevaleret – 75013 Paris – FRANCE

Olivier.Laurent@ens-lyon.fr

January 2004

Abstract

Starting with the idea of reflexive objects in Selinger's control categories, we define three different denotational models of Parigot's untyped lambda-mu calculus. The first one is built from an intersection types system for the lambda-mu calculus leading to a generalization of Engeler's model of the untyped lambda calculus. The second model introduces correlation spaces (coming from Girard's model of classical logic) in the usual coherent model of the untyped lambda calculus. The third model is simply obtained by showing that Ker-Nickau-Ong's game model of the untyped lambda calculus is also a model of the untyped lambda-mu calculus.

1 Introduction

The study of models of the λ-calculus has been a major subject in the study of the λ-calculus and in the development of the Curry-Howard correspondence. Starting with the work of D. Scott [Sco72], a huge number of contributions addressed questions like the construction of models, the general characterization of models (in relation with combinatory algebras and categories), the comparison of models, the study of the λ-theories induced by these models, the relations with properties of syntax, ... In this way, domain theory led to the discovery of some important properties of terms, like continuity, stability, sequentiality, strong stability, ..., giving an abstract way of describing notions of computation.

In the beginning of the 90's, the work of T. Griffin [Gri90] has allowed to go through one of the main limitations of the Curry-Howard correspondence: intuitionism. He discovered that control operators such as `call/cc` in Scheme can be typed with classical (non intuitionistic) tautologies as Peirce's law. This has led to a lot of work on both the computer science and the logic sides for developing extensions of the relation between the λ-calculus and minimal natural deduction. Among them, M. Parigot's $\lambda\mu$-calculus [Par92] appears as a very satisfactory approach by adding a new atomic construction $\mu\alpha[\beta]t$ to the λ-calculus which corresponds to right-hand side structural rules of classical logic. From a typing judgments point of view, this corresponds to move from intuitionistic typing judgments $x_1 : A_1, \ldots, x_n : A_n \vdash t : A$ of the λ-calculus to classical typing judgments $x_1 : A_1, \ldots, x_n : A_n \vdash t : A \mid \alpha_1 : B_1, \ldots, \alpha_m : B_m$ where the α_is are μ-variables (also called *channel names*).

In the spirit of cartesian closed categories as models of the simply typed λ-calculus, categorical definitions of models of the simply typed $\lambda\mu$-calculus have been given. However this has shown to be a long time work and after L. Ong's work [Ong96], M. Hofmann-T. Streicher's work [HS02] and various others, P. Selinger introduced, ten years after T. Griffin's paper, the notion of *control category* [Sel01] which seems to be the most satisfactory answer. Control categories are described as extensions of cartesian closed categories with

a premonoidal structure [PR97] (endowing each object with a structure of monoid) which interacts in the appropriate way with the cartesian structure.

This leads to the current situation: models of the simply typed λ-calculus, of the untyped λ-calculus and of the simply typed $\lambda\mu$-calculus are well understood, but what about *models of the untyped $\lambda\mu$-calculus*? As far as we know, this question has been almost ignored. In the spirit of extending the Curry-Howard correspondence to classical features, it seems very important to have a good understanding of the computational behavior of the $\lambda\mu$-calculus. At some point, this requires to move to more abstract structures as denotational models of the $\lambda\mu$-calculus. A good knowledge in the untyped $\lambda\mu$-calculus should allow to go outside the simply typed world and to look at more general programming constructions in a setting with control operators: recursion, more general typing systems (subtyping, recursive types, ...), ... Important models of complex typed systems are built from models of the underlying untyped calculus: realizability models, models of `XML` related programming languages, ... In particular, using models of the untyped $\lambda\mu$-calculus, it could be possible to apply J.-L. Krivine's realizability [Kri01] in order to build models of powerful typed systems and even of ZF set theory.

Moving the knowledge about models of the untyped $\lambda\mu$-calculus at the level of what we know about the λ-calculus is a huge piece of work. This paper only tries, as a first step in this seemingly new area, to make clear how some usual constructions of models of the λ-calculus can be modified and extended to give (what we think to be) the first models of the untyped $\lambda\mu$-calculus. As a kind of general setting, we show that the notion of *reflexive object* in cartesian closed categories can be applied in control categories to move from models of the untyped λ-calculus to models of the untyped $\lambda\mu$-calculus. This shows that the idea of finding a model of a untyped calculus from a model of the associated simply typed one through an equation of the kind $D = D \to D$ is still valid for the $\lambda\mu$-calculus (a similar idea has been applied in a syntactical setting to build proof-nets for the untyped $\lambda\mu$-calculus [Lau03]).

Our first construction consists in the extension of an intersection types system [CDCV81, BCDC83] to the $\lambda\mu$-calculus. The key idea is to add a "union" construction $\sqcup$. The intersection $\sqcap$ is related with the comma on the left-hand side of typing judgments of the λ-calculus, since it allows to give a single common type $A \sqcap B$ to two different occurrences of a variable x with distinct types A and B. In the same spirit, union is related with the comma on the right-hand side of typing judgments of the $\lambda\mu$-calculus, allowing to give a common type $A \sqcup B$ to two different occurrences of a channel name α with distinct types A and B. In the λ-calculus we have to deal with interactions between $\to$ and $\sqcap$. Here things get a little more complicated since we have possible interactions between $\to$, $\sqcap$ and $\sqcup$. Using the idea that we can restrict types to the shape $A_1 \sqcap \cdots \sqcap A_n \to A$ in the λ-calculus, we use types restricted to unions of types of this shape. This leads to three levels of types in the typing system and to three levels of "power-sets" in the definition of Engeler's model (instead of two in the usual case).

This typing system leads to a very simple and concrete model of the $\lambda\mu$-calculus by a natural extension of Engeler's model. Moreover by applying the usual techniques of intersection types, we get characterizations of solvable terms and normalizable terms of the untyped $\lambda\mu$-calculus. This is the main syntactical contribution of this paper.

The only related work we are aware of concerning intersection types for classical extensions of the λ-calculus has been carried out independently by D. Dougherty, S. Ghilezan and P. Lescanne [LGD03]. They give a typing system for the $\overline{\lambda}\mu\tilde{\mu}$-calculus [CH00] which characterizes strongly normalizable terms. However, in their system, typing is not preserved by expansion and cannot lead to a denotational model. The reason for that probably comes from the simplicity of their system which only uses $\to$ and $\sqcap$. It would be interesting to see the precise relation between these systems. It might be the case that a system related to theirs could be obtained by collapsing $\sqcup$ and $\sqcap$ in ours.

The introduction of *stability* [Ber78] in models of the λ-calculus by G. Berry has been followed by the definition of the model of *coherence spaces* by J.-Y. Girard [Gir86, Gir87]. This model, which can by easily presented as an interpretation of terms as cliques in a reflexive graph, has been used for modeling both the untyped λ-calculus and a propositional sequent calculus for classical logic LC [Gir91] (LC and the simply typed $\lambda\mu$-calculus are two strongly related systems born together in Paris around 1991). Using both

constructions, we show that merging them properly allows us to build a coherent model of the untyped $\lambda\mu$-calculus (which validates also the $\eta\theta$-reduction). The key point is to introduce, at each step of the inductive construction of the untyped model, a monoid structure corresponding to right-hand side structural rules of classical logic. The model of LC (correlation spaces) which is based on this monoid structure ensures us that all the constructions preserve monoids. From a categorical point of view, we get a reflexive object in the control category of correlation spaces. We do not claim that this section introduces very new ideas. Our goal is just to show how we should proceed if we want to put together a model of the untyped λ-calculus and a model of the simply typed $\lambda\mu$-calculus to build a model of the untyped $\lambda\mu$-calculus.

Game models have been very important for the semantics of extensions of the simply typed λ-calculus by giving the first fully abstract models. The model we give for the untyped $\lambda\mu$-calculus is mainly an example of what we can do with game models. It is really a direct application of what we learned about game models of the untyped λ-calculus [KNO02] and about game models of the simply typed $\lambda\mu$-calculus [Lai97, Lau02, Lau05]. These two directions happen to fit very well together and to immediately give a model of the untyped $\lambda\mu$-calculus compatible with $\eta\theta$-reduction.

Remark: In the whole paper, we will use the word "*classical*" to mean "related with classical computation" that is with the $\lambda\mu$-calculus (as opposed to "intuitionistic" for the λ-calculus). This has not to be confused with "*usual*"!

2 Lambda-mu calculus

While working on deduction systems for classical logic (free deduction, ...), M. Parigot introduced an extension of natural deduction for classical logic based on sequents with more than one formula on the right-hand side. Under the Curry-Howard correspondence, he extracted the underlying extension of the λ-calculus: the $\lambda\mu$-calculus [Par92] which can be described as both a typed and an untyped calculus.

Terms. Given a set of λ-variables x, y, ... and a set of μ-variables α, β, ... The $\lambda\mu$-terms are presented as an extension of the λ-calculus syntax with a new construction:

$$\begin{array}{lcl} t & ::= & x \quad | \quad \lambda x.t \quad | \quad (t\ t) \quad | \quad \mu\alpha[\beta]t \\ n & ::= & [\beta]t \end{array}$$

where μ is a binder for μ-variables, so that β is free in $\mu\alpha[\beta]t$ if and only if $\alpha \neq \beta$ and α is never free in $\mu\alpha[\beta]t$. Terms are considered up to α-equivalence on both λ and μ variables.

The intermediary notation n is often useful. These terms are called *named terms* and are such that if n is a named term, $\mu\alpha.n$ is a $\lambda\mu$-term.

Reduction rules. Together with the usual β-reduction, two new reduction rules are added in the $\lambda\mu$-calculus:

$$\begin{array}{r} (\lambda x.t\ u) \to_\beta t[{}^{u}/_{x}] \\ (\mu\alpha.n\ u) \to_\mu \mu\alpha.n[{}^{[\alpha](v\ u)}/_{[\alpha]v}] \\ [\beta]\mu\alpha.n \to_\rho n[{}^{\beta}/_{\alpha}] \end{array}$$

The μ-reduction is the main novelty of the $\lambda\mu$-calculus. A simple computational interpretation is to say that an argument u given to a term $\mu\alpha.n$ is propagated to all the subterms listening on the channel α (those starting with $[\alpha]$). The substitution $t[{}^{[\alpha](v\ u)}/_{[\alpha]v}]$ (which could also be denoted by $t[{}^{[\alpha](\_\ u)}/_{[\alpha]\_}]$) is obtained

by replacing any subterm v of t under a $[\alpha]$ by $(v\ u)$. It is inductively given by:

$$\begin{aligned}
x[^{[\alpha](v\ u)}/_{[\alpha]v}] &= x \\
(\lambda x.t)[^{[\alpha](v\ u)}/_{[\alpha]v}] &= \lambda x.(t[^{[\alpha](v\ u)}/_{[\alpha]v}]) \\
(t\ t')[^{[\alpha](v\ u)}/_{[\alpha]v}] &= (t[^{[\alpha](v\ u)}/_{[\alpha]v}]\ t'[^{[\alpha](v\ u)}/_{[\alpha]v}]) \\
(\mu\beta[\alpha]t)[^{[\alpha](v\ u)}/_{[\alpha]v}] &= \mu\beta[\alpha](t[^{[\alpha](v\ u)}/_{[\alpha]v}]\ u) \\
(\mu\beta[\gamma]t)[^{[\alpha](v\ u)}/_{[\alpha]v}] &= \mu\beta[\gamma](t[^{[\alpha](v\ u)}/_{[\alpha]v}])
\end{aligned}$$

with $\beta \neq \alpha$ and $\gamma \neq \alpha$

The ρ-reduction is a simple simplification rule in order to remove successions $\mu\alpha[\beta]\mu\alpha'[\beta']$ in terms. The substitution $t[^{\beta}/_{\alpha}]$ is the natural notion of substitution of a free μ-variable by another one.

We denote by $\to$ the union of the three relations $\to_\beta$, $\to_\mu$ and $\to_\rho$.

Proposition 1 (Church-Rosser [Par92])
The reduction rules of the $\lambda\mu$-calculus are confluent.

Definition 1 (Solvable term)
A *head normal form* is a term of the shape:

$$\lambda \dots \lambda\mu\alpha_1[\beta_1] \cdots \lambda \dots \lambda\mu\alpha_2[\beta_2] \cdots (x\ t_1 \dots t_n)$$

without any succession $\mu\alpha_i[\beta_i]\mu\alpha_{i+1}[\beta_{i+1}]$.

A $\lambda\mu$-term is *solvable* if it reduces into a head normal form.

Equational theories. The equivalence relation $=_{\beta\mu\rho}$ is the smallest congruence containing the reflexive symmetric transitive closure of $\to$.

$=_{\beta\eta\mu\rho\theta}$ is obtained by extending $=_{\beta\mu\rho}$ with the following two equations:

$$\begin{aligned}
\lambda x.(t\ x) &=_\eta t && x \notin t \\
\mu\alpha[\alpha]t &=_\theta t && \alpha \notin t
\end{aligned}$$

where θ is the natural rule of the $\lambda\mu$-calculus corresponding to η.

Our goal is not to study the extension of the theory of λ-algebras, λ-models, ... [Bar84]. We just give a very simple notion of models of the $\lambda\mu$-calculus and we will see a natural categorical way of building them in the spirit of what happens for the λ-calculus.

Definition 2 (Denotational model)
A pair $(\mathcal{M}, [\![.]\!])$ is a *denotational model* (or just *model*) (resp. *η-model*) of the $\lambda\mu$-calculus if $\mathcal{M}$ is a set, $[\![.]\!]$ is a function from the set of closed $\lambda\mu$-terms into $\mathcal{M}$ and $t_1 =_{\beta\mu\rho} t_2 \Rightarrow [\![t_1]\!] = [\![t_2]\!]$ (resp. $t_1 =_{\beta\eta\mu\rho\theta} t_2 \Rightarrow [\![t_1]\!] = [\![t_2]\!]$).

3 Categorical approach

We show how the notion of reflexive objects in cartesian closed categories (which gives models of the untyped λ-calculus) can be easily applied to control categories and to the $\lambda\mu$-calculus.

Definition 3 (Control category [Sel01])
A category $(\mathcal{C}, \times, \top, \to, \parr, \bot)$ is a *control category* if:

- $(\mathcal{C}, \times, \top, \to)$ is a cartesian closed category;
- $(\mathcal{C}, \parr, \bot)$ is a symmetric premonoidal category [PR97];

- each object A has a given $\parr$-monoidal structure (A, wk_A, ctr_A) (with $wk_A \in \mathcal{C}(\bot, A)$ and $ctr_A \in \mathcal{C}(A \parr A, A)$);
- $\parr$ distributes over $\times$;
- there exists a natural isomorphism str between $(A \to B) \parr C$ and $A \to (B \parr C)$;
- three coherence conditions on morphisms from $(A \to B) \parr (C \to D)$ to $C \to A \to (B \parr D)$, from $(A \to B) \parr (A \to B)$ to $A \to B$ and from $\bot$ to $A \to B$ (see [Sel01]) are satisfied.

The following notations for canonical morphisms, coming from the previously described structures, will be useful:

$$(A \to B) \times A \xrightarrow{eval} B$$

$$A \xrightarrow{\Delta_A} A \times A$$

$$A_1 \times \cdots \times A_n \xrightarrow{\pi_i^n} A_i$$

$$A \parr B \xrightarrow{commut_\parr} B \parr A$$

$$\bot \xrightarrow{wk_A} A$$

$$A \parr A \xrightarrow{ctr_A} A$$

$$A \parr (B \times C) \xrightarrow{distr} (A \parr B) \times (A \parr C)$$

$$A \xrightarrow{\tilde{f}} (B \to C) \text{ if } (A \times B) \xrightarrow{f} C$$

A morphism f from A to B in a control category is *central* if for any morphism g from C to D, $(f \parr id_D) \circ (id_A \parr g) = (id_B \parr g) \circ (f \parr id_C)$ from $A \parr C$ to $B \parr D$.

Definition 4 (Reflexive object)
A *reflexive object* (resp. *η-reflexive object*) in a control category $\mathcal{C}$ is a triple (O, i, j) where O is an object, i is a morphism from $O \to O$ to O and j is a *central* morphism from O to $O \to O$ and $j \circ i = id_{O \to O}$ (resp. $j \circ i = id_{O \to O}$ and $i \circ j = id_O$).

Proposition 2 (Categorical models)
A reflexive object (resp. η-reflexive object) O in a (locally small) control category $\mathcal{C}$ gives a model (resp. η-model) $(\mathcal{C}(\top, O), [\![.]\!]_O)$ of the $\lambda\mu$-calculus (where $[\![.]\!]_O$ is obtained from the usual interpretation of the simply typed $\lambda\mu$-calculus in $\mathcal{C}$ [Sel01], see table 1).

PROOF: A $\lambda\mu$-term t with its free λ-variables among $\Gamma = x_1, \ldots, x_n$ and its free μ-variables among $\Delta = \alpha_1, \ldots, \alpha_m$ is interpreted as a morphism $[\![t]\!]_{\Gamma \cdot \Delta}$ from $O \times \cdots \times O$ (n times) to $O \parr \cdots \parr O$ ($m+1$ times).

We adapt the usual interpretation of the simply typed $\lambda\mu$-calculus in control categories [Sel01] as we would do for the λ-calculus with cartesian closed categories (see table 1, where we omit the isomorphisms related with units, commutativity and associativity except when they are particularly crucial).

Using the fact that a control category is a η-model of the simply typed $\lambda\mu$-calculus, we can show:

- $[\![(\lambda x.t\ u)]\!]_{\Gamma \cdot \Delta} = [\![t[^u/_x]]\!]_{\Gamma \cdot \Delta}$ with $j \circ i = id_{O \to O}$
- $[\![(\mu\alpha.n\ u)]\!]_{\Gamma \cdot \Delta} = [\![\mu\alpha.n[^{[\alpha](v\ u)}/_{[\alpha]v}]]\!]_{\Gamma \cdot \Delta}$ with j central
- $[\![\mu\gamma[\beta]\mu\alpha.n]\!]_{\Gamma \cdot \Delta} = [\![\mu\gamma.n[^\beta/_\alpha]]\!]_{\Gamma \cdot \Delta}$
- $[\![\lambda x.(t\ x)]\!]_{\Gamma \cdot \Delta} = [\![t]\!]_{\Gamma \cdot \Delta}$ if $x \notin t$ with $i \circ j = id_O$
- $[\![\mu\alpha[\alpha]t]\!]_{\Gamma \cdot \Delta} = [\![t]\!]_{\Gamma \cdot \Delta}$ if $\alpha \notin t$

If t is a closed $\lambda\mu$-term, we define $[\![t]\!]_O = [\![t]\!]_{\emptyset \cdot \emptyset} \in \mathcal{C}(\top, O)$. We have just shown the result: $(\mathcal{C}(\top, O), [\![.]\!]_O)$ is a model (resp. η-model) if O is a reflexive (resp. η-reflexive) object in $\mathcal{C}$. □

$$[\![x_i]\!]_{\Gamma\cdot\Delta} = \prod_n O \xrightarrow{\pi_i^n} O \xrightarrow{O \parr \sum_m wk_O} O \parr \sum_m O$$

$$[\![\lambda x.t]\!]_{\Gamma\cdot\Delta} = \prod_n O \xrightarrow{\widetilde{[\![t]\!]}_{(x,\Gamma)\cdot\Delta}} (O \to \sum_{m+1} O) \xrightarrow{str^{-1}} (O \to O) \parr \sum_m O \xrightarrow{i \parr \sum_m O} O \parr \sum_m O$$

$$[\![(t\ u)]\!]_{\Gamma\cdot\Delta} = \prod_n O \xrightarrow{\Delta_{\Pi_n O}} \prod_n O \times \prod_n O \xrightarrow{[\![t]\!]_{\Gamma\cdot\Delta} \times [\![u]\!]_{\Gamma\cdot\Delta}} \sum_{m+1} O \times \sum_{m+1} O$$
$$\xrightarrow{(j \parr \sum_m O) \times \sum_{m+1} O} ((O \to O) \parr \sum_m O) \times (O \parr \sum_m O) \xrightarrow{distr^{-1}} ((O \to O) \times O) \parr \sum_m O \xrightarrow{eval \parr \sum_m O} O \parr \sum_m O$$

$$[\![\mu\alpha[\alpha]t]\!]_{\Gamma\cdot\Delta} = \prod_n O \xrightarrow{[\![t]\!]_{\Gamma\cdot(\alpha,\Delta)}} O \parr O \parr \sum_m O \xrightarrow{ctr_O \parr \sum_m O} O \parr \sum_m O$$

$$[\![\mu\alpha[\beta]t]\!]_{\Gamma\cdot(\beta,\Delta)} = \prod_n O \xrightarrow{[\![t]\!]_{\Gamma\cdot(\alpha,\beta,\Delta)}} O \parr O \parr O \parr \sum_m O \xrightarrow{commut \parr \sum_{m+1} O} O \parr O \parr O \parr \sum_m O \xrightarrow{O \parr ctr_O \parr \sum_m O} O \parr \sum_{m+1} O$$

with $\prod_n O = O \times \cdots \times O$ (n times) and $\sum_m O = O \parr \cdots \parr O$ (m times).

Table 1: Interpretation of the untyped $\lambda\mu$-calculus in a control category with a reflexive object

4 A continuous model: Intersection types

In order to define a denotational model of the untyped $\lambda\mu$-calculus, we are going to define a typing system for untyped terms which verifies both subject reduction and subject expansion. Such a system defines a model by associating with each term the set of its possible types. As usual with the λ-calculus, we use *intersection types* to make typable a reasonable subset of $\lambda\mu$-terms. The corresponding model is an extension of Engeler's model [Kri93] of continuous functions.

4.1 Intersection types for the $\lambda\mu$-calculus

We develop an intersection types system [CDCV81, BCDC83] for the $\lambda\mu$-calculus, based on the introduction of a union $\sqcup$ connective for dealing with channel names of $\lambda\mu$-terms as the intersection $\sqcap$ deals with λ-variables.

If X denotes atomic types, *types* are of three kinds:

arrow types	A	::=	$X \mid I \to A$
intersection types	I	::=	$\Omega \mid U \sqcap \cdots \sqcap U$
union types	U	::=	$A \sqcup \cdots \sqcup A$

and a typing judgment has the shape $x_1 : I_1, \ldots, x_n : I_n \vdash t : U \mid \alpha_1 : U_1, \ldots, \alpha_m : U_m$. Types are considered up to commutativity, idempotency and unit element for $\sqcap$ and $\sqcup$ ($U \sqcap V = V \sqcap U$, $U \sqcap U = U$, $U \sqcap \Omega = U$, $A \sqcup B = B \sqcup A$, $A \sqcup A = A$).

If $U = A_1 \sqcup \cdots \sqcup A_p$ ($p > 0$) and I are two types, we use the notation $U \in I$ to say that U is one of the components of I (*i.e.* $U \notin \Omega$ and $U \in U_1 \sqcap \cdots \sqcap U_q$ if $U = U_k$), and the notation $I \to U$ for $(I \to A_1) \sqcup \cdots \sqcup (I \to A_p)$ (well defined since unions, as opposed to intersections, are never empty).

Typing rules are given on table 2.

$$\frac{}{\Gamma, x : I \vdash x : U \mid \Delta}\ var \quad \text{if } U \in I \qquad \frac{\Gamma, x : I \vdash t : U \mid \Delta}{\Gamma \vdash \lambda x.t : I \to U \mid \Delta}\ lam$$

$$\frac{\Gamma \vdash t : (I_1 \to A_1) \sqcup \cdots \sqcup (I_p \to A_p) \mid \Delta \qquad (\cdots\ \Gamma \vdash u : U \mid \Delta\ \cdots \quad \forall U \in I_1 \sqcap \cdots \sqcap I_p)}{\Gamma \vdash (t\ u) : A_1 \sqcup \cdots \sqcup A_p \mid \Delta}\ app$$

$$\frac{\Gamma \vdash t : V \mid \Delta, \alpha : U}{\Gamma \vdash \mu\alpha[\alpha]t : V \sqcup U \mid \Delta}\ mu_1 \qquad \frac{\Gamma \vdash t : W \mid \Delta, \alpha : U, \beta : V}{\Gamma \vdash \mu\alpha[\beta]t : U \mid \Delta, \beta : W \sqcup V}\ mu_2 \quad \text{if } \alpha \neq \beta$$

Table 2: Typing rules for intersection types

Remark: The last two rules of table 2 are equivalent to:

$$\frac{\Gamma \vdash t : V \mid \Delta, \alpha : U}{\Gamma \vdash [\alpha]t \mid \Delta, \alpha : V \sqcup U} \qquad \frac{\Gamma \vdash n \mid \Delta, \alpha : U}{\Gamma \vdash \mu\alpha.n : U \mid \Delta}$$

The three following lemmas are proved by induction on the typing derivation for t:

Lemma 1 (λ-substitution)
If $\Gamma, x : I \vdash t : U \mid \Delta$ *and for each* $V \in I$, $\Gamma \vdash u : V \mid \Delta$ *($x \notin u$), then* $\Gamma \vdash t[^u/_x] : U \mid \Delta$.

Lemma 2 (ρ-substitution)
If $\Gamma \vdash t : U \mid \Delta, \beta_1 : V_1, \beta_2 : V_2$ *then* $\Gamma \vdash t[^\beta/_{\beta_1}, ^\beta/_{\beta_2}] : U \mid \Delta, \beta : V_1 \sqcup V_2$ *($\beta \notin \Delta$).*

Lemma 3 (μ-substitution)
If $\Gamma \vdash t : U \mid \Delta, \alpha : (I_1 \to A_1) \sqcup \cdots \sqcup (I_p \to A_p)$ *and for each* $V \in I_1 \sqcap \cdots \sqcap I_p$, $\Gamma \vdash u : V \mid \Delta$ *($\alpha \notin u$), then* $\Gamma \vdash t[^{[\alpha](v\ u)}/_{[\alpha]v}] : U \mid \Delta, \alpha : A_1 \sqcup \cdots \sqcup A_p$.

Theorem 1 (Subject equivalence)
If $t_1 =_{\beta\mu\rho} t_2$ *and* $\Gamma \vdash t_1 : U \mid \Delta$ *then* $\Gamma \vdash t_2 : U \mid \Delta$.

Proof: We use lemmas 1, 2 and 3 to prove the preservation of typing by β, ρ and μ reduction.

We prove by induction on the size of a derivation of $\Gamma \vdash t[^u/_x] : U \mid \Delta$ that we can find an intersection type I such that $\Gamma, x : I \vdash t : U \mid \Delta$ and for each element V of I, $\Gamma \vdash u : V \mid \Delta$. We then conclude that $\Gamma \vdash (\lambda x.t\ u) : U \mid \Delta$.

We prove by induction on the size of a derivation of $\Gamma \vdash t[^\beta/_\alpha] : V \mid \Delta, \beta : U$ that $U = U_1 \sqcup U_2$ and $\Gamma \vdash t : V \mid \Delta, \beta : U_1, \alpha : U_2$. We then show that $\Gamma \vdash n[^\beta/_\alpha] \mid \Delta$ entails $\Gamma \vdash [\beta]\mu\alpha.n \mid \Delta$ (with the modified μ-rules of the preceding remark).

We prove by induction on the size of a derivation of $\Gamma \vdash t[^{[\alpha](v\ u)}/_{[\alpha]v}] : U \mid \Delta, \alpha : V$ that $V = A_1 \sqcup \cdots \sqcup A_p$, $\Gamma \vdash t : U \mid \Delta, \alpha : (I_1 \to A_1) \sqcup \cdots \sqcup (I_p \to A_p)$ and for each element W of $I_1 \sqcap \cdots \sqcap I_p$, $\Gamma \vdash u : W \mid \Delta$. And we conclude that $\Gamma \vdash \mu\alpha.n[^{[\alpha](v\ u)}/_{[\alpha]v}] : U \mid \Delta$ entails $\Gamma \vdash (\mu\alpha.n\ u) : U \mid \Delta$. □

Corollary 1 (Intersection model)
Let $\mathcal{U}$ *be the set of union types, and for any closed term* t $[\![t]\!]_\mathcal{U} = \{U \in \mathcal{U} \mid\ \vdash t : U \mid\ \}$, $(\mathcal{U}, [\![.]\!]_\mathcal{U})$ *is a model of the* $\lambda\mu$*-calculus.*

4.2 Classical Engeler's model

Engeler's model of the λ-calculus is strongly related with intersection types for the λ-calculus. In the same spirit, starting with our generalized intersection types with unions, we build the required hierarchy of sets.

$$[\![x_k]\!]_E(i_1,\ldots,i_n) = \left\{ (u,u_1,\ldots,u_m) \;\middle|\; u \in i_k \wedge (u_1,\ldots,u_m) \in \mathcal{U}^m \right\}$$

$$[\![\lambda x_0.t]\!]_E(i_1,\ldots,i_n) = \left\{ \big(\{(i_0,a_1),\ldots,(i_0,a_p)\},u_1,\ldots,u_m\big) \;\middle|\; i_0 \in \mathcal{P}_{fin}(\mathcal{U}) \right.$$
$$\left. \wedge \big(\{a_1,\ldots,a_p\},u_1,\ldots,u_m\big) \in [\![t]\!]_E(i_0,i_1,\ldots,i_n) \right\}$$

$$[\![(t\ t')]\!]_E(i_1,\ldots,i_n) = \left\{ \big(\{a_1,\ldots,a_p\},u_1,\ldots,u_m\big) \;\middle|\; \exists (j_1,\ldots,j_p) \in \mathcal{P}_{fin}(\mathcal{U})^p \right.$$
$$\left. \big(\{(j_1,a_1),\ldots,(j_p,a_p)\},u_1,\ldots,u_m\big) \in [\![t]\!]_E(i_1,\ldots,i_n) \wedge \forall u \in \bigcup_{1\le k\le p} j_k\ ,\ (u,u_1,\ldots,u_m) \in [\![t']\!]_E(i_1,\ldots,i_n) \right\}$$

$$[\![\mu\alpha_k[\alpha_k]t]\!]_E(i_1,\ldots,i_n) = \left\{ (u \cup u_k,u_1,\ldots,u_{k-1},u_{k+1},\ldots,u_m) \;\middle|\; (u,u_1,\ldots,u_m) \in [\![t]\!]_E(i_1,\ldots,i_n) \right\}$$

$$[\![\mu\alpha_k[\alpha_{k'}]t]\!]_E(i_1,\ldots,i_n) = \left\{ (u_k,u_1,\ldots,u_{k-1},u_{k+1},\ldots,u_{k'-1},u_{k'} \cup u,u_{k'+1},\ldots,u_m) \;\middle|\; (u,u_1,\ldots,u_m) \in [\![t]\!]_E(i_1,\ldots,i_n) \right\}$$

Table 3: Interpretation of terms in classical Engeler's model

Let $\mathcal{S}$ be a set of atoms, we define:

$$\begin{aligned} \mathcal{A} &::= \mathcal{S} \mid \mathcal{P}_{fin}(\mathcal{P}^{\star}_{fin}(\mathcal{A})) \times \mathcal{A} \\ \mathcal{U} &= \mathcal{P}^{\star}_{fin}(\mathcal{A}) \\ \mathcal{E} &= \mathcal{P}(\mathcal{U}) \end{aligned}$$

where $\mathcal{P}(\mathcal{X})$, $\mathcal{P}_{fin}(\mathcal{X})$ and $\mathcal{P}^{\star}_{fin}(\mathcal{X})$ are, respectively, the set of the subsets of $\mathcal{X}$, of the finite subsets of $\mathcal{X}$, and of the non-empty finite subsets of $\mathcal{X}$. The definition of $\mathcal{A}$ directly comes from the grammar of types with $\mathcal{P}_{fin}(.)$ corresponding to the construction of intersections and $\mathcal{P}^{\star}_{fin}(.)$ to the construction of unions (required to be non-empty).

If the free variables of t are contained in $\{x_1,\ldots,x_n,\alpha_1,\ldots,\alpha_m\}$, we define the continuous function $[\![t]\!]_E$ from $\mathcal{P}(\mathcal{U})^n$ into $\mathcal{P}(\mathcal{U}^{m+1})$ as in table 3.

In the particular case of the λ-calculus, we can restrict $\mathcal{U}$ to singletons and we can consider only $m = 0$. In this way, we get back the usual Engeler's model of the λ-calculus.

If we consider the intersection types system with types built with the set $\mathcal{S}$ as atomic types, we have a one-to-one correspondence between arrow types and elements of $\mathcal{A}$, between union types and elements of $\mathcal{U}$ and between intersection types and elements of $\mathcal{P}_{fin}(\mathcal{U})$. With the appropriate adapted notations we then have:

Proposition 3 (Engeler and intersection types)
For any $\lambda\mu$-term t, $[\![t]\!]_E(I_1,\ldots,I_n) = \{(U,U_1,\ldots,U_m) \mid x_1 : I_1,\ldots,x_n : I_n \vdash t : U \mid \alpha_1 : U_1,\ldots,\alpha_m : U_m\}$.

Proof: By a simple induction on t with tables 2 and 3. □

Corollary 2 (Engeler's model)
If t is closed, we define $[\![t]\!]_{\mathcal{E}} = [\![t]\!]_E \in \mathcal{E}$. $(\mathcal{E},[\![.]\!]_{\mathcal{E}})$ is a model of the $\lambda\mu$-calculus.

Proof: By corollary 1 and proposition 3. □

4.3 Syntactical properties

As a key application of intersection types, we can derive typing characterizations of various normalization properties of $\lambda\mu$-terms. This relates the denotational model and syntactical properties, and gives a useful way of looking at the syntax. We give here the natural extensions of the results concerning the λ-calculus.

Proposition 4 (Reduction of typing derivations)
If $\Gamma \vdash t : U \mid \Delta$, there exists a term t' with $t \to^* t'$ such that $\Gamma \vdash t' : U \mid \Delta$ in a typing derivation without any lam-app, mu-app nor mu-mu succession of rules.

Proof: A redex in a typing derivation is a lam-app, mu-app or mu-mu succession of rules (respectively called β-redexes, μ-redexes and ρ-redexes). The size of a β-redex (resp. μ-redex) r is the pair $(|A|, 1)$ (resp. $(|A|, 2)$) where $|A|$ is the size of the functional type of the redex. Let π be the typing derivation of $\Gamma \vdash t : U \mid \Delta$, and let (k, m) be the maximal size of its β and μ redexes and n be the number of such maximal redexes in π, the size of π is $(k, m, n, |\pi|)$ where $|\pi|$ is the number of rules of π. We show the result by induction on $(k, m, n, |\pi|)$ (with the lexicographic order). If π contains some ρ-redex, we eliminate it (as for lemma 2). This makes $|\pi|$ decrease and does not modify k, m, and n. Otherwise, we choose a maximal (β or μ) redex without any maximal redex above it and we eliminate it (as for lemmas 1 and 3). We can show that it makes the size of π decrease. □

Corollary 3 (Sub-formula property)
If $\Gamma \vdash t : U \mid \Delta$, there exists a term t' with $t \to^* t'$ such that $\Gamma \vdash t' : U \mid \Delta$ in a typing derivation which contains only sub-formulas of Γ, U and Δ.

Corollary 4 (Solvable terms)
A $\lambda\mu$-term t is solvable if and only if there exist Γ, U and Δ such that $\Gamma \vdash t : U \mid \Delta$.

Corollary 5 (Normalizable terms)
A $\lambda\mu$-term t is normalizable if and only if there exist Γ, U and Δ containing no Ω and such that $\Gamma \vdash t : U \mid \Delta$.

It is certainly the case that strongly normalizable $\lambda\mu$-terms are exactly those typable without any use of the Ω type in typing derivations. This would require a more complicated proof as in the λ-calculus case, probably without leading to anything new.

5 A stable model: Correlation spaces

Based on the notion of stable functions [Ber78] in domains, *coherence spaces* have been introduced by Girard [Gir86, Gir87] to study sum types. They lead to a "concrete" notion of domains coming from a web: a reflexive graph. In a typed setting they allow to give a model of system F [Gir86]. In an untyped setting, it is possible to solve the equation $D \to D = D$ leading to an extensional model of the untyped λ-calculus.

Definition 5 (Coherence space)
A *coherence space* A is a reflexive (symmetric) graph $(|A|, \frown_A)$.

Given a set S, we denote by $\mathcal{M}_S$ the set of finite multisets of elements of S. A *clique* in the coherence space A is a subset a (denoted by $a \sqsubset A$) of $|A|$ such that for all $x, y \in a$, $x \frown_A y$. We denote by $\mathsf{FC}(A)$ the set of the elements a of $\mathcal{M}_{|A|}$ such that the underlying set of a is a clique in A.

Constructions.

- $A^\perp = (|A|, \asymp_A)$ where $x \asymp_A y$ if $\neg(x \frown_A y)$ or $x = y$.
- $A \parr B = (|A| \times |B|, \frown_{A \parr B})$ where $(x, y) \frown_{A \parr B} (x', y')$ if $x \frown_A x'$ or $y \frown_B y'$ (with $x \frown_A y$ when $x \frown_A y$ and $x \neq y$).
- $!A = (\mathsf{FC}(A), \frown_{!A})$ where $a \frown_{!A} b$ if $a + b \in |!A|$.

- $A \multimap B = A^\perp \parr B$ and $A \to B = !A \multimap B$.

In order to move from a model of intuitionistic (or linear) logic to (one of) the first denotational model of classical logic, Girard refined coherence spaces by correlation spaces [Gir91] coming with some additional structure. Intuitively, these richer objects come with the information required to interpret structural rules (weakening and contraction) on the right-hand side of sequents in classical sequent calculus.

Definition 6 (Correlation space)
A *correlation space* A is a tuple $(|A|, \frown_A, wk_A, ctr_A)$ where $(|A|, \frown_A)$ is a coherence space, $wk_A \sqsubset A$ and $ctr_A \sqsubset (A \parr A) \multimap A$ satisfying:

$$\forall x \in |A| \quad \exists y \in wk_A \quad ((x,y),x) \in ctr_A$$
$$\forall ((x,y),z) \in ctr_A \quad ((y,x),z) \in ctr_A$$
$$\forall ((x,y),u) \in ctr_A \quad \forall ((u,z),t) \in ctr_A \quad \exists v \in |A| \quad ((y,z),v) \in ctr_A \wedge ((x,v),t) \in ctr_A$$

A correlation space is a $\parr$-monoid in the cartesian closed category of coherence spaces.

Definition 7 (Correlation subspace)
Let $A = (|A|, \frown_A, wk_A, ctr_A)$ be a correlation space, a *correlation subspace* B of A is a correlation space $(|B|, \frown_B, wk_B, ctr_B)$ such that $B \subset A$, $\frown_B = \frown_A \cap (|B| \times |B|)$, $wk_B = wk_A \cap |B|$ and $ctr_B = ctr_A \cap (|B| \times |B| \times |B|)$.

Constructions. The correlation structure is not preserved by all the constructions of coherence spaces. In particular, if A is a correlation space, $A^\perp$ and $!A$ are not correlation spaces. While if A is any coherence space, there is a natural way of putting a correlation structure on the coherence space $?A = (!A^\perp)^\perp$. The main cases we need are: if A and B are correlation spaces, then $A \parr B$ and $?A^\perp$ are correlation spaces (it is also true for $\bot$, $\top$ and $A \,\&\, B$, see [Gir91]).

- $A \parr B = (|A| \times |B|, \frown_{A \parr B}, wk_A \times wk_B, ctr_{A \parr B})$ with $ctr_{A \parr B} = \{((x,x'),(y,y'),(z,z')) \mid (x,y,z) \in ctr_A \wedge (x',y',z') \in ctr_B\}$.
- $?A^\perp = (\mathsf{FC}(A), \frown_{(!A)^\perp}, \{[\,]\}, ctr_{?A^\perp})$ with $ctr_{?A^\perp} = \{(a,b,a+b) \mid a+b \in |?A^\perp|\}$.

so that $A \to B = ?A^\perp \parr B$ has a correlation structure as soon as B does.

As remarked in [LR03]:

Proposition 5 (Correlation category)
Correlation spaces with morphisms from A to B given by cliques of $A \to B$ is a control category.

In order to build a reflexive object in this category (thus a model of the $\lambda\mu$-calculus), we follow the construction usually used for the λ-calculus as given in [Kri93].

Let $(|S|, \frown_S, wk_S, ctr_S)$ be a correlation space such that none of the elements of $|S|$ are ordered pairs. We define the correlation space C_n by:

$$C_0 = (|S|, \frown_S, wk_S, ctr_S)$$
$$|C_{n+1}| = ((\mathsf{FC}(C_n) \times |C_n|) \setminus (\{[\,]\} \times |S|)) \cup |S|$$

The coherence relation of C_{n+1}, and the cliques $wk_{C_{n+1}}$ and $ctr_{C_{n+1}}$ are defined in the natural way such that the correlation space C_{n+1} is isomorphic to $C_n \to C_n$: we have an immediate bijection i_n between $|C_n \to C_n|$ and $|C_{n+1}|$, and we define $\frown_{C_{n+1}}$ in such a way that $i_n(x) \frown_{C_{n+1}} i_n(y) \iff x \frown_{C_n \to C_n} y$, ...

Lemma 4
For all n, C_n is a correlation subspace of C_{n+1}.

The correlation space C is given by $|C| = \bigcup_{n \geq 0} |C_n|$, $\frown_C = \bigcup_{n \geq 0} \frown_{C_n}$, $wk_C = \bigcup_{n \geq 0} wk_{C_n}$ and $ctr_C = \bigcup_{n \geq 0} ctr_{C_n}$.

Theorem 2
C is a correlation space which is isomorphic to $C \to C$.

PROOF: By definition, $(|C|, \frown_C)$ is a coherence space. If $x, y \in wk_C$, let n be such that $x, y \in C_n$, we have $x, y \in wk_{C_n}$ thus $x \frown_{C_n} y$ and $x \frown_C y$ so that wk_C is a clique of C. In the same way we prove that $ctr_C \sqsubset (C \parr C) \multimap C$, and that the required equations are satisfied.

We define the function i from $\mathsf{FC}(C) \times |C|$ into $|C|$ by $i([\,], x) = x$ if $x \in |S|$ and $i(a, x) = (a, x)$ otherwise. The function i is a bijection since it is clearly injective and since any point in $|C|$ is either in $|S|$ or in some $(\mathsf{FC}(C_n) \times |C_n|) \setminus (\{[\,]\} \times |S|)$. We finally verify that $i(\frown_{C \to C}) = \frown_C$, $i(wk_{C \to C}) = wk_C$ and $i(ctr_{C \to C}) = ctr_C$. □

Corollary 6 (Correlation model)
C is a η-model of the $\lambda\mu$-calculus.

PROOF: C is a η-reflexive object in the control category of correlation spaces and we conclude with proposition 2. □

6 A sequential model: Game semantics

The introduction of game semantics [AJM00, HO00, Nic94] for languages as powerful as PCF led to the definition of fully abstract models of many typed languages (with control operators, states, ...).

In [KNO02], Ker, Nickau and Ong define a η-reflexive object in a cartesian closed category of games. It appears that their category is in fact a control category [Lai97, Lau02, Lau05] (thus a model of the simply typed $\lambda\mu$-calculus) so that their model is a η-model of the $\lambda\mu$-calculus.

We recall the main definitions of game semantics, see [McC96, Lai98, Har99] for more details.

Definition 8 (Arena)
An *arena* A is a forest, that is a partial order $(|A|, \leq_A)$ such that for all x, $x{\downarrow} = (\{y \in |A| \mid y \leq_A x\}, \leq_A)$ is a finite total order. The elements of $|A|$ are called the *moves* of A. If x is a minimal element of A (called an *initial* move), we use the notation $\vdash_A x$, otherwise we use $y \vdash_A x$ if y is the maximal element of $x{\downarrow} \setminus \{x\}$.

An arena is characterized by $(|A|, \vdash_A)$, and we will forget $\leq_A$. The set of the initial moves of A is denoted by A^i. The polarity of a move a is the parity of the length of the path from an initial move to a: Opponent (or O) for even length and Player (or P) for odd length. So that an initial move is always an Opponent move.

Constructions. If A and B are two arenas, we define:

- $A \times B = (|A| + |B|, \vdash_A + \vdash_B)$.
- $A+B = ((A^i \times B^i) + (A^i \times (|B| \setminus B^i)) + ((|A| \setminus A^i) \times B^i), \vdash_{A+B})$ where $\vdash_{A+B} (a_0, b_0)$, $(a_0, b) \vdash_{A+B} (a_0, b')$ if $b \vdash_B b'$ and $(a, b_0) \vdash_{A+B} (a', b_0)$ if $a \vdash_A a'$ (for $a_0 \in A^i$, $b_0 \in B^i$ and any a, a', b and b').
- $A \to B = ((|A| \times B^i) + |B|, \vdash_{A \to B})$ where $(a, b_0) \vdash_{A \to B} (a', b_0)$ if $a \vdash_A a'$, $b \vdash_{A \to B} b'$ if $b \vdash_B b'$ and $b_0 \vdash_{A \to B} (a, b_0)$ (for any $b_0 \in B^i$ and any a, a', b and b'). So that an initial move in $A \to B$ is an initial move in B and the polarity of a move (a, b_0) is the opposite of the polarity of a in A.

Definition 9 (Arena T)
The universal arena T is the countably deep tree in which each node is countably branching (*i.e.* sequences of natural numbers ordered with the prefix order).

A *justified sequence* s in A is a sequence of moves together with, for each occurrence of a non-initial move b, a pointer to an earlier occurrence of move a such that $a \vdash_A b$ (this entails that a justified sequence starts with an Opponent move). A *play* is a justified sequence in which the polarity of moves alternates. The set of the justified sequences (resp. plays) in A is denoted by $\mathcal{J}_A$ (resp. $\mathcal{P}_A$).

The *view* $\ulcorner s \urcorner$ of a play s is defined by $\ulcorner sa \urcorner = a$ if a is an initial move, $\ulcorner sa \urcorner = \ulcorner s \urcorner a$ if a is a Player move, and $\ulcorner satb \urcorner = \ulcorner s \urcorner ab$ if b is an Opponent move pointing to a.

Definition 10 (Strategy)
A *strategy* σ on A is a non-empty set of even-length plays of A which is closed under even prefixes and such that if $sab \in \sigma$ and $sac \in \sigma$ then $sab = sac$.

A strategy σ is *innocent* if

- for any $sab \in \sigma$, the pointer of b goes to a move in $\ulcorner sa \urcorner$;
- $sab \in \sigma$, $t \in \sigma$, $ta \in \mathcal{P}_A$ and $\ulcorner sa \urcorner = \ulcorner ta \urcorner$ entails $tab \in \sigma$.

If s is a justified sequence in $A \to B$, we denote by $s \restriction_A$ (resp. $s \restriction_B$) the justified subsequence of s containing only the moves in A (resp. B). If σ is a strategy on $A \to B$ and τ is a strategy on $B \to C$, the *composition* of σ and τ is the strategy on $A \to C$ defined by:

$$\sigma ; \tau = \{ s \restriction_{A \to C} \mid s \in \mathcal{J}_{(A \to B) \to C}, s \restriction_{A \to B} \in \sigma, s \restriction_{B \to C} \in \tau, s \restriction_{A \to C} \in \mathcal{P}_{A \to C} \}$$

where, in $s \restriction_{A \to C}$, the occurrence a of an initial move in A (which points to b in s) points to the occurrence c of an initial move of C on which b points in s.

If σ and τ are innocent then $\sigma ; \tau$ is innocent.

As described in [Lau05], we get the following categorical structure:

Proposition 6 (Game control category)
The category $\mathcal{G}$ with objects given by arenas and morphisms from A to B by innocent strategies on $A \to B$ is a control category (the cartesian product is given by $A \times B$ and the premonoidal product by $A + B$).

Proposition 7 (Game model)
T is a η-model of the $\lambda\mu$-calculus.

Proof: T is a η-reflexive object in $\mathcal{G}$ (as proved by Ker, Ong and Nickau [KNO02]), and we conclude with propositions 2 and 6. □

7 Conclusion

We have described three different models of the untyped $\lambda\mu$-calculus as a very first step in the study of denotational semantics of classical extensions of the untyped λ-calculus.

A lot of different directions have now to be covered, as for example:

- define a notion of *classical combinatory algebra* using the C combinator (typable of type $((A \to B) \to A) \to A)$) in the spirit of Hilbert's deduction system for classical logic;
- give a general study of $\lambda\mu$-algebras, of $\lambda\mu$-models, and of the corresponding reflexive objects in control categories;
- try to define variants of our intersection type systems (for example with subtyping) to deal with θ and η-reduction;
- apply all these models, in particular for realizability constructions.

Instead of our direct approach through control categories, the $\lambda\mu$-calculus can also be studied through the use of CPS translations [Plo75, HS02]. Here we do prefer to have a direct description of models (as control categories are important even if a representation theorem with respect to continuation models holds), but the CPS direction has to be investigated further.

As it is now well known since Selinger's work, there exists a syntactical duality between the call-by-name and the call-by-value $\lambda\mu$-calculi [Sel01, CH00]. This should help to develop the theory of these two

systems together, however the closed structure of control categories used to interpret functions in the call-by-name $\lambda\mu$-calculus is not the one used in the call-by-value case (the weak co-closed structure is used instead, see [Sel01]).

Thanks to Chantal Berline, Jean-Louis Krivine, Paul-André Melliès, and Paul Rozière for very helpful discussions on various topics related to this work.

References

[AJM00] Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409–470, December 2000.

[Bar84] Henk Barendregt. *The lambda calculus, its syntax and semantics.* Number 103 in Studies in Logic and the Foundations of Mathematics. North-Holland, second edition, 1984.

[BCDC83] Henk Barendregt, Mario Coppo, and Mariangiola Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *Journal of Symbolic Logic*, 48:931–940, 1983.

[Ber78] Gérard Berry. Stable models of typed lambda-calculi. In *International Colloquium on Automata, Languages and Programming*, volume 62 of *Lecture Notes in Computer Science*. Springer, 1978.

[CDCV81] Mario Coppo, Mariangiola Dezani-Ciancaglini, and Betti Venneri. Functional characters of solvable terms. *Zeitschrift für Mathematische Logik*, 27:45–58, 1981.

[CH00] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. In *Proceedings of the International Conference on Functional Programming*, volume 35(9) of *ACM SIGPLAN Notices*, pages 233–243. Association for Computing Machinery, ACM Press, September 2000.

[Gir86] Jean-Yves Girard. The system F of variable types, fifteen years later. *Theoretical Computer Science*, 45:159–192, 1986.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[Gir91] Jean-Yves Girard. A new constructive logic: classical logic. *Mathematical Structures in Computer Science*, 1(3):255–296, 1991.

[Gri90] Timothy Griffin. A formulae-as-types notion of control. In *Proceedings of the 17th ACM SIGPLAN-SIGACT symposium on Principles of Programming Languages*, pages 47–58. Association for Computing Machinery, ACM Press, 1990.

[Har99] Russell Harmer. *Games and Full Abstraction for Nondeterministic Languages.* Ph.D. thesis, Imperial College and University of London, 1999.

[HO00] Martin Hyland and Luke Ong. On full abstraction for PCF. *Information and Computation*, 163(2):285–408, December 2000.

[HS02] Martin Hofmann and Thomas Streicher. Completeness of continuation models for lambda-mu-calculus. *Information and Computation*, 179(2):332–355, December 2002.

[KNO02] Andrew Ker, Hanno Nickau, and Luke Ong. Innocent game models of untyped lambda calculus. *Theoretical Computer Science*, 272:247–292, 2002.

[Kri93] Jean-Louis Krivine. *Lambda-calculus, types and models.* Ellis Horwood, 1993.

[Kri01] Jean-Louis Krivine. Typed lambda-calculus in classical Zermelo-Fraenkel set theory. *Archive for Mathematical Logic*, 40(3):189–205, 2001.

[Lai97] James Laird. Full abstraction for functional languages with control. In *Proceedings of the twelfth annual symposium on Logic In Computer Science*, pages 58–67, Warsaw, June 1997. IEEE, IEEE Computer Society Press.

[Lai98] James Laird. *A semantic analysis of control*. Ph.D. thesis, University of Edinburgh, 1998.

[Lau02] Olivier Laurent. Polarized games (extended abstract). In *Proceedings of the seventeenth annual symposium on Logic In Computer Science*, pages 265–274, Copenhagen, July 2002. IEEE, IEEE Computer Society Press.

[Lau03] Olivier Laurent. Polarized proof-nets and $\lambda\mu$-calculus. *Theoretical Computer Science*, 290(1):161–188, January 2003.

[Lau05] Olivier Laurent. Syntax vs. semantics: a polarized approach. *Theoretical Computer Science*, 343(1–2):177–206, October 2005.

[LGD03] Pierre Lescanne, Silvia Ghilezan, and Daniel Dougherty. Types avec intersection dans une extension de $\overline{\lambda}\mu\tilde{\mu}$. Slides of a talk in Paris. Available at `http://perso.ens-lyon.fr/pierre.lescanne/PUBLICATIONS/lac.pdf`, September 2003.

[LR03] Olivier Laurent and Laurent Regnier. About translations of classical logic into polarized linear logic. In *Proceedings of the eighteenth annual symposium on Logic In Computer Science*, pages 11–20, Ottawa, June 2003. IEEE, IEEE Computer Society Press.

[McC96] Guy McCusker. *Games and Full Abstraction for a Functional Metalanguage with Recursive Types*. Ph.D. thesis, Imperial College and University of London, 1996. Published in Springer-Verlag's Distinguished Dissertations in Computer Science series, 1998.

[Nic94] Hanno Nickau. Hereditarily sequential functionals. In Anil Nerode and Yuri Matiyasevich, editors, *Logical Foundations of Computer Science*, volume 813 of *Lecture Notes in Computer Science*, pages 253–264. Springer, 1994.

[Ong96] Luke Ong. A semantic view of classical proofs: type-theoretic, categorical, denotational characterizations. In *Proceedings of the eleventh annual symposium on Logic In Computer Science*, pages 230–241, New Brunswick, July 1996. IEEE, IEEE Computer Society Press.

[Par92] Michel Parigot. $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In *Proceedings of International Conference on Logic Programming and Automated Reasoning*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992.

[Plo75] Gordon Plotkin. Call-by-name, call-by-value and the λ-calculus. *Theoretical Computer Science*, 1:125–159, 1975.

[PR97] John Power and Edmund Robinson. Premonoidal categories and notions of computation. *Mathematical Structures in Computer Science*, 7(5):453–468, October 1997.

[Sco72] Dana Scott. Continuous lattices. In Lawvere, editor, *Toposes, Algebraic Geometry and Logic*, volume 274 of *Lecture Notes in Mathematics*, pages 97–136. Springer, 1972.

[Sel01] Peter Selinger. Control categories and duality: on the categorical semantics of the lambda-mu calculus. *Mathematical Structures in Computer Science*, 11(2):207–260, April 2001.