Université Paris-Nord

Mémoire d'habilitation à diriger des recherches

Spécialité : Informatique

Lambda Calculus, Linear Logic and Symbolic Computation

Giulio Manzonetto

Présenté aux rapporteurs :

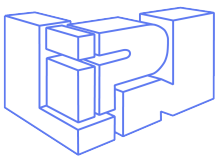
Jean Goubault-Larrecq	CNRS & ENS de Cachan, France
Martin Hyland	King's College, Royaume-Uni
Jan-Willem Klop	Vrije Universiteit, Pays-Bas

afin d'être soutenu devant la commission d'examen formée de :

Henk Barendregt	Radboud University, Pays-Bas
Christophe Fouqueré	Univ. Paris-Nord, France
Mai Gehrke	CNRS & Univ. Paris-Diderot, France
Jean Goubault-Larrecq	CNRS & ENS de Cachan, France
Stefano Guerrini	Univ. Paris-Nord, France
Martin Hyland	King's College, Royaume-Uni
Delia Kesner	Univ. Paris-Diderot, France

Laboratoire d'Informatique de Paris-Nord (LIPN), UMR CNRS 7030
IUT de Villetaneuse et Institut Galilée - Université Paris 13 SPC

UNIVERSITÉ PARIS 13



Thinking about λ-calculus...

Painting by Laura Fontanella, from a picture taken by Paolo Tranquilli in 2011, at the Rocky Mountains, Canada.

Short contents

Contents

Preface

This document is a synthesis of the research I carried out in the past eight years and is part of my dossier to obtain the *habilitation à diriger les recherches*. My research interests lie at the interface between computer science and mathematical logic. More precisely, my research focuses on the theory of programming languages, and in particular on the λ-calculus and its typed, non-deterministic and resource sensitive extensions. Most of my results have been obtained in collaboration with other researchers, and have been influenced by the stimulating environments where I was working. I will therefore present a quick overview of my previous academic positions and of the research teams I was a member of.

I defended my PhD thesis on equational theories and denotational models of the untyped λ-calculus in February 2008. After one additional year spent at the University Paris-Diderot as *Attaché Temporaire d'Enseignement et de Recherche*, I worked as an INRIA postdoc in the Moscova project-team leaded by Jean-Jacques Lévy. Subsequently, I spent six months working as a postdoc in the laboratory LIPN of the Université Paris-Nord, and one year and a half as a member of the ICIS team, at the Radboud University of Nijmegen. In the Netherlands, I was the principal investigator of the NWO project *Calmoc*, and I had the honour of working with Henk Barendregt and Mai Gehrke. During my years of postdoc, I enjoyed a priceless freedom that allowed me to work on a wide range of topics, including resource sensitive extensions of λ-calculus, several kinds of type systems, universal algebra and categorical semantics.

In September 2011, I have been employed as a *Maître de Conférences* at the University Paris-Nord. Since then, I am a permanent member of the team *Logique, calcul, raisonnement* within the laboratory LIPN, and of the department *Réseaux et Telecom*, IUT de Villetaneuse.

I recently had the opportunity of supervising with Stefano Guerrini a PhD student, Domenico Ruoppolo, who defended his thesis on December 13, 2016. Ruoppolo worked on relational models of λ-calculus in connection with Morris's original observational theory and the ω-rule. I also supervised a number of master students and postdocs on various topics related to λ-calculus and categorical semantics.

This manuscript was written during a one year *délégation CNRS* spent at the laboratory IRIF of the Université Paris-Diderot. I decided to keep a separate bibliography (Page 119) containing all my articles in order to give a global view of my research.

Giulio Manzonetto,
January 1, 2017, Paris

Acknowledgements

Behind each of my articles there is a story made of days spent in front of a whiteboard with colleagues and friends, calculations handwritten on paper towels at the cafeteria, scientific trips around the world, and so on. I wish to express here my gratitude to all the people I had the opportunity to work with, since from each of them I learnt so many things and they helped me to become the researcher I am today.

Heartfelt thanks to Martin Hyland, Jan-Willem Klop, and Jean Goubault-Larrecq for accepting the task of reviewing this manuscript and for their attentive reading. In particular, I am grateful to Jean for bringing to my attention a strong connection between our technique for algebraizing logics and the theory of binary/multi-valued decision diagrams.

I am also deeply indebted to Stefano Guerrini, who agreed to be my godfather for this *habilitation à diriger des recherches*. Together with Stefano I have supervised my first PhD student Domenico Ruoppolo, that I wish to thank.

I express all my gratitude to the LIPN for providing all the means and the support necessary to work out my research, and IRIF for hosting me during a one-year délégation CNRS. Thanks to all my colleagues of the University Paris-Nord and Paris-Diderot for the friendly and stimulating environment.

A special thank, and all my love, goes to Laura, who encouraged me during all these years, and to my unborn daughter who gave me the right stimulus to finalize this manuscript in a reasonable amount of time.

Introduction

The purpose of this manuscript is to give an overview of the results we obtained, in collaboration with other colleagues, from 2008 to 2016. In order to put things in perspective and provide some context, we also review some literature that inspired us and we explain some of the intuitions that are behind our work.

As a matter of presentation, we have regrouped such results into four research axes that are however deeply interconnected[1].

1. The Lambda Calculus and its Type Disciplines

The λ-calculus is a foundational "idealized programming language" that has been extensively studied in theoretical computer science [Bar84]. When focusing on the equivalences between programs, rather than on the process of computation, particular congruences called *λ-theories* become the main object of study. Certain λ-theories are particularly interesting because they arise as *observational equivalences* — this means that two programs M and N are considered equivalent whenever it is possible to plug them into any context $C(\!|-|\!)$ without noticing any difference in the global behaviour: in other words, $C(\!|M|\!)$ produces a result exactly when $C(\!|N|\!)$ does. Therefore observational equivalences depend on the notion of *result*, also called *observable*, we are interested in. The λ-theory $\mathcal{H}^*$ is by far the most famous and well studied observational equivalence, and it is obtained by considering as observables the head normal forms, that represent sufficient stable amount of information coming out of the computation [Hyl75, Wad76, Hyl76, Gou95b, DFH99, M17].

In [M9], we rather focused on the λ-theory $\mathcal{H}^+$ which is obtained by considering completely defined β-normal forms as observables [Mor68, Lév76, CDZ87, RP04, Lév05, M21]. On the syntactic side, we proved that $\mathcal{H}^+$ satisfies a strong form of extensionality, known as "the ω-rule", which has been extensively studied in the literature in connection with several λ-theories [IS09, Bar71, Plo74, BBKV78, IS04]. This result is somehow expected, but the proof is non trivial and brings us closer to the solution of a conjecture, formulated by Sallé in the eighties, stating that $\mathcal{H}^+$ cannot be obtained just by adding the ω-rule to the λ-theory equating all terms having the same Böhm tree. On the semantic side, we provided a characterization of all relational graph models having as theory exactly $\mathcal{H}^+$. This result was inspired by Breuvart's PhD thesis [Bre15], where he gave sufficient and necessary conditions for a *K-model*[2] to induce as theory $\mathcal{H}^*$. All previous results concerning $\mathcal{H}^+$ and $\mathcal{H}^*$ were much weaker, since researchers were only able to provide individual models or identify sufficient conditions for a model to induce one of those theories.

Concerning typed λ-calculi [BDS13], we found in [M30] a perhaps unexpected connection between two major undecidability results. In [Loa01], Loader studied the models of simply typed λ-calculus with one ground type and proved that, given the full model $\mathcal{F}$ over a finite set, the question whether some element $f \in \mathcal{F}$ is λ-definable is undecidable. In [Urz99], Urzyczyn studied the intersection type system based on countably many atoms and proved that it is undecidable to determine whether a type is inhabited. We have shown that these two results of undecidability follow from each other in a natural way, by interpreting intersection types as continuous functions logically related to elements of $\mathcal{F}$.

[1] In particular, techniques like the Taylor expansion coming from the differential or resource calculus (Chapter 2) have been used for studying the untyped λ-calculus (Chapter 1) as well.

[2] Using the terminology of [Ber00], Krivine's K-models constitute a particular subclass of continuous models of λ-calculus. For instance, Scott's model $\mathcal{D}_\infty$ can be presented as a K-model.

From this, and a result by Joly [Jol03] on λ-definability, we get that Urzyczyn's theorem also holds for intersection types with at most two atoms.

Concerning second order type systems, we proved in [M28] a strong normalization result for ML^F, an extension of ML with first-class polymorphism as in system F [Gir72]. The proof of this result was achieved in several steps. We first focused on xML^F, the Church-style version of ML^F, and showed that it can be translated into a calculus of coercions: terms are mapped into terms and instantiations into coercions. This coercion calculus can be seen as a decorated version of system F, so that the simulation result entails strong normalization of xML^F through the same property of system F. We then transferred the result to all other versions of ML^F using the fact that they can be compiled into xML^F and showing that there is a bisimulation between the two.

2. The Resource Calculus and its Denotational Semantics

The λ-calculus is not resource conscious, in the sense that a λ-term can erase or duplicate its arguments an arbitrary number of times during its reduction. Inspired by the quantitative semantics of linear logic, Ehrhard defined the *differential λ-calculus* [ER03], that is a nondeterministic extension of λ-calculus with a syntactic derivative operator that allows to improve the control over the consumption of resources. Building on these insights, and inspired by Boudol's λ-calculus with multiplicities [Bou93], Tranquilli designed the *resource calculus* where the derivative operator is replaced by a linear application of a term to a "bag" (multiset) of resources, and reusable resources are annotated with an explicit promotion $(\cdot)^!$. Both calculi have their own interest, but they can be also used to infer properties of the regular λ-calculus and of nondeterministic calculi through the *Taylor expansion* originally defined by Ehrhard and Regnier in [ER03]. The idea behind this expansion is to expose the amount of resources possibly used by a λ-term M during its execution. This is done by transforming M into a power series of linear terms that approximate its behaviour.

We studied the resource calculus both from a syntactic and from a semantic perspective. On the syntactic side, we have shown in [M19] that the resource calculus satisfies a semi-separability result that can be seen as a reformulation of the Böhm Theorem [Böh68]. On the semantic side, starting from the work of Blute, Cockett and Seely [BCS09], we proposed the notions of Cartesian closed differential categories [M5] and linear reflexive objects [M18] living in such categories as models of the simply typed and of the untyped resource calculus, respectively. These notions are general enough to encompass all models of resource calculus that have been individually introduced in the literature [HNPR06, dC07, M3], and are *equationally complete* under certain hypotheses. In particular, the relational semantics can be seen as a Cartesian closed differential category and the relational graph models induce linear reflexive objects. Therefore, it is natural to wonder whether the relational graph model[3] $\mathcal{D}_\omega$ introduced in [M3], which we proved to be fully abstract for the regular λ-calculus [M17], is also fully abstract for the resource calculus. In [Bre13], Breuvart showed that this is not the case by providing an ingenious counterexample. However, we were able to demonstrate in [M1, M2] that $\mathcal{D}_\omega$ is actually fully abstract for a resource calculus extended with a convergency test mechanism first arisen in the context of differential interaction nets [EL10].

[3]Notice that $\mathcal{D}_\omega$ is not Scott's pioneering model $\mathcal{D}_\infty$, but rather its relational version.

3. Nondeterminism in the Quantitative Setting

The resource calculus is intrinsically nondeterministic because a resource term needs to choose nondeterministically how to use the different resources contained in its bags. This opens the way to study differential categories as models of nondeterministic languages as well. For instance, since in the relational semantics programs are interpreted as relations, and relations are closed under arbitrary unions, it is natural to interpret nondeterministic choice in that context as set-theoretical union. As the union of two relations is non-empty whenever at least one of these relations is non-empty, this approach captures a notion of *may-convergency*: the nondeterministic choice between two terms converges if at least one of the two terms does. In particular reflexive objects, like $\mathcal{D}_\omega$, an interpretation of the parallel composition is also at hand and can be obtained by combining the *mix-rule* of linear logic with the contraction rule, like Danos and Krivine did in [DK00]. This corresponds to a notion of *must-convergency*: the parallel composition between two terms converges if both terms converge. We have shown in [M6] that $\mathcal{D}_\omega$ is an adequate model of a call-by-name λ-calculus extended with nondeterministic choice and parallel composition, and in [M12] that this approach generalizes to the call-by-value setting. In both cases, the models that we found are adequate but not fully abstract — this is due to the fact that the observational equivalence is not resource conscious while the semantics of parallel-composition is.

In order to obtain a full abstraction result, we need to consider resource sensitive extensions of typed λ-calculi with constants. Keeping this purpose in mind, we designed *Resource* PCF, a PCF-like language endowed with a linear-head reduction, and proved that its relational semantics is fully abstract. This result is actually a consequence of a more general construction we described in [M13, M14] to build a differential category from an arbitrary symmetric monoidal closed category $\mathbf{C}$. The key steps are the following: we first perform the free sup-lattice enrichment of $\mathbf{C}$ and add freely countable biproducts, if needed, then we apply the Karoubi envelope and finally we consider the co-Kleisli category. This method is general enough to reconstruct known examples like the relational semantics of linear logic [Gir88] and the category of nondeterministic games defined by Harmer and McCusker in [HM99], and allows to expose their differential structure.

In [M15], we started from the consideration that the category of sets and relations can be seen as the biproduct completion of the Boolean ring of truth values. Inspired by the work done in [M13, M14], we generalized this construction to an arbitrary continuous semiring $\mathcal{R}$, producing a cpo-enriched category which is a semantics of linear logic. We have shown that its co-Kleisli category is an adequate model of an extension of PCF, parametrized by the continuous semiring $\mathcal{R}$: terms in this extended language can be instrumented by elements of $\mathcal{R}$, leading to an operational notion of reduction weighted by values in $\mathcal{R}$. Thus the choice of $\mathcal{R}$ and of how terms are instrumented allowed us to model, both operationally and denotationally, a range of quantitative properties of program execution. For instance, we have shown that specific instances of $\mathcal{R}$ allow to compare programs not only with respect to “what they can do", but also “in how many steps" or “in how many different ways" (for nondeterministic PCF) or even “with what probability" (for probabilistic PCF).

4. Factor Algebras and Symbolic Computation

Algebraic logic investigates the connections between a logic and algebraic properties of the corresponding class of algebras. The origin of modern algebraic logic goes back to Tarski’s 1935 paper [Tar35], where he established the correspondence between classical

propositional logic and cylindric algebras. Subsequently, a number of different logics were algebraized in this way, like the intuitionistic logic and the multi-valued logics of Post, of Gödel and of Łukasiewicz. The idea is to consider a logical formula as an algebraic term, the tautologies being those expressions that can be proven equivalent to "true". This approach is heterogeneous because for each logic one needs to find the corresponding kind of algebras, study their properties, and find *ad hoc* methods for proving equivalences between terms using their axioms.

In the pioneering paper [M29], we proposed a unifying method for determining whether a propositional formula ϕ is a tautology. This approach is general enough to be applicable to any multi-valued matrix logic $\mathcal{L}$ with truth values $v_1, \ldots, v_n$ among which there is a designated element t representing the truth value "verum". The idea is to define a translation mapping formulas into terms of a factor algebra. Both truth values and propositional variables, that are static objects in the logic $\mathcal{L}$, become dynamic entities after the translation: truth values become fresh algebraic variables $\xi_1, \ldots, \xi_n$ that can receive substitutions; a propositional variable P becomes an n-ary decomposition operator f_P whose behaviour is axiomatized by three simple equations; connectives are implemented via substitutions and logical operations on the indices of the variables ξ_i. Our main theorem states that a formula ϕ is a tautology exactly when its translation is provable equal to ξ_{t} using the axioms of a factor variety. We then showed that this approach generalizes to the model theory of quantified matrix logics, and of first order classical logic (with or without equality) for which we were able to provide a completeness theorem.

The theory of decomposition operators was previously applied to study the denotational models of λ-calculus [M23, M25] and lattices of equational theories [M24].

Preliminaries

We fix some basic notions and notations that will be used in the rest of the manuscript. Some symbol will be overloaded, but the reader should always be able to understand its meaning from the context. A comprehensive table of symbols is given on Page 97.

Sets and multisets. We denote by $\mathbb{N}$ the set of natural numbers and by $\mathbb{R}^+$ the set of positive real numbers. Given a set A, we denote by $|A|$ its cardinality and by $\mathcal{P}(A)$ its powerset. We write $A \subseteq_{\mathrm{f}} B$ whenever A is a finite subset of B.

A *multiset* over A is a partial map $a : A \to \mathbb{N}\backslash\{0\}$ and its *support* is its domain $\mathrm{dom}(a)$. For each $\alpha \in A$, $a(\alpha)$ gives the *multiplicity of* α *in* a, that is the number of occurrences of α in a. Given two multisets a_1 and a_2 over A, their *multiset union* $a_1 \uplus a_2$ is defined as the pointwise sum $(a_1 \uplus a_2)(\alpha) = a_1(\alpha) + a_2(\alpha)$. A multiset a is called *finite* if it has a finite support. An infinite sequence $(a_i)_{i\in\mathbb{N}}$ of finite multisets over A is called *quasi-finite* whenever a_i is non-empty for finitely many indices i. We write $\mathcal{M}_{\mathrm{f}}(A)$ for the set of all finite multisets over A and $\mathcal{M}_{\mathrm{f}}(A)^{(\omega)}$ for the set of all quasi-finite sequences of finite multisets over A.

We will systematically represent a multiset a as an unordered list $[\alpha_1, \alpha_2, \dots, \alpha_i, \dots]$ possibly with repetitions. In particular, the empty multiset will be represented by $[]$.

Category theory. We generally use the notation of [Mel09] for category theory. Given a category $\mathbf{C}$ and two objects A, B we denote by $\mathbf{C}(A, B)$ or $\mathrm{Hom}(A, B)$ the corresponding homset and by $f, g, h, \dots$ its elements. We write the identity morphism on A as Id^A, or simply A. Composition is written using infix ; in diagrammatic order.

In a *symmetric monoidal category* (*smc*) $\mathbf{C}$, we denote by $\otimes$ the tensor product and by 1 its unit. When $\mathbf{C}$ is monoidal closed (*smcc*), the monoidal exponential object is denoted as $A \multimap B$. We use $\mathrm{ev}_{A,B} \in \mathbf{C}((A \multimap B) \otimes A, B)$ for the monoidal evaluation morphism and $\lambda(f) \in \mathbf{C}(A, B \multimap C)$ for the monoidal Currying of a morphism $f \in \mathbf{C}(A \otimes B, C)$.

When $\mathbf{C}$ is moreover $\star$-autonomous with respect to a dualizing object $\bot$, we indicate by $A^\bot$ the dual object $A \multimap \bot$. We will often elide the associativity and unit isomorphisms associated with monoidal categories.

In a *Cartesian category* $\mathbf{C}$, we write T for the terminal object and T^A for the unique morphism in $\mathbf{C}(A, \mathsf{T})$. We use $A \times B$ to denote the product of A and B, $\langle f, g\rangle$ for the pairing of maps $f \in \mathbf{C}(A, B)$ and $g \in \mathbf{C}(A, C)$, and π_1, π_2 for the corresponding projections. In presence of biproducts $A \oplus B$, we denote by $[f, g]$ the copairing of $f \in \mathbf{C}(A, C)$ and $g \in \mathbf{C}(B, C)$ and by ι_1, ι_2 the corresponding injections.

When $\mathbf{C}$ is moreover Cartesian closed (*ccc*), we denote the exponential object by $A \to B$, the evaluation map by $\mathrm{Eval}^{A,B} \in \mathbf{C}((A \to B) \times A, B)$ and the Currying of $f \in \mathbf{C}(A \times B, C)$ by $\Lambda(f) \in \mathbf{C}(A, B \to C)$.

Categorical semantics of linear logic. We now describe in a nutshell the categorical semantics of *linear logic* as formulated in Lafont's thesis [Laf88], that is as *Lafont categories*. This is not the most general definition of a model of linear logic, but it has the advantage of being simple and general enough to encompass all the models we will use in the rest of the manuscript. Our main reference for categorical models of linear logic is the survey paper by Melliès [Mel09].

Recall that an object A of an smcc $\mathbf{C}$ is a *comonoid* if it is equipped with a multiplication $\mathrm{c} \in \mathbf{C}(A, A \otimes A)$ and a unit $\mathrm{w} \in \mathbf{C}(A, 1)$ satisfying the usual associativity and unit equa-

tions. A *comonoid morphism* f from $(A_1, \mathrm{c}_1, \mathrm{w}_1)$ to $(A_2, \mathrm{c}_2, \mathrm{w}_2)$ is defined as a morphism $f \in \mathbf{C}(A_1, A_2)$ such that the following two diagrams commute:

A symmetric monoidal closed category $\mathbf{C}$ is a *Lafont category* whenever:

(i) it has finite products and,

(ii) for every object A, there exists an object $!A$ being the free commutative comonoid generated by A.

Condition (ii) is equivalent to ask that the forgetful functor $U : \mathbf{C}^{\otimes} \to \mathbf{C}$, where $\mathbf{C}^{\otimes}$ denotes the category of commutative comonoids and comonoid morphisms in $\mathbf{C}$, has a right adjoint F, and that $! := F; U$ is the comonad over $\mathbf{C}$ of this adjunction.

Unfolding this definition, one gets that for every objet A, there is an object $!A$ endowed with a commutative comonoid structure:

$$\mathrm{contr}^A \in \mathbf{C}(!A, !A \otimes !A), \qquad \mathrm{weak}^A \in \mathbf{C}(!A, 1),$$

and a morphism $\mathrm{der}^A \in \mathbf{C}(!A, A)$ satisfying the following universality property — for every commutative comonoid B and for every morphism $f \in \mathbf{C}(B, A)$ there exists a unique comonoid morphism $f^\dagger \in \mathbf{C}(B, !A)$ satisfying $f^\dagger; \mathrm{der}^A = f$. The multiplication and the unit of $!A$ are called respectively *contraction* and *weakening*, while der is called *dereliction*.

Every Lafont category $\mathbf{C}$ is equipped with a comonad $(!, \mathrm{der}, \mathrm{dig})$ defined as follows:

- the endofunctor $!$ sends every object A into the free commutative comonoid $!A$ and every morphism $f \in \mathbf{C}(A, B)$ into $(\mathrm{der}^A; f)^\dagger \in \mathbf{C}(!A, !B)$,
- the multiplication is called *digging* and defined as $\mathrm{dig}^A := (\mathrm{Id}^{!A})^\dagger \in \mathbf{C}(!A, !!A)$,
- the unit is the morphism $\mathrm{der}^A \in \mathbf{C}(!A, A)$ given above.

Moreover, the functor $!$ is equipped with a monoidal structure turning it into a symmetric monoidal functor from the smc $(\mathbf{C}, \otimes)$ to the smc $(\mathbf{C}, \times)$: the corresponding two isomorphisms are given by

$$\mathrm{m}^{\mathsf{T}} := (\mathsf{T}^1)^\dagger \in \mathbf{C}(1, !\mathsf{T}), \quad \mathrm{m}^{A,B} := \langle (\mathrm{der}^A \otimes \mathrm{weak}^B), (\mathrm{weak}^A \otimes \mathrm{der}^B) \rangle^\dagger \in \mathbf{C}(!A \otimes !B, !(A \times B)).$$

The (co)Kleisli $\mathbf{C}_!$ over the comonad $(!, \mathrm{dig}, \mathrm{der})$ is defined as the category having the same objects of $\mathbf{C}$, while the homset is given by $\mathbf{C}_!(A, B) := \mathbf{C}(!A, B)$. The composition in $\mathbf{C}_!$ is denoted by $;_!$ and defined by $f ;_! g := \mathrm{dig}; !f; g$. The identities in $\mathbf{C}_!$ are given by $A := \mathrm{der}^A$.

It is well known that the Kleisli category $\mathbf{C}_!$ of a Lafont category $\mathbf{C}$ is Cartesian closed: indeed, the structure of cartesian smcc of $\mathbf{C}$ is lifted to a Cartesian closed structure in $\mathbf{C}_!$ by the m's isomorphisms. The exponential object $A \to B$ is defined as $!A \multimap B$ and the morphism $\mathrm{Eval}^{A,B} \in \mathbf{C}_!((A \to B) \times A, B)$ is given by

$$(\mathrm{m}^{!A \multimap B, A})^{-1}; (\mathrm{der}^{!A \multimap B} \otimes !A); \mathrm{ev}^{!A,B}.$$

This defines an exponentiation since for every $f \in \mathbf{C}_!(C \times A, B)$ there is a unique morphism $\Lambda(f) := \lambda(\mathrm{m}^{C,A}; f) \in \mathbf{C}_!(C, A \to B)$ satisfying $\Lambda(f) \times A = f$.

1 *The Lambda Calculus and its Type Disciplines*

In which we show that a quarter of century after the publication of Barendregt's bible on the λ-calculus, there are still interesting problems to solve and new connections to find.

- **Loader and Urzyczyn are Logically Related.**
  S. Salvati, G. Manzonetto, M. Gehrke and H.P. Barendregt.
  Automata, Languages and Programming - 39th International Colloquium (ICALP'12), Proceedings, Part II, ed. A. Czumaj et al., Lecture Notes in Computer Science, Volume 7392, pages 364-376, Springer, 2012.
- **Relational Graph Models, Taylor Expansion and Extensionality.**
  G. Manzonetto and D. Ruoppolo.
  Mathematical Foundations of Programming Semantics XIV (MFPS'14), Electronic Notes in Theoretical Computer Science, Vol. 308, pages 245-272, 2014.
- **New Results on Morris's Observational Theory.**
  F. Breuvart, G. Manzonetto, A. Polonsky and D. Ruoppolo.
  In Proceedings of Formal Structures for Computation and Deduction (FSCD 2016), LIPIcs Vol. 52, pages 15:1-15:18, 2016.
- **Strong normalization of ML^{F} via a calculus of coercions.**
  G. Manzonetto and P. Tranquilli.
  Theoretical Computer Science, Volume 417, pages 74–94, 2012.

My PhD thesis mainly focused on denotational models and equational theories of the untyped λ-calculus. Subsequently, I broadened my scientific interests by considering non-deterministic languages, resource sensitive extensions, first-order and second-order type systems, and intersection types. However, I still think that the study of λ-calculus remains central in theoretical computer science, as several proof techniques originally developed for this system can be exported to other languages and frameworks.

In this chapter I survey the most important properties of λ-calculus, and I present some results of mine. During my post-doc at INRIA (2010) I studied with Lévy the normalization of system F. Together with Tranquilli, I proved a strong normalization result for ML^{F}, a functional programming language extending ML with first-class polymorphism.

During my postdoc in Nijmegen (2011), I worked with Barendregt and Gehrke on a connection between the λ-definability problem of simply typed λ-calculus and the inhabitation problem of intersection types newly discovered by Salvati. The four of us together, proved that the two problems are equidecidable: the undecidability of the former follows from the undecidability of the latter, and *vice versa*.

In 2012, together with my PhD student Ruoppolo, I started analyzing Morris's original extensional observational theory both from a syntactic and from a semantic point of view. In an extended collaboration with Breuvart and Polonsky, we provided a characterization of all relational graph models that are fully abstract for Morris's theory. Moreover, we answered positively the question whether such a theory satisfies the ω-rule.

1.1 THE LAMBDA CALCULUS IN A NUTSHELL

The λ-*calculus* was introduced by Church around 1930 as the kernel of an investigation in the foundation of mathematics and logic, where the notion of function instead of set was taken as primitive. Subsequently, it became a key tool in the study of computability and, with the rise of computers, the formal basis of the functional programming paradigm. Today, the λ-calculus plays an important role as a bridge between logic and computer science, which explains the general interest in this formalism among computer scientists.

In this section we present the main notions and results concerning the λ-calculus that will be useful in the rest of the manuscript. We also profit from the occasion to fix some notations, even if we generally use the ones from Barendregt's book [Bar84].

The syntax. A beautiful aspect of the λ-calculus is that its syntax is extremely simple and, despite that, it is powerful enough to represent all partial computable functions. The set Λ of λ-terms over an infinite set Var of variables is defined by the following grammar:

$$\Lambda: \quad M, N, P, Q ::= \; x \;\mid\; \lambda x.M \;\mid\; MN \qquad \text{for all } x \in \mathrm{Var}.$$

For the sake of simplicity we consider λ-terms up to α-conversion [Bar84, Def. 2.1.11], and we often focus on *closed* λ-terms, also called *combinators*, that are λ-terms M whose set of *free variables* $\mathrm{FV}(M)$ is empty. It should be understood that all results presented for closed λ-terms can be generalized to arbitrary terms. The set of closed λ-terms is denoted by Λ^o.

Concerning specific combinators, we consider fixed:

$$\begin{array}{lll} \mathrm{I} := \lambda x.x & \Omega := (\lambda x.xx)(\lambda x.xx) & \mathrm{Y} := \lambda f.(\lambda x.f(xx))(\lambda x.f(xx)) \\ \mathrm{K} := \lambda xy.x & \mathrm{K}' := \lambda xy.y & \mathrm{J} := \mathrm{Y}(\lambda jxy.x(jy)) \end{array}$$

where I is the identity, K and K$'$ are the first and second projection, Ω is the paradigmatic looping λ-term, and Y is Curry's fixed point combinator. The combinator J will play a central role in Section 1.2 in connection with the notion of "infinite η-expansions".

Operational semantics. The main rewriting rule of the λ-calculus is the β-*reduction*:

$$(\beta) \quad (\lambda x.M)N \to_\beta M\{N/x\}$$

where $M\{N/x\}$ denotes the capture-free simultaneous substitution of N for all free occurrences of x in M. In general, the λ-calculus is an intensional language: this means that there are different λ-terms having the same extensional behaviour. We are sometimes interested in considering the extensional version of λ-calculus obtained by adding the η-*reduction*:

$$(\eta) \quad \lambda x.Mx \to_\eta M \text{ provided } x \notin \mathrm{FV}(M).$$

We write $\to_{\beta\eta}$ for $\to_\beta \cup \to_\eta$. Given a reduction $\to_{\mathtt{R}}$, the *multistep* $\mathtt{R}$-*reduction* $\twoheadrightarrow_{\mathtt{R}}$ (resp. the $\mathtt{R}$-*conversion* $=_{\mathtt{R}}$) is defined as its transitive-reflexive (and symmetric) closure.

The first studies on the λ-calculus concerned its rewriting theory. The system was proved to be confluent by Church and Rosser [CR36], a property which implies the uniqueness of the β-normal forms and ultimately the consistency of the calculus. Subsequently, λ-terms were investigated from the point of view of their capability of interaction with the environment. The question is whether, given a closed λ-term M, there is a sequence of arguments $P_1, \ldots, P_n \in \Lambda^o$ that are able to transform M into some other closed λ-term N:

$$MP_1 \cdots P_n =_\beta N \tag{1.1}$$

More generally, researchers undertook a quest for solutions of systems of equations between closed λ-terms: $M_1\vec{P}_1 =_\beta N_1 \wedge \cdots \wedge M_k\vec{P}_k =_\beta N_k$. Clearly, a system of equation of this form is not always satisfiable and the problem of whether a system is satisfiable can be difficult. For this reason Böhm restricted his attention to M's and N's in β-normal forms and systems having only two equations. This kind of investigations led him to the following definition of separability: two closed λ-terms M, N are called *separable* if there exist $\vec{P} \in \Lambda^o$ such that $M\vec{P} =_\beta \mathrm{K}$ and $N\vec{P} =_\beta \mathrm{K}'$. The first and second projections were chosen because they are very different β-normal forms that cannot be consistently equated.

The Böhm Theorem, which is a fundamental result in λ-calculus, states that all η-distinct β-normal forms can be separated.

Theorem 1.1.1 (Böhm Theorem [Böh68])**.**
If M and N are two distinct $\beta\eta$-normal closed λ-terms, then there exist $P_1, \ldots, P_n \in \Lambda^o$ such that:

$$MP_1 \cdots P_n =_\beta \mathrm{K} \qquad \textit{and} \qquad NP_1 \cdots P_n =_\beta \mathrm{K}'$$

Solvability. The Böhm Theorem fits in an early tradition that tends to divide the closed λ-terms into two classes: those with normal forms, whose "values" are perfectly defined, and those without normal form which are regarded as "undefined". In the seventies Barendregt realized that such a division was too discrete: there are λ-terms, like Ω, that are completely undefined and therefore unable of any interaction with the environment, but there are also λ-terms that are defined in some respects but not in others, which are nevertheless capable of such an interaction.

To determine whether a closed λ-term M belongs to the first or the second class of undefined terms, Barendregt identified the following test [Bar71]: M is *solvable* if there exist $P_1, \ldots, P_n \in \Lambda^o$ such that $MP_1 \cdots P_n =_\beta \mathrm{I}$; M is called *unsolvable*, otherwise. Notice that the equation characterizing solvability is just another instance of (1.1) where N is set to I. An important result due to Wadsworth is the fact that solvable terms can be characterized from an operational point of view as those terms M having a *head normal form* (*hnf*), that is such that $M =_\beta \lambda x_1 \ldots x_n.x_i M_1 \cdots M_k$ for some $n, k \geq 0$. Moreover, the hnf of M can be obtained by *head reduction*, that is by always contracting the redex of M in head position.

Theorem 1.1.2 (Wadsworth [Wad76])**.** *Given $M \in \Lambda^o$, the following are equivalent:*

1. *M is solvable,*
2. *M has a head normal form,*
3. *the head-reduction of M terminates.*

Böhm trees. The study of solvability suggests a notion of separability weaker than the one given by Böhm: two closed λ-terms M, N are *semi-separable* whenever there are $\vec{P} \in \Lambda^o$ such that $M\vec{P}$ is solvable, while $N\vec{P}$ is unsolvable. On the other hand, starting from a solvable λ-term M, one can head-reduce it to its hnf $\lambda x_1 \ldots x_n.x_i M_1 \cdots M_k$, and since "$\lambda x_1 \ldots x_n.x_i$" represents a stable amount of information, one can continue the head-reduction on the subterms. Clearly, if one of the M_i's is unsolvable, that particular head-reduction will not terminate, but otherwise this gives an effective algorithm to extract from M all the stable pieces of its (possibly infinite) output. By pushing this iteration to the limit and using an oracle to determine whether a term is solvable, Barendregt defined a tree representing the whole execution of a λ-term M, namely its Böhm tree. Formally,

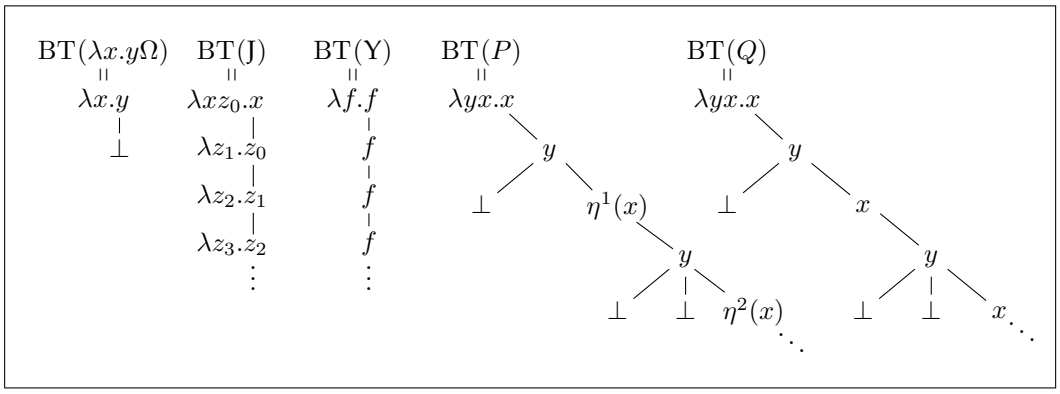


Figure 1.1: Examples of Böhm trees. See [Bar84, Lemma 16.4.4] for the definition of P, Q.

the Böhm tree $\mathrm{BT}(M)$ of M is coinductively defined as follows: if M is unsolvable then $\mathrm{BT}(M) = \bot$; otherwise, if M is solvable and its hnf is $\lambda x_1 \dots x_n.yM_1 \cdots M_k$ then:

$$\mathrm{BT}(M) = \quad \begin{array}{c} \lambda x_1 \dots x_n.y \\ \diagup \quad \cdots \quad \diagdown \\ \mathrm{BT}(M_1) \quad \cdots \quad \mathrm{BT}(M_k) \end{array}$$

Some examples of Böhm trees are given in Figure 1.1. As we will see in Section 1.2, it is impossible to semi-separate two λ-terms having the same Böhm tree, but there are also non semi-separable terms having different Böhm trees.

The λ-theories are the equational theories of the λ-calculus, namely those congruences on the set Λ containing the β-conversion. They become the main object of study when considering the computational equivalence more important than the process of calculus. The set of all λ-theories, ordered by inclusion, forms a complete lattice $\lambda\mathcal{T}$ of cardinality $2^{\aleph_0}$ and constitutes a very rich mathematical structure as shown by Salibra in his work [LS04].

When two λ-terms M, N are equated in a λ-theory $\mathcal{T}$ we write $\mathcal{T} \vdash M = N$ or $M =_{\mathcal{T}} N$.

Some λ-theories are particularly interesting for our discussion since they arise from operational properties of λ-terms. For instance, the least λ-theory $\boldsymbol{\lambda}$ captures β-convertibility. A λ-theory is called *extensional* when it also includes the η-conversion, therefore the least extensional λ-theory $\boldsymbol{\lambda}\eta$ captures exactly the $\beta\eta$-convertibility. Certain λ-theories are called *sensible* because they equate all unsolvable λ-terms. We indicate by $\mathcal{H}$ the least sensible λ-theory and by $\mathcal{B}$ the sensible λ-theory equating all λ-terms having the same Böhm tree. The next section is devoted to discuss in detail two sensible extensional λ-theories, $\mathcal{H}^*$ and $\mathcal{H}^+$, that are important since they characterize observational equivalences between λ-terms.

A λ-theory may also arise from semantical considerations, that is as the theory induced by some model. Models of λ-calculus can be defined algebraically as combinatory algebras satisfying some additional axioms, or categorically as reflexive objects in Cartesian closed categories. The *interpretation of a λ-term M* in a model $\mathcal{M}$ will be denoted by $[\![M]\!]^{\mathcal{M}}$. We write $\mathcal{M} \models M = N$ to indicate that the λ-terms M, N have the same interpretation in $\mathcal{M}$. Every model $\mathcal{M}$ induces a λ-theory as follows: $\mathrm{Th}(\mathcal{M}) = \{M = N \mid \mathcal{M} \models M = N\}$. Finally, we say that $\mathcal{M}$ is *fully abstract* for the λ-theory $\mathcal{T}$ whenever $\mathrm{Th}(\mathcal{M}) = \mathcal{T}$.

1.2 OBSERVATIONAL EQUIVALENCES

The problem of determining when two programs are equivalent is crucial in computer science: for instance, it allows to verify that the optimizations performed by a compiler preserve the meaning of the input program. For λ-calculi, Morris proposed to regard two λ-terms M and N as equivalent when they are contextually equivalent with respect to some fixed set $\mathcal{O}$ of *observables* [Mor68]. This means that one can plug either M or N into any context $C(\!|-|\!)$ without noticing any difference in the global behaviour. Formally, two closed λ-terms M, N are $\mathcal{O}$-equivalent, written $M \equiv^{\mathcal{O}} N$, if for all $P_1, \dots, P_k \in \Lambda^o$:

$$MP_1 \cdots P_k \text{ gives an observable in } \mathcal{O} \text{ exactly when } NP_1 \cdots P_k \text{ does.}$$

The underlying intuition is that the terms in $\mathcal{O}$ represent sufficient stable amounts of information coming out of the computation. The problem of working with this definition, is that the quantification over all possible contexts is difficult to handle. Therefore, various researchers undertook a quest for characterizing observational equivalences both semantically, by defining fully abstract denotational models, and syntactically, by comparing their Böhm trees up to some extensional equivalence.

The λ-theory $\mathcal{H}^*$. The observational equivalence obtained by considering as observables the λ-terms in head normal form is by far the most famous and well studied since it enjoys many interesting properties. For instance, it corresponds to the λ-theory $\mathcal{H}^*$ which is the greatest sensible consistent λ-theory [Bar84, Lemma 16.2.4]. As shown in [Bar84, Thm. 16.2.7], two λ-terms are equivalent in $\mathcal{H}^*$ exactly when their Böhm trees are equal up to denumerably many η-expansions of (possibly) infinite depth. The typical example of an η-expansion of infinite depth is given by the term J satisfying the following property:

$$\mathrm{J}x =_\beta \lambda z_0.x(\mathrm{J}z_0) =_\beta \lambda z_0.x(\lambda z_1.z_0(\mathrm{J}z_1)) =_\beta \lambda z_0.x(\lambda z_1.z_0(\lambda z_2.z_1(\mathrm{J}z_2))) =_\beta \cdots$$

Obviously J and I are $\beta\eta$-distinct, but the Böhm tree of J (depicted in Figure 1.1) is an infinite η-expansion of the identity I, written $\mathrm{BT}(\mathrm{J}) =_{\eta\infty} \mathrm{BT}(\mathrm{I})$. Therefore $\mathcal{H}^* \vdash \mathrm{I} = \mathrm{J}$.

From a semantic perspective, it is now well known that $\mathcal{H}^*$ is the λ-theory induced by the pioneering model of λ-calculus $\mathcal{D}_\infty$ introduced by Scott within the continuous semantics [Sco72]. This result, first reported in [Hyl76, Wad76], means that two λ-terms M, N are equivalent in $\mathcal{H}^*$ exactly when their interpretations $[\![M]\!]^{\mathcal{D}_\infty}$ and $[\![N]\!]^{\mathcal{D}_\infty}$ coincide. In other words, this shows that the model $\mathcal{D}_\infty$ is *fully abstract for* $\mathcal{H}^*$.

Theorem 1.2.1 (Hyland [Hyl76], Wadsworth [Wad76])**.**
Given $M, N \in \Lambda^o$, the following are equivalent

1. $\mathcal{H}^* \vdash M = N$,
2. $\mathrm{BT}(M) =_{\eta\infty} \mathrm{BT}(N)$,
3. $\mathcal{D}_\infty \models M = N$.

Subsequently, several fully abstract models for $\mathcal{H}^*$ were individually introduced in other semantics, for instance in the category of coherence spaces and stable functions [HR90], or in categories of games [DFH99]. Until recently, the most general results consisted in providing sufficient conditions for models living in some class to be fully abstract [Gou95b, M17]. A substantial advance was made by Breuvart in [Bre14] where he proposed the notion of *hyperimmune* model of λ-calculus, and showed that a continuous *K-model* (using the terminology in [Ber00]) is fully abstract for $\mathcal{H}^*$ exactly when it is extensional and hyperimmune, thus providing a characterization.

The λ-theory $\mathcal{H}^+$. Even if taking the head normal forms as observables has become standard for λ-calculi, it is not the only reasonable choice. In particular, when Morris introduced the first observational equivalence in his PhD thesis [Mor68], he considered as observables the β-normal forms. We will denote by $\mathcal{H}^+$ the λ-theory corresponding to the original Morris's observational equivalence[1]. It is easy to check that the λ-theory $\mathcal{H}^+$ is extensional and sensible. Therefore, since $\mathcal{H}^*$ is maximal among sensible theories, we can conclude that $\mathcal{H}^+ \subsetneq \mathcal{H}^*$. Despite the fact that it has been less ubiquitously studied in the literature, also the equality in $\mathcal{H}^+$ has been characterized both syntactically, in terms of Böhm trees, and semantically by providing a fully abstract model.

More precisely, Hyland proved that two λ-terms M, N are equivalent in $\mathcal{H}^+$ exactly when their Böhm trees are equal up to denumerably many η-expansions of finite depth [Hyl75], written $\mathrm{BT}(M) =_{\eta\mathrm{fin}} \mathrm{BT}(N)$. The typical example are the λ-terms P, Q built in [Bar84, §16.4] whose Böhm trees are depicted in Figure 1.1. In this figure, $\eta^n(x)$ denotes the η-expansion of x having depth n, for instance $\eta^3(x) = \lambda z_1.x(\lambda z_2.z_1(\lambda z_3.z_2 z_3))$. Therefore, the Böhm tree of P is such that at every level $2n$ the variable x is η-expanded (in depth) n times. We conclude that $\mathcal{H}^+ \vdash P = Q$ because one can perform infinitely many η-reductions of finite, but increasing, depth in $\mathrm{BT}(P)$ and obtain $\mathrm{BT}(Q)$.

As a brief digression, notice that the existence of such λ-terms P, Q also shows that the λ-theory $\mathcal{B}\eta$ generated by adding the η-equivalence to $\mathcal{B}$ is different form $\mathcal{H}^+$. Indeed, Barendregt proved in [Bar84, Lemma 16.4.3] that by performing one step of η-reduction in a λ-term M it is possible to erase from $\mathrm{BT}(M)$ at most *one* η-redex at every level. Therefore, a proof of $\mathcal{B}\eta \vdash P = Q$ would require to use $=_\eta$ an infinite number of times, which is impossible. As a consequence, we get that the λ-theory $\mathcal{B}\eta$ is strictly included in $\mathcal{H}^+$.

A semantic characterization of Morris's observational theory was provided by Coppo, Dezani and Zacchi in [CDZ87]. They introduced a filter model $\mathcal{D}_{\mathrm{CDZ}}$ and proved, by adapting Tait's reducibility technique, that normalizable terms can be recognized from their semantics in that model — a closed λ-term is normalizable if and only if its interpretation is contained in a specific open subset (with respect to the Scott topology) of $\mathcal{D}_{\mathrm{CDZ}}$. By exploiting this property, and without using the characterization of $\mathcal{H}^+$ in terms of Böhm trees, they were able to prove that two λ-terms are Morris equivalent exactly when they have the same interpretation in $\mathcal{D}_{\mathrm{CDZ}}$. In other words, they proved that $\mathcal{D}_{\mathrm{CDZ}}$ is a fully abstract model of $\mathcal{H}^+$. This solved negatively the conjecture stated in [Böh75, open prob. II.3] that all continuous models built as an inverse limit of a chain of projections induce the maximal theory $\mathcal{H}^*$. Summing up, the analogous of Theorem 1.2.1 holds.

Theorem 1.2.2 (Hyland [Hyl75], Coppo et Al. [CDZ87])**.**
Given $M, N \in \Lambda^o$, the following are equivalent

1. $\mathcal{H}^+ \vdash M = N$
2. $\mathrm{BT}(M) =_{\eta\mathrm{fin}} \mathrm{BT}(N)$
3. $\mathcal{D}_{\mathrm{CDZ}} \models M = N$

As far as we know, there have been no attempts to individuate sufficient conditions for models living in some semantics to be fully abstract for $\mathcal{H}^+$. The situation seems even worse, to the best of our knowledge $\mathcal{D}_{\mathrm{CDZ}}$ is the only model in the literature capturing Morris's equivalence. For this reason, with our PhD student Ruoppolo, we undertook a quest for fully abstract models of $\mathcal{H}^+$ within the relational semantics of λ-calculus.

[1] Note however that this λ-theory is denoted by $\mathscr{T}_{\mathrm{NF}}$ in Barendregt's book [Bar84].

1.3 The Relational Semantics and its Graph Models

The *relational semantics* was introduced by Girard in [Gir88] as a particularly simple quantitative semantics of multiplicative exponential linear logic (MELL, for short). It corresponds to the category **Rel** of sets and relations, where the promotion ! is given by the comonad of finite multisets $\mathcal{M}_{\mathrm{f}}(-)$. Since the intuitionistic arrow $A \to B$ can be decomposed into a linear arrow together with the exponential modality $!A \multimap B$, a relational semantics for the λ-calculus is also at hand. Indeed, a program P of type $A \to B$ will be interpreted as a relation from $\mathcal{M}_{\mathrm{f}}(A)$ to B:

$$[\![P]\!] \subseteq \mathcal{M}_{\mathrm{f}}(A) \times B$$

The relational semantics is called *quantitative* because the multiplicities in the multiset keep track of how many times a resource is necessary during the computation. In other words, $([\alpha, \alpha], \beta) \in [\![P]\!]$ means that the program P needs to make two calls to its input α of type A in order to produce the output β of type B. Besides an explicit handle of resources, the interest of working within the relational semantics is that relational models are simpler than, say, Scott-continuous ones because their elements are not partially ordered.

The underlying category. Formally, the relational semantics of λ-calculus is given by the co-Kleisli category **MRel** of the finite multisets comonad on **Rel**. The category **MRel** can be directly described as follows: its objects are all the sets; a morphism $f : A \to B$ is a relation from $\mathcal{M}_{\mathrm{f}}(A)$ to B; the composition of $f : A \to B$ and $g : A \to B$ is given by:

$$f; g = \{(a_1 \uplus \cdots \uplus a_k, \gamma) \mid (a_1, \beta_1), \ldots, (a_k, \beta_k) \in f \text{ and } ([\beta_1, \ldots, \beta_k], \gamma) \in g\}.$$

From the $\star$-autonomous structure of **Rel**, it follows that **MRel** is Cartesian closed [Gir87, See89] and since the exponential object $[A \to B]$ representing the homset $\mathbf{MRel}(A, B)$ is the set $\mathcal{M}_{\mathrm{f}}(A) \times B$, the category contains reflexive objects $[D \to D] \lhd D$. Therefore, from a categorical point of view, **MRel** constitutes a valid semantics of the untyped λ-calculus.

However the category **MRel** is not *well-pointed*: because of the relational nature of its composition, there are distinct maps $f, g : A \to B$ that coincide on all points $x : 1 \to A$. This was considered a big issue because of a result by Koymans [Koy82] stating that reflexive objects living in well-pointed semantics give rise to combinatory algebras that are moreover *λ-models* while, without the well-pointed condition, one only obtains *λ-algebras*[2]. The difference is that any λ-model induces a λ-theory through the kernel congruence relation of its interpretation function, while this might not be the case for arbitrary λ-algebras. Therefore, some researchers interested in λ-models were reluctant to work with the relational semantics. A first attempt of reconciling the categorical and algebraic point of views is the article [Sel02] where Selinger shows that also a λ-algebra can induce a λ-theory, if one considers a different notion of interpretation, called *absolute* in his terminology. Subsequently, together with Bucciarelli and Ehrhard, we overcame the issue in [M3] by providing a construction, different from the one proposed by Koymans but nevertheless canonical, general enough to turn any reflexive object of a Cartesian closed category into a λ-model. Nowadays the fact that the category **MRel** constitutes a valid semantics of λ-calculus is unanimously accepted by the scientific community.

[2]The notions of λ-algebras and λ-models are two algebraic definitions of a model of λ-calculus. A λ-algebra is a combinatory algebra satisfying Curry's axioms [Bar84, Thm. 5.2.5]. A λ-model is a λ-algebra that moreover satisfies the Meyer-Scott axiom of "weak extensionality" [Bar84, Def. 5.2.7(ii)].

Relational graph models. Some relational models of λ-calculus were individually introduced in the literature [M3, HNPR06]. In 2012, together with Ruoppolo, we initiated a systematic study of the models living in the relational semantics. The aim was to provide some general methods to construct them and to study the induced λ-theories.

In [M21] we defined the class of *relational graph models* (rgm), that can be seen as the relational analogue of the graph models living in Scott's continuous semantics [Ber00].

Definition 1.3.1 (Relational Graph Models)**.** *A* relational graph model $\mathcal{D} = (D, i)$ *is given by an infinite set* D *and a total injection* $i : \mathcal{M}_{\mathrm{f}}(D) \times D \to D$*.*

Intuitively, since the exponential object $[D \to D]$ of $\mathbf{MRel}$ is exactly $\mathcal{M}_{\mathrm{f}}(D) \times D$, every element $(a, \alpha) \in \mathrm{dom}(i)$ represents an arrow of the form $a \to \alpha$ and the map i determines what arrows are identified (in an injective way) with some elements of D. In other words $i(a, \alpha) = \beta$ corresponds to the equality $a \to \alpha = \beta$. Since the injection i has a unique inverse, every rgm $\mathcal{D}$ univocally induces a reflexive object $[D \to D] \lhd D$ in $\mathbf{MRel}$. Notice that the reflexive object under consideration here is D itself, while for a regular graph model it is of the form $(\mathcal{P}(D), \subseteq)$. As a consequence, an rgm $\mathcal{D}$ is extensional whenever i is bijective, while there are no extensional graph models in the continuous semantics.

The notion of rgm is general enough to encompass all previously known examples of relational models. For instance, we can define the relational analogues of:

- Engeler's model [Eng81]: the rgm $\mathcal{E}$, first defined in [HNPR06], has denumerably many atoms and the inclusion map as injection;
- Coppo, Dezani and Zacchi's model [CDZ87]: the extensional rgm $\mathcal{D}_\star$ of [M21] has only one atom $\star$ and its injection is generated by the equation $[\star] \to \star = \star$.
- Scott's model [Sco72]: the extensional rgm $\mathcal{D}_\omega$ introduced in [M3] has only one atom ε and its injection is generated by the equation $[] \to \varepsilon = \varepsilon$.

The model $\mathcal{D}_\omega$ has been studied in [M17], where we proved that it is fully abstract for $\mathcal{H}^*$. Together with Ruoppolo, we have shown in [M21] that $\mathcal{D}_\star$ is fully abstract for $\mathcal{H}^+$. The fact that the λ-theory induced by $\mathcal{E}$ is exactly $\mathcal{B}$ will appear in his PhD thesis [Ruo16].

In general, when investigating the λ-theory induced by some model, the first step is to verify whether the model satisfies an approximation theorem. Approximation theorems state that the interpretation of a λ-term M can be calculated from the interpretations of its "finite approximants", by taking some sort of least upper bound. The standard choice consists in considering as approximants of M the set $\mathsf{App}(M)$ of all finite subtrees of $\mathrm{BT}(M)$. It turns out that *all* relational graph models satisfies the following approximation theorem.

Theorem 1.3.2 (The Approximation Theorem for Böhm Trees [M21])**.**
Given a relational graph model $\mathcal{D}$ *and a closed* λ*-term* M*, we have that:*

$$\sigma \in [\![M]\!]^{\mathcal{D}} \iff \exists t \in \mathsf{App}(M) \text{ such that } \sigma \in [\![t]\!]^{\mathcal{D}}$$

As a direct consequence, the λ-theory induced by an rgm $\mathcal{D}$ satisfies $\mathcal{B} \subseteq \mathrm{Th}(\mathcal{D})$. Perhaps surprisingly, it is enough to have that $\mathcal{D}$ is extensional to conclude that $\mathcal{H}^+ \subseteq \mathrm{Th}(\mathcal{D})$. The latter result follows easily from a characterization of $\mathcal{H}^+$ given by Lévy in terms of "extensional approximants" of a Böhm tree [Lév05].

Problem 1. *The* λ*-theories representable by relational graph models belong to the interval* $[\mathcal{B}, \mathcal{H}^*]$*. How many distinct* λ*-theories can be represented by* rgms*? Is it possible to give a complete characterization of all representable* λ*-theories?*

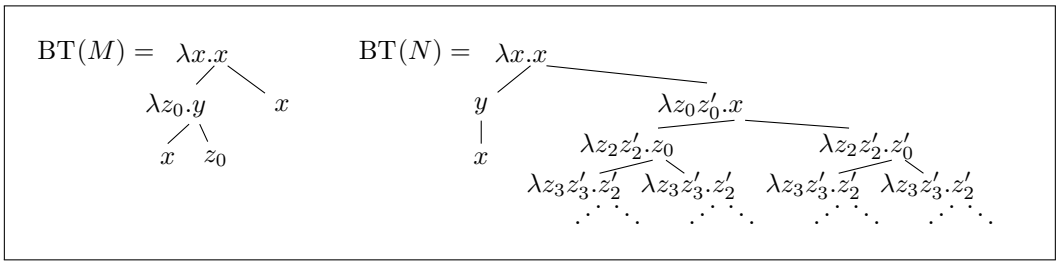


Figure 1.2: The Böhm trees of two λ-terms M, N such that $\mathcal{H}^* \vdash M = N$, but $\mathcal{H}^+ \vdash M \neq N$.

1.4 Characterizing Fully Abstract Relational Models of $\mathcal{H}^+$

Together with Ruoppolo, we provided sufficient conditions for an rgm $\mathcal{D}$ to induce as λ-theory $\mathcal{H}^+$. Namely, we proved in [M21] that every extensional rgm *preserving the polarities of the empty multiset* (in a technical sense) is fully abstract for $\mathcal{H}^+$. In a larger collaboration including Breuvart and Polonsky [M9], we strengthened this result and obtained a characterization of all relational graph models that are fully abstract for $\mathcal{H}^+$.

Since all extensional rgms equate *at least as* $\mathcal{H}^+$, the difficult part is to find a condition guaranteeing that they do not equate more. In other words, we need to analyze in detail the equations in $\mathcal{H}^* - \mathcal{H}^+$ and establish a separation result.

A new separation theorem. The crucial observation is that whenever two λ-terms M and N are equal in $\mathcal{H}^*$, but not in $\mathcal{H}^+$, their Böhm trees are similar but there exists a (possibly virtual) position where they differ because of an infinite η-expansion of a variable x. Such an expansion is not always of the form $\mathrm{BT}(\mathrm{J}x)$, since J is not the only infinite η-expansion of I. An example of this situation is depicted in Figure 1.2, where $\mathrm{BT}(M)$ and $\mathrm{BT}(N)$ differ at position $\langle 1 \rangle$ because of an infinite η-expansion of x that "follows the structure" of the complete infinite binary tree.

More generally, for every computable infinite tree T, one can define a combinator J_T whose Böhm tree is an infinite η-expansion of the identity *following the structure of* T. This characterization is complete, in the sense that all λ-definable infinite η-expansions of the identity can be described as the Böhm tree of some J_T for a suitable computable infinite T.

Thanks to a refined version of the Böhm-out technique [Böh68], we have shown that it is always possible to extract such a difference by defining a suitable applicative context. As usual, finite η-differences can be destroyed during the process of Böhming out. This theorem shows that infinite η-differences can always be preserved.

Theorem 1.4.1 (Weak Separation Theorem [M9])**.**
Let $M, N \in \Lambda^o$ such that $\mathcal{H}^* \vdash M = N$ while $\mathcal{H}^+ \vdash M \neq N$. Then, there exist $P_1, \dots, P_k \in \Lambda^o$ such that, for some infinite computable tree T, we have

$$MP_1 \cdots P_k =_{\beta\eta} \mathrm{I} \qquad NP_1 \cdots P_k =_{\mathcal{B}} \mathrm{J}_T \qquad \text{(or } \textit{vice versa}\text{)}$$

The consequences of this theorem are both semantical and syntactical. On the one side it is central in the characterization of all relational graph models that are fully abstract for $\mathcal{H}^+$. On the other side it implies that the λ-theory $\mathcal{H}^+$ validates the ω-rule, a result that will be presented in Section 1.5.

Lambda-König relational models. Thanks to Theorem 1.4.1, the problem of distinguishing all λ-terms M, N that are equal in $\mathcal{H}^*$ and different in $\mathcal{H}^+$, boils down to distinguish the identity I from all the combinators J_T, where T is a computable infinite tree. To ensure that this difference is still detectable in an rgm, we introduce the notion of a *λ-König model*. Intuitively, an rgm $\mathcal{D}$ is λ-König when every computable infinite tree T has an infinite path f (which exists by König's lemma) witnessed by some element of $\mathcal{D}$.

Definition 1.4.2. *Let $\mathcal{D}$ be an rgm.*

(i) Given an infinite tree T and a function $f : \mathbb{N} \to \mathbb{N}$ representing an infinite path of T, we say that an element $\alpha \in D$ is a witness for T following f *whenever:*

$$\alpha = a_0 \to \cdots \to a_{f(0)} \to \alpha'$$

and there is a witness $\beta \in a_{f(0)}$ for the subtree of T rooted at position $f(0)$ following the function $k \mapsto f(k+1)$.

(ii) We denote by $\mathsf{W}_{\mathcal{D}}(T)$ the set of all witnesses for T following some infinite path f.

(iii) We say that $\mathcal{D}$ is λ-König *whenever $\mathsf{W}_{\mathcal{D}}(T) \neq \emptyset$ for all infinite computable trees T.*

As we will see, the set $\mathsf{W}_{\mathcal{D}}(T)$ is inhabited by those $\alpha \in D$ such that $[\alpha] \to \alpha \notin [\![\mathsf{J}_T]\!]$. We try to explain the reasons that are behind such a characterization of $\mathsf{W}_{\mathcal{D}}(T)$.

Suppose, by the way of contradiction, that $[\alpha] \to \alpha \in [\![\mathsf{J}_T]\!]$ for some $\alpha \in \mathsf{W}_{\mathcal{D}}(T)$. By the approximation theorem, $[\alpha] \to \alpha$ belongs to the interpretation of some *finite* approximant t of $\mathrm{BT}(\mathsf{J}_T)$. Now, assume that T at the root has k children $T_1, \ldots, T_k$, then $\mathsf{J}_T =_\beta \lambda x y_1 \ldots y_k.x(\mathsf{J}_{T_1} y_1) \cdots (\mathsf{J}_{T_k} y_k)$ and its approximant t is of the form $\lambda x y_1 \ldots y_k.x t_1 \cdots t_k$ for $t_i \in \mathsf{App}(\mathsf{J}_{T_i} y_i)$. From the fact that α is a witness for T following some path f we know that $\alpha = a_0 \to \cdots \to a_{f(0)} \to \alpha'$ and there is $\beta \in a_{f(0)}$ which is a witness for $T_{f(0)}$. The definition of $[\![t]\!]$ is such that $[\alpha] \to \alpha \in [\![t]\!]$ implies $[\beta] \to \beta \in [\![\lambda y_{f(0)}.t_{f(0)}]\!]$, which in its turn entails $t_{f(0)} \neq \bot$. (The last implication follows from $[\![\bot]\!] = \emptyset$.) Since this reasoning can be iterated indefinitely along the path f, at every level ℓ the term t should have a subterm $t_{f(\ell)} \neq \bot$, which is impossible because t is a finite approximant.

Proposition 1.4.3. *For an rgm $\mathcal{D}$, we have $\mathsf{W}_{\mathcal{D}}(T) = \{\alpha \in D \mid [\alpha] \to \alpha \notin [\![\mathsf{J}_T]\!]^{\mathcal{D}}\}$. Therefore, for any $\alpha \in \mathsf{W}_{\mathcal{D}}(T)$, we have $[\alpha] \to \alpha \in [\![\mathrm{I}]\!] - [\![\mathsf{J}_T]\!]$ which entails that $\mathcal{D} \models \mathrm{I} \neq \mathsf{J}_T$ for all T.*

From this proposition and Theorem 1.4.1, we get the following result.

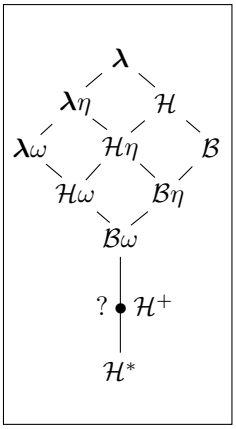
Theorem 1.4.4 (Breuvart et Al. [M9])**.**
An extensional rgm $\mathcal{D}$ is λ-König if and only if $\mathcal{D}$ is fully abstract for $\mathcal{H}^+$.

Proof sketch. ($\Rightarrow$) Since $\mathcal{D}$ is extensional we have $\mathcal{H}^+ \subseteq \mathrm{Th}(\mathcal{D})$, we need to show that the other inclusion holds. By contradiction, suppose there are two closed λ-terms M and N such that $\mathcal{D} \models M = N$ but $\mathcal{H}^+ \vdash M \neq N$. By maximality of $\mathcal{H}^*$ we must have $\mathcal{H}^* \vdash M = N$, so we can apply Theorem 1.4.1 and obtain $\vec{P} \in \Lambda^o$ such that, say, $\mathcal{H}^+ \vdash M\vec{P} = \mathrm{I}$ and $\mathcal{H}^+ \vdash N\vec{P} = \mathsf{J}_T$ for some infinite computable tree T. By monotonicity of the interpretation, we get $[\![\mathrm{I}]\!]^{\mathcal{D}} = [\![M\vec{P}]\!]^{\mathcal{D}} \subseteq [\![N\vec{P}]\!]^{\mathcal{D}} = [\![\mathsf{J}_T]\!]^{\mathcal{D}}$. We derive a contradiction by applying Proposition 1.4.3.

($\Leftarrow$) Suppose $\mathrm{Th}(\mathcal{D}) = \mathcal{H}^+$, then $\mathcal{D} \models \mathrm{I} \neq \mathsf{J}_T$ for all infinite computable tree T. As a consequence of Proposition 1.4.3, we get that $\mathsf{W}_{\mathcal{D}}(T) \neq \emptyset$ for all such trees T. □

Problem 2. *Is it possible to adapt the techniques developed by Breuvart in [Bre14] to characterize all relational graph models that are fully abstract for $\mathcal{H}^*$?*

1.5 THE ω-RULE



During his PhD, Barendregt studied the problem whether the ω-rule is valid in λ-calculus [Bar71]. The ω-rule is defined by:

$$(\omega) \qquad \forall P \in \Lambda^o. MP = NP \text{ entails } M = N.$$

In other words, the ω-rule states that two closed λ-terms are equal exactly when they coincide on all *closed* arguments, it is therefore clear that it concerns some kind of extensionality. It is easy to check that if a λ-theory $\mathcal{T}$ satisfies the ω-rule, written $\mathcal{T} \vdash \omega$, then $\mathcal{T}$ is extensional. On the other hand it is non trivial to verify that the extensionality induced by the ω-rule is strictly stronger than the one corresponding to η-conversion. Indeed, the proof needs to consider Plotkin's terms [Bar84, Def. 17.3.26], which are very convoluted *universal generators* [Bar84, Def. 8.2.7(ii)].

Given a λ-theory $\mathcal{T}$, we denote by $\mathcal{T}\eta$ the smallest extensional λ-theory including $\mathcal{T}$, and by $\mathcal{T}\omega$ its closure under the ω-rule. The picture on the side, where $\mathcal{T}$ is above $\mathcal{T}'$ if $\mathcal{T} \subsetneq \mathcal{T}'$, is taken from [Bar84, Thm. 17.4.16] and shows some known results about the λ-theories introduced in Section 1.1. Most of the inclusions follow from the definition of such λ-theories, and the fact that $\mathcal{T} \subseteq \mathcal{T}'$ entails $\mathcal{T}\eta \subseteq \mathcal{T}'\eta$ and $\mathcal{T}\omega \subseteq \mathcal{T}'\omega$. In general, the difficult part is to show that, for $\mathcal{T} \in \{\boldsymbol{\lambda}, \mathcal{H}, \mathcal{B}\}$, the inclusion $\mathcal{T}\eta \subseteq \mathcal{T}\omega$ is actually strict. As mentioned above, the counterexample showing that $\boldsymbol{\lambda}\eta \nvdash \omega$ is based on Plotkin's terms. We do not discuss the technique for proving $\mathcal{H} \nvdash \omega$, which is presented in [Bar84, §17.4]. Here we prefer to focus on those λ-theories including $\mathcal{B}$ and strictly contained in $\mathcal{H}^*$, since the fact that $\mathcal{H}^* \vdash \omega$ clearly follows from the maximality of $\mathcal{H}^*$ among sensible theories.

We have seen that the λ-terms P, Q of Figure 1.1 are equated in $\mathcal{H}^+$, but different in $\mathcal{B}\eta$. Perhaps surprisingly, they can also be used to prove that $\mathcal{B}\eta \subsetneq \mathcal{B}\omega$ since $P =_{\mathcal{B}\omega} Q$ holds. The proof uses the following interesting fact: for every $M \in \Lambda^o$, there exists $k \geq 0$ such that $M\Omega \cdots \Omega$, k times, becomes unsolvable [Bar84, Lemma 17.4.4]. By inspecting Figure 1.1, we notice that in $\mathrm{BT}(P)$ the variable y is applied to an increasing number of Ω's (represented by $\bot$). So, when substituting some $M \in \Lambda^o$ for y in $\mathrm{BT}(Py)$, there is a level k of the tree where $M\Omega \cdots \Omega$ becomes $\bot$, thus cutting $\mathrm{BT}(PM)$ at level k. The same reasoning can be done for $\mathrm{BT}(QM)$. Therefore $\mathrm{BT}(PM)$ and $\mathrm{BT}(QM)$ only differ because of finitely many η-expansions. Since $\mathcal{B}\omega$ is extensional and M is arbitrary, we conclude that $\mathcal{B}\omega \vdash P = Q$.

The question whether $\mathcal{H}^+ \vdash \omega$ was a longstanding open problem, which we solved positively in [M9]. The key point is to show that the property "M has a β-normal form, while N does not" can be preserved using a *non-empty* applicative context $(\!|-|\!)P_1\vec{P}$. This is trivial when M, N are semi-separable, since M can be sent to the identity and N to an unsolvable term. Otherwise, it is possible to apply our Theorem 1.4.1, and send M to I and N to J_T. It is easy to check that the property under consideration is stable under applications of suitable finite η-expansions of the identity.

Theorem 1.5.1 (Breuvart et Al. [M9]). *$\mathcal{H}^+$ satisfies the ω-rule.*

Problem 3. *As reported in [Bar84, Proof of Thm. 17.4.16], Sallé conjectured that the inclusion $\mathcal{B}\omega \subseteq \mathcal{H}^+$ is actually strict. Prove Sallé's conjecture, or show $\mathcal{B}\omega = \mathcal{H}^+$.*

The Simply Typed λ-Calculus

$$\frac{}{\Delta, x : A \vdash x : A}\ (\mathrm{var}) \qquad \frac{\Delta, x : A \vdash M : B}{\Delta \vdash \lambda x.M : A \to B}\ (\mathrm{lam}) \qquad \frac{\Delta \vdash M : A \to B \quad \Delta \vdash N : A}{\Delta \vdash MN : B}\ (\mathrm{app})$$

Figure 1.3: The inference rules of simply typed λ-calculus. In (lam) we assume $x \notin \mathrm{dom}(\Delta)$

1.6 Simple Types and Intersection Types

The λ-calculus can be endowed with several kinds of type systems, that allow to assign elements of a given set $\mathbb{T}$ of *types* to untyped λ-terms. A particular type system depends on two parameters: the set $\mathbb{T}$ of types and the inference rules of type assignment.

The simply typed λ-calculus was introduced by Church in [Chu40]. Under the Curry-Howard isomorphism, it correspond to intuitionistic logic [SU06]. The interest on this system among researchers has been revitalized by the recent publication of Barendregt's book on typed λ-calculi [BDS13].

We consider the set $\mathbb{T}^0$ of *simple types* that are built from a single atomic type 0 using the arrow constructor.

$$\mathbb{T}^0 : \qquad A, B, C \ ::= \ 0 \ \mid \ A \to B$$

The main idea is that a λ-term M gets the type $A \to B$ when it is considered as a function from terms of type A to those of type B. In this case, if N has type A, then the application MN is "legal" and gets type B. To assign types to open λ-terms we use *type environments* that are partial functions from Var to $\mathbb{T}^0$. We denote by $\Delta = x_1 : A_1, \ldots, x_n : A_n$ the type environment satisfying $\mathrm{dom}(\Delta) = \{x_1, \ldots, x_n\}$ and $\Delta(x_i) = A_i$ for all $x_i \in \mathrm{dom}(\Delta)$.

The inference rules of the simply typed λ-calculus are presented in Figure 1.3. When $\Delta \vdash M : A$ can be derived using these rules, we say that *M has type A in the environment Δ*.

One of the most important features of the simply typed λ-calculus is that it enjoys normalization. As discussed in [Gan80a], Turing first noticed that by reducing in a simply typable λ-term M the innermost redex having highest type A, one obtains a λ-term M' having fewer redexes of type A. This reasoning led to a proof of weak normalization by transfinite induction up to ω^2. Subsequently, Gandy obtained a semantic proof of strong normalization by exploiting denotational models based on strictly monotone functions.

Theorem 1.6.1 (Gandy [Gan80b])**.** *The simply typed λ-calculus is strongly normalizable.*

In [dV87], de Vrier refined Gandy's proof by showing that it is possible to associate with every simply typable λ-term M the exact bound of the lengths of its reductions. Other proofs of strong normalization based on Tait's reducibility technique appeared in the literature [GLT89, Sch91]. However, none of these proofs explains what quantity actually decreases during the reduction. Indeed, as we think that "the maximum number of steps towards its normal form" is not a satisfying answer, we consider the problem open.

Problem 4. *Construct an "easy" assignment $\#(-)$ of (possibly transfinite) ordinals to simply typed λ-terms in such a way that $M \to_\beta N$ entails that $\#M > \#N$. By the fact that the ordinals are well-ordered, this immediately shows that the system is strongly normalizing.*

Denotational models of simply typed λ-calculus. We recall here the set-theoretical definition of a model of simply typed λ-calculus.

A *typed applicative structure* $\mathcal{M}$ is given by a pair $((\mathcal{M}_A)_{A\in\mathbb{T}^0}, \cdot\ )$ where each $\mathcal{M}_A$ is a structure whose carrier is non-empty, and $\cdot$ is a function that associates to every $d \in \mathcal{M}_{A\to B}$ and every $e \in \mathcal{M}_A$ an element $d \cdot e$ in $\mathcal{M}_B$. We say that typed applicative structure $\mathcal{M}$ is *extensional* whenever for every $d, d' \in \mathcal{M}_{A\to B}$, $d \cdot e = d' \cdot e$ for all $e \in \mathcal{M}_A$ entails $d = d'$. This is always the case when $\mathcal{M}_{A\to B}$ is a set of functions and $\cdot$ is functional application. A typed applicative structure $\mathcal{M}$ is called *hereditarily finite* if every $\mathcal{M}_A$ is finite.

To interpret the free variables of M in the structure $\mathcal{M}$ we need a *valuation* ν, that is a map from Var to elements of $\mathcal{M}$. A valuation ν *agrees* with a type environment Δ when $\Delta(x) = A$ implies $\nu(x) \in \mathcal{M}_A$. Given a valuation ν and an element $d \in \mathcal{M}$, we write $\nu\{d/x\}$ for the valuation ν' that coincides with ν, except for x, where ν' takes the value d.

A *model* $\mathcal{M}$ of simply typed λ-calculus is an extensional typed applicative structure such that the clauses below define a total interpretation function $[\![\cdot]\!]^{\mathcal{M}}_{(\cdot)}$ which maps derivations $\Delta \vdash M : A$ and valuations ν agreeing with Δ to elements of $\mathcal{M}_A$:

- $[\![\Delta \vdash x : A]\!]^{\mathcal{M}}_\nu = \nu(x)$,
- $[\![\Delta \vdash NP : A]\!]^{\mathcal{M}}_\nu = [\![\Delta \vdash N : B \to A]\!]^{\mathcal{M}}_\nu \cdot [\![\Delta \vdash P : B]\!]^{\mathcal{M}}_\nu$,
- $[\![\Delta \vdash \lambda x.N : A \to B]\!]^{\mathcal{M}}_\nu \cdot d = [\![\Delta, x : A \vdash N : B]\!]^{\mathcal{M}}_{\nu\{d/x\}}$ for every $d \in \mathcal{M}_A$.

When the derivation $\Delta \vdash M : A$ is clear we simply write $[\![M]\!]^{\mathcal{M}}_\nu$ for its interpretation. Moreover, whenever M is a closed λ-term, we simplify the notation further and write $[\![M]\!]^{\mathcal{M}}$ since its interpretation is independent from the valuation.

Logical relations have been extensively used in the study of typed λ-calculi. They constitute a powerful tool for establishing links between syntax and semantics, or for relating different models. For instance, they can be used for proving Friedman's completeness theorem [Fri73], which characterizes $\beta\eta$-equality, and Jung-Tiuryn's [JT93] and Sieber's [Sie92] theorems on the characterization of λ-definability. We refer the reader to [AC98, §4.5] and [BDS13, §3C] for a more detailed presentation.

For our purposes, we only need the following semantic notion of a logical relation.

Definition 1.6.2. *Given two valuation models $\mathcal{M}, \mathcal{N}$, a* logical relation $\mathcal{R}$ between $\mathcal{M}$ and $\mathcal{N}$ *is a family $\{\mathcal{R}_A\}_{A\in\mathbb{T}^0}$ of binary relations $\mathcal{R}_A \subseteq \mathcal{M}_A \times \mathcal{N}_A$ such that for all $A, B \in \mathbb{T}^0$, $f \in \mathcal{M}_{A\to B}$ and $g \in \mathcal{N}_{A\to B}$ we have:*

$$f\ \mathcal{R}_{A\to B}\ g \text{ if and only if } \forall h \in \mathcal{M}_A, h' \in \mathcal{N}_A\ [\ h\ \mathcal{R}_A\ h' \Rightarrow f(h)\ \mathcal{R}_B\ g(h')\ ].$$

Given a logical relation $\mathcal{R}$ and $f \in \mathcal{M}_A$ we define $\mathcal{R}_A(f) = \{g \in \mathcal{N}_A \mid f\ \mathcal{R}_A\ g\}$ and, for $Y \subseteq \mathcal{M}_A$, $\mathcal{R}_A(Y) = \bigcup_{f\in Y} \mathcal{R}_A(f)$. Moreover, we write $\mathcal{R}^-$ for the inverse of $\mathcal{R}$.

It is well known that a logical relation $\mathcal{R}$ is univocally determined by the value of $\mathcal{R}_0$, and that the following fundamental lemma of logical relations holds [AC98, §4.5].

Lemma 1.6.3 (Fundamental Lemma)**.** *Let $\mathcal{R}$ be a logical relation between $\mathcal{M}$ and $\mathcal{N}$ then, for all closed λ-terms M having simple type A, we have $[\![M]\!]^{\mathcal{M}}\ \mathcal{R}_A\ [\![M]\!]^{\mathcal{N}}$.*

Intersection types were introduced in the late '70s by Coppo and Dezani [CD80]. Subsequently, researchers developed several type systems based on intersection types [BCD83, CDV80, RV84, CDZ87]. We refer to [BDS13, Part 3] for a survey.

We present here the system CDV defined by Coppo, Dezani and Venneri in [CDV80].

(a) **The System CDV**

$$\frac{}{x_1:\sigma_1,\ldots,x_n:\sigma_n \vdash_\wedge x_i:\sigma_i}\ (\text{ax}) \qquad \frac{\Gamma\vdash_\wedge M:\tau\to\sigma \quad \Gamma\vdash_\wedge N:\tau}{\Gamma\vdash_\wedge MN:\sigma}\ (\to_E)$$

$$\frac{\Gamma,x:\sigma\vdash_\wedge M:\tau}{\Gamma\vdash_\wedge \lambda x.M:\sigma\to\tau}(\to_I) \quad \frac{\Gamma\vdash_\wedge M:\sigma \quad \Gamma\vdash_\wedge M:\tau}{\Gamma\vdash_\wedge M:\sigma\wedge\tau}\ (\wedge_I) \quad \frac{\Gamma\vdash_\wedge M:\sigma \quad \sigma\le\tau}{\Gamma\vdash_\wedge M:\tau}\ (\le)$$

(b) **The Subtyping Rules**

$$\sigma\le\sigma\ (\text{refl}) \qquad \sigma\wedge\tau\le\sigma\ (\text{incl}_L) \qquad \sigma\wedge\tau\le\tau\ (\text{incl}_R)$$

$$(\sigma\to\tau)\wedge(\sigma\to\tau')\le\sigma\to(\tau\wedge\tau') \quad (\to_\wedge)$$

$$\frac{\sigma\le\gamma \quad \gamma\le\tau}{\sigma\le\tau}\ (\text{trans}) \qquad \frac{\sigma\le\tau \quad \sigma\le\tau'}{\sigma\le\tau\wedge\tau'}\ (\text{glb}) \qquad \frac{\sigma'\le\sigma \quad \tau\le\tau'}{\sigma\to\tau\le\sigma'\to\tau'}\ (\to)$$

Figure 1.4: The intersection type system CDV and its subtyping rules.

Given a set $\mathbb{A}$ of atomic types, the set $\mathbb{T}^{\mathbb{A}}_\wedge$ of *intersection types* over $\mathbb{A}$ is generated by:

$$\mathbb{T}^{\mathbb{A}}_\wedge: \qquad \sigma,\tau ::= \alpha \mid \sigma\to\tau \mid \sigma\wedge\tau \qquad (\text{for } \alpha\in\mathbb{A})$$

The set $\mathbb{T}^{\mathbb{A}}_\wedge$ is partially ordered by the *subtyping* relation $\le$ defined in Figure 1.4(b). We write $\simeq$ for the equivalence generated by setting $\sigma\simeq\tau$ if and only if both $\sigma\le\tau$ and $\tau\le\sigma$ hold. Environments $\Gamma = x_1:\tau_1,\ldots,x_n:\tau_n$ are handled as in the simply typed case. The inference rules for deriving $\Gamma\vdash_\wedge M:\sigma$ in the system CDV are given in Figure 1.4(a). The idea is that whenever a λ-term M has both type σ and type τ, it also gets the type $\sigma\wedge\tau$.

Also this system CDV, like the simply typed λ-calculus, enjoys strong normalization. It is however well known that there are strongly normalizable terms, like $\lambda x.xx$, that are not simply typable. On the contrary, CDV provides a characterization of normalizable λ-terms.

Theorem 1.6.4 (Coppo et Al. [CDV80])**.**
A λ-term M is typable in CDV *if and only if M is strongly normalizable.*

In collaboration with Barendregt, Coppo and Dezani [BCD83] introduced a special atomic type ω, and a subtyping rule specifying that $\sigma\le\omega$ for all types σ. As a consequence, all λ-terms become typable in the new system, since they can have ω as a type.

Another way to understand the language of intersection types is as a formal language for representing the compact elements of the domain $(\mathcal{D},\sqsubseteq)$ which they serve to define. Intuitively, the intersection types σ,τ represent compact elements d,e of $\mathcal{D}$, the association being bijective but order-reversing (i.e., $\sigma\le\tau$ entails $e\sqsubseteq d$). The type $\sigma\to\tau$ represents a step function, which is a compact element of $\mathcal{D}$, $\sigma\wedge\tau$ represents the join $d\sqcup e$ and ω represents $\bot$. This correspondence is at the basis of the presentation of intersection type systems as *filter models*. In these models the interpretation of M corresponds to the set of types σ such that $\vdash_\wedge M:\sigma$, which is an ideal w.r.t. $\le$, therefore $[\![M]\!]$ is a filter w.r.t. $\sqsubseteq$.

1.7 Lambda Definability and Type Inhabitation are Undecidable

We discuss here two apparently unrelated problems concerning the typed λ-calculi presented in Section 1.6. The first one is the problem of λ-definability for the simply typed λ-calculus, while the second one is the problem of type inhabitation for the system CDV.

Definability for simply typed λ-calculus. Consider the simply typed λ-calculus on simple types $\mathbb{T}^0$ with one ground type 0. A (hereditarily finite) *full model* of this calculus is given by: a collection of sets $\mathcal{F} = (\mathcal{F}_A)_{A\in\mathbb{T}^0}$ such that $\mathcal{F}_0 \neq \emptyset$ is finite and $\mathcal{F}_{A\to B} = \mathcal{F}_B^{\mathcal{F}_A}$ (that is, the set of functions from $\mathcal{F}_A$ to $\mathcal{F}_B$); the map $\cdot$ is simply functional application.

We say that an element $f \in \mathcal{F}_A$ is λ-*definable* whenever there exists a closed λ-term M having type A whose interpretation is f, i.e. such that $[\![M]\!]^{\mathcal{F}} = f$. The following question, raised by Plotkin in [Plo73], is known as the *Definability Problem*:

DP: "Given an element f of any hereditarily finite full model, is f λ-definable?"

A natural restriction considered in the literature [Jol03, Loa01] is the following:

DP_n: "Given an element f of $\mathcal{F}_n$, is f λ-definable?"

where $\mathcal{F}_n$ (for $n \geq 1$) denotes the unique (up to isomorphism) full model whose ground set $\mathcal{F}_0$ has n elements. Statman's conjecture stating that DP is decidable [Sta82] was refuted by Loader [Loa01], who proved in 1993 (but published in 2001) that DP_n is undecidable for every $n > 6$. Such a result was subsequently strengthened by Joly, who showed in [Jol03] that DP_n is undecidable for all $n > 1$.

Theorem 1.7.1 (Loader [Loa01], plus Joly [Jol03])**.**

1. *(Loader) The Definability Problem is undecidable.*
2. *(Loader/Joly) DP_n is undecidable for every $n > 6$ (resp. $n > 1$).*

Inhabitation for intersection types. Consider now the λ-calculus endowed with the intersection type system CDV based on a countable set $\mathbb{A}$ of atomic types. We say that a type $\sigma \in \mathbb{T}_\wedge$ is *inhabited* if the judgement $\vdash_\wedge M : \sigma$ is derivable for some closed λ-term M.

The *Inhabitation Problem* for this system is formulated as follows:

IHP: "Given an intersection type σ, is σ inhabited?"

We will also be interested in the following restriction of IHP:

IHP_n: "Given an intersection type σ with at most n atoms, is σ inhabited?"

In 1999, Urzyczyn [Urz99] proved that IHP is undecidable for suitable intersection types, called "game types" in [BDS13, §17E], and therefore for the whole CDV. His idea was to prove that solving the inhabitation problem for a game type σ is equivalent to winning a suitable "tree game" G. An arbitrary number of atoms may be needed since, in the Turing-reduction, the actual amount of atoms in σ is determined by the tree game G.

Theorem 1.7.2 (Urzyczyn [Urz99])**.**

1. *The Inhabitation Problem for game types is undecidable.*
2. *The Inhabitation Problem is undecidable.*

An unexpected connection. The undecidability of DP and that of IHP are major results in theoretical computer science. For a thorough presentation, we refer the reader to Section 4A and Section 17E of [BDS13], respectively. Both proofs of undecidability are obtained by reducing these problems to well-known undecidable problems (and eventually to the Halting problem). However, the instruments that are used to achieve these results are very different — the proof by Loader proceeds by reducing DP to the two-letter word rewriting problem, while the proof by Urzyczyn reduces (through a series of reductions) IHP to the emptiness problem for queue automata. The fact that these proofs are different is not surprising since the two problems, at first sight, really seem unrelated.

Starting from an original idea by Salvati, and in collaboration with Barendregt and Gehrke, we proved in [M30] that DP and IHP are actually Turing-equivalent, by providing a perhaps unexpected link between the two problems. More precisely, in that work we describe a construction that allows to obtain the following Turing-reductions[3]:

(i) Inhabitation Problem for game types $\leq_T$ Definability Problem,

(ii) Definability Problem $\leq_T$ Inhabitation Problem (cf. [Sal09]),

(iii) $\mathrm{DP}_n \leq_T \mathrm{IHP}_n$ (cf. [Sal09]).

Therefore, by (i) and (ii) we get that the undecidability of DP and IHP follows from each other. Moreover, by (iii) and Theorem 1.7.1(2) we conclude that IHP_n is undecidable whenever $n > 1$, which is a new result refining Urzyczyn's one.

In the next sections we present the main ingredients of our constructions, while the actual undecidability results will be presented in Section 1.10.

1.8 Uniform Intersection Types.

In general, a useful approach to prove that a decision problem is undecidable, is to identify a "sufficiently difficult" fragment of the problem. As mentioned earlier, Urzyczyn has shown the undecidability of inhabitation for game types that constitute a proper subset $\mathcal{G}$ of intersection types. Formally, the set of game types is given by $\mathcal{G} := \mathbb{A} \cup \mathcal{B} \cup \mathcal{C}$:

$$\mathcal{A} := \mathbb{A}^\wedge, \quad \mathcal{B} := (\mathcal{A} \to \mathcal{A})^\wedge, \quad \mathcal{C} := (\mathcal{D} \to \mathcal{A})^\wedge \text{ for } \mathcal{D} := \{\sigma \wedge \tau \mid \sigma, \tau \in (\mathcal{B} \to \mathcal{A})\}$$

where $Y^\wedge := \{\sigma_1 \wedge \cdots \wedge \sigma_n \mid \sigma_i \in Y \text{ for all } 1 \leq i \leq n\}$ and $Y \to Z := \{\tau \to \sigma \mid \tau \in Y, \sigma \in Z\}$.

In our case we focus on intersection types that are *uniform* with simple types. The idea behind this notion, is that we want to compare the set of intersection types with a full model of simply typed λ-calculus $\mathcal{F}$ over a set X. Therefore we need to consider the set $\mathbb{T}^X_\wedge$ of intersection types having the elements of X as atomic types, and we need to stratify such a set following the arrow structure of the simple types.

Definition 1.8.1. *The set $\mathcal{U}_X(A)$ of* intersection types uniform with $A \in \mathbb{T}^0$ *is given by:*

$$\mathcal{U}_X(0) := X^\wedge, \qquad \mathcal{U}_X(B \to C) := (\mathcal{U}_X(B) \to \mathcal{U}_X(C))^\wedge.$$

It turns out that Urzyczyn's game types are all uniform: $\mathcal{A} \subseteq \mathcal{U}_{\mathbb{A}}(0)$, $\mathcal{B} \subseteq \mathcal{U}_{\mathbb{A}}(0 \to 0)$ and $\mathcal{D} \subseteq \mathcal{U}_{\mathbb{A}}((0 \to 0) \to 0)$ thus $\mathcal{C} \subseteq \mathcal{U}_{\mathbb{A}}(((0 \to 0) \to 0) \to 0)$. As a consequence, the inhabitation problem for uniform intersection types over $\mathbb{A}$ is undecidable as well.

[3] Recall that if the problem P_1 is undecidable and $P_1 \leq_T P_2$, then also P_2 is undecidable

CDV$^\omega$. To relate the uniform intersection types in CDV with the full model $\mathcal{F}$, we need an intermediate type system CDV$^\omega$. The system CDV$^\omega$ is a variation of CDV where intersection types are extended by adding a distinguished element ω at ground level.

The set $\mathcal{U}_{X\cup\{\omega\}}(A)$ of *intersection types with ω uniform with A* will be denoted by $\mathcal{U}^\omega_X(A)$ and we write ω_A for the type in $\mathcal{U}^\omega_X(A)$ defined by setting $\omega_0 := \omega$ and $\omega_{B\to C} := \omega_B \to \omega_C$.

The system CDV$^\omega$ over $\mathbb{T}^{\mathbb{A}\cup\{\omega\}}_\wedge$, whose judgments are denoted by $\Gamma \vdash^\omega_\wedge M : \sigma$, is generated by adding the following rule to the definition of $\leq$ in Figure 1.4:

$$\frac{\sigma \in \mathcal{U}^\omega_{\mathbb{A}}(A)}{\sigma \leq \omega_A}\ (\leq_A)$$

By construction, for every $A \in \mathbb{T}^0$, the type ω_A is a maximal element of $\mathcal{U}^\omega_{\mathbb{A}}(A)$. Therefore, the system CDV$^\omega$ should not be confused with the usual intersection type systems with ω, where ω is a maximal element for all intersection types and can be assigned to any λ-term.

The new subtyping relation respects the stratification of uniform types, in the sense that only intersection types that are uniform with the same simple type A can be comparable.

Lemma 1.8.2. *Let $\sigma \in \mathcal{U}^\omega_{\mathbb{A}}(A)$ and $\tau \in \mathcal{U}^\omega_{\mathbb{A}}(A')$. Then we have that $\sigma \leq \tau$ entails $A = A'$.*

Despite the fact CDV$^\omega$ is an intersection type system, the β-normal forms that can be typed using uniform intersection types coincide with those that are typable in the simply typed λ-calculus. Notice that we can focus on normal forms without loss of generality because both systems are strongly normalizable.

Lemma 1.8.3. *For every β-normal closed λ-term M and for every $\sigma \in \mathcal{U}^\omega_{\mathbb{A}}(A)$ we have:*

$$\vdash^\omega_\wedge M : \sigma \text{ entails } \vdash M : A.$$

The property above does not generalize to arbitrary closed λ-terms because, while we consider only uniform intersection types, we do not restrict the intersection type system. In other words, non-uniform intersection types may still be used in a deduction, as shown in the following example.

Example 1.8.4. *Let us consider two λ-terms $M = \lambda zy.y$ and $N = \lambda x.xx$. It is easy to check that*

$$\vdash^\omega_\wedge M : ((\beta \wedge (\beta \to \beta)) \to \beta) \to \alpha \to \alpha \text{ and } \vdash^\omega_\wedge N : (\beta \wedge (\beta \to \beta)) \to \beta.$$

Therefore, in CDV$^\omega$ it is possible to assign MN the uniform type $\alpha \in \mathcal{U}^\omega_{\mathbb{A}}(0 \to 0)$ as follows:

$$\frac{\vdash^\omega_\wedge \lambda zy.y : ((\beta \wedge (\beta \to \beta)) \to \beta) \to \alpha \to \alpha \quad \vdash^\omega_\wedge \lambda x.xx : (\beta \wedge (\beta \to \beta)) \to \beta}{\vdash^\omega_\wedge (\lambda zy.y)(\lambda x.xx) : \alpha \to \alpha}$$

However N is not simply typable, hence neither is MN.

Clearly every normal form M which is typable with a simple type A can be also typed with an intersection type σ which is uniform with A. We can conclude that the type systems CDV and CDV$^\omega$are equivalent on normal forms in the following (strong) sense.

Lemma 1.8.5. *For every β-normal closed λ-term M, and for every $\sigma \in \mathcal{U}_{\mathbb{A}}(A)$ we have:*

$$\Gamma \vdash_\wedge M : \sigma \iff \Gamma \vdash^\omega_\wedge M : \sigma.$$

1.9 THE MONOTONE MODEL OVER $\mathcal{P}(X)$

It is well known that the hereditarily finite full model $\mathcal{F}$ contains a lot of undefinable elements (*junk*). For this reason it is useful to define another model $\mathcal{S}$, based on monotone functions, which can be seen as a refinement of $\mathcal{F}$. The interpretations of a λ-term in the two models will be subsequently related using the fundamental lemma of logical relations (Lemma 1.6.3).

Definition 1.9.1. *Given a finite set $X \subseteq \mathbb{A}$ of atoms, the* monotone model *$\mathcal{S}$ over $\mathcal{P}(X)$ corresponds to the typed applicative structure $((\mathcal{S}_A, \sqsubseteq_A)_{A\in\mathbb{T}^0}, \cdot)$, were $\cdot$ is functional application and:*

- *$\mathcal{S}_0 = \mathcal{P}(X)$ and $f \sqsubseteq_0 g$ if and only if $f \subseteq g$,*
- *$\mathcal{S}_{A\to B}$ is the set of the monotone functions from $\mathcal{S}_A$ to $\mathcal{S}_B$ endowed with the following partial order: $f \sqsubseteq_{A\to B} g$ holds if and only if for all $h \in \mathcal{S}_A$ we have $f(h) \sqsubseteq_B g(h)$.*

Each $\mathcal{S}_A$ is a finite join-semilattice, and thus a complete lattice, whose join is denoted by $\sqcup$ and whose bottom is denoted by $\bot_A$.

Given $f \in \mathcal{S}_A$ we write $f\uparrow$ for its upward closure in $\mathcal{S}_A$, that is $f\uparrow = \{f' \in \mathcal{S}_A \mid f \sqsubseteq f'\}$.

As mentioned in Section 1.6, intersection types can be used to represent elements of a filter model. Similarly, uniform intersection types over $X \cup \{\omega\}$ correspond to elements of the monotone model $\mathcal{S}$. The intuition is the following: at ground type, every intersection $\alpha_1 \wedge \cdots \wedge \alpha_k$ of atoms can be viewed as a set $\{\alpha_1, \ldots, \alpha_k\}$, the type ω being represented by the empty set; at higher levels, every arrow type $\sigma \to \tau$ can be seen as a "step function".

Step functions constitute the "building blocks" of monotone functions, and are defined as follows. Given $f \in \mathcal{S}_A$ and $g \in \mathcal{S}_B$ we write $f \mapsto g$ for the corresponding *step function*:

$$(f \mapsto g)(h) = \begin{cases} g & \text{if } f \sqsubseteq_A h, \\ \bot_B & \text{otherwise.} \end{cases}$$

We are now going to formalize the correspondence between uniform intersection types and elements of the model $\mathcal{S}$.

For all $A \in \mathbb{T}^0$ we define a function $\iota_A : \mathcal{U}^\omega_X(A) \to \mathcal{S}_A$ by induction on A as follows. Given $\alpha \in X$ and $\sigma, \tau \in \mathcal{U}^\omega_X(0)$ we let

$$\iota_0(\alpha) = \{\alpha\}, \qquad \iota_0(\omega) = \bot_0 = \emptyset, \qquad \iota_0(\sigma \wedge \tau) = \iota_0(\sigma) \sqcup \iota_0(\tau)$$

For $\sigma, \tau \in \mathcal{U}^\omega_X(A \to B)$ we define:

$$\iota_{A\to B}(\sigma \to \tau) = \iota_A(\sigma) \mapsto \iota_B(\tau), \qquad \iota_{A\to B}(\sigma \wedge \tau) = \iota_{A\to B}(\sigma) \sqcup \iota_{A\to B}(\tau).$$

Thanks to the presence of the maximal element ω_A, the correspondence between $\mathcal{U}^\omega(A)$ and $\mathcal{S}_A$ is actually very faithful. First, one can verify that for every $\sigma \in \mathcal{U}^\omega(A)$, we have that $\sigma \simeq \omega_A$ entails $\iota_A(\sigma) = \bot_A$. Second, one can check that for all $A \in \mathbb{T}^0$ and $\sigma, \tau \in \mathcal{U}^\omega(A)$ we have that $\sigma \leq \tau$ holds if and only if $\iota_A(\tau) \sqsubseteq \iota_A(\sigma)$. In other words, the order between the elements is reversed under the action of ι.

Theorem 1.9.2. *The map ι_A is an order-reversing bijection on $\mathcal{U}^\omega_X(A)/\simeq$.*

The above results are related to the Stone duality for intersection types, first noticed by Coppo and Dezani in [CD80] and by Abramsky in [Abr87].

Proposition 1.9.3. *Let M be a β-normal closed λ-term such that $\vdash M : A$, then for all $\sigma \in \mathcal{U}^\omega_X(A)$ we have:*

$$\vdash^\omega_\wedge M : \sigma \qquad \Longleftrightarrow \qquad \iota_A(\sigma) \sqsubseteq [\![M]\!]^{\mathcal{S}}_\nu$$

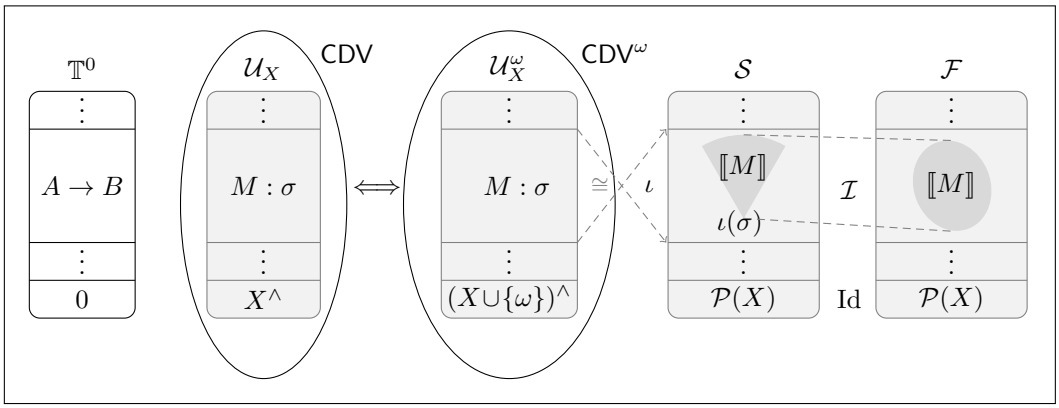


Figure 1.5: Inhabitation in CDV over X reduces to definability in $\mathcal{F}$ over $\mathcal{P}(X)$.

1.10 DP AND IHP ARE EQUIDECIDABLE

IHP $\leq_T$ DP. We show that the undecidability of the Definability Problem follows from the undecidability of the inhabitation problem (for game types) in CDV. The reduction is obtained by linking through a suitable logical relation $\mathcal{I}$ the continuous model $\mathcal{S}$ built in the previous section and the full model $\mathcal{F}$ over $\mathcal{P}(X)$, where $X \subsetneq \mathbb{A}$ is a finite set of atoms.

We consider the logical relation $\mathcal{I}$ generated by taking the identity at ground level. This actually constitutes a *logical retract*, in the sense that at every level $A \in \mathbb{T}^0$ we have:

(i) for all $f \in \mathcal{S}_A$ there is $g \in \mathcal{F}_A$ such that $f\ \mathcal{I}_A\ g$,

(ii) for all $f, f' \in \mathcal{S}_A, g \in \mathcal{F}_A$ if $f\ \mathcal{I}_A\ g$ and $f'\ \mathcal{I}_A\ g$ then $f = f'$.

As a consequence we get, for every subset $S \subseteq \mathcal{S}_A$, that $\mathcal{I}_A^-(\mathcal{I}_A(S)) = S$.

For every β-normal closed λ-term M having type A and every $\sigma \in \mathcal{U}_X(A)$ we have the following computable chain of equivalences:

$$\begin{array}{rcll} \vdash_\wedge M : \sigma & \iff & \vdash_\wedge^\omega M : \sigma, & \text{by Lemma 1.8.5,} \\ & \iff & [\![M]\!]^{\mathcal{S}} \in \iota_A(\sigma)\uparrow, & \text{by Proposition 1.9.3,} \\ & \iff & [\![M]\!]^{\mathcal{F}} \in \mathcal{I}_A(\iota_A(\sigma)\uparrow), & \text{by Lemma 1.6.3 plus condition (ii).} \end{array}$$

Suppose, by the way of contradiction, that DP is decidable. We want to decide whether $\sigma \in \bigcup_{A \in \mathbb{T}^0, X \subseteq_f \mathbb{A}} \mathcal{U}_X(A) \subseteq \mathcal{G}$ is inhabited in CDV. As CDV is strongly normalizable, we can focus on β-normal forms. Since for a β-normal form M being typable with $\sigma \in \mathcal{U}_X(A)$ implies being typable with A (by Lemma 1.8.3) we can focus on simply typed λ-terms. Now we can take the set X of all atoms in σ, compute the simple type A such that $\sigma \in \mathcal{U}_X(A)$, and effectively construct the finite set $\mathcal{I}_A(\iota_A(\sigma)\uparrow) \subseteq \mathcal{F}$ (the full model over $\mathcal{P}(X)$). If DP is decidable, then we can also decide with finitely many tests whether there is a λ-definable $f \in \mathcal{I}_A(\iota_A(\sigma)\uparrow)$. By the above equivalences, such an f exists if and only if σ is inhabited. This yields a reduction of IHP for game types (hence for uniform types) to DP.

Theorem 1.10.1 (Salvati et Al. [M30])**.** *The undecidability of the Definability Problem follows by a reduction from the one of the Inhabitation Problem for game types, Theorem 1.7.2(1).*

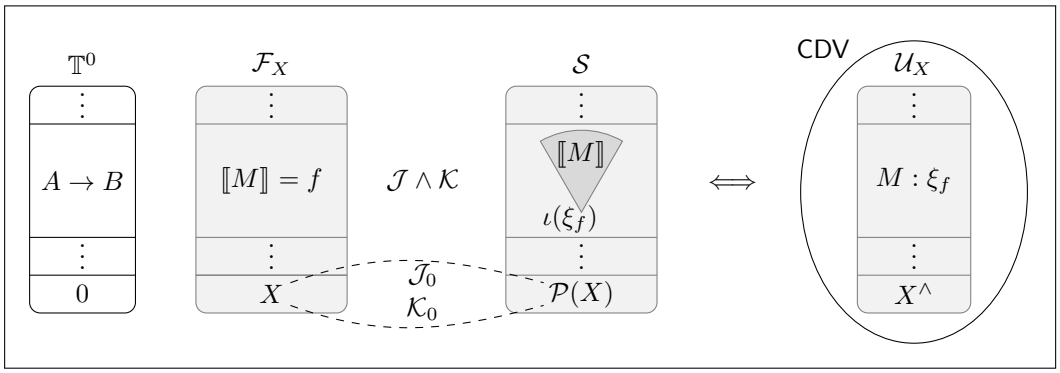


Figure 1.6: Definability in $\mathcal{F}$ over X reduces to inhabitation in CDV over X.

DP $\leq_T$ IHP. We now show that the undecidability of inhabitation in CDV follows directly from the undecidability of λ-definability in the full model $\mathcal{F}$ over a finite set $X \subseteq \mathbb{A}$. The main idea is a simple embedding of the elements of $\mathcal{F}$ into the uniform intersection types: given f in $\mathcal{F}_A$ we can define an intersection type ξ_f in $\mathcal{U}_X(A)$ as follows:

- if $A = 0$, then $\xi_f = f$,
- if $A = B \to C$, then $\xi_f = \bigwedge_{g \in \mathcal{F}_B} \xi_g \to \xi_{f(g)}$.

Also in this case the continuous model $\mathcal{S}$ over $\mathcal{P}(X)$ will play a key role. This time, we consider two logical relations between the full model $\mathcal{F}$ and the continuous model $\mathcal{S}$:

- the relation $\mathcal{J}$ generated by $\mathcal{J}_0 = \{(f, F) \mid f \in F \subseteq \mathcal{F}_0\}$ and
- the relation $\mathcal{K}$ generated by $\mathcal{K}_0 = \{(f, \{f\}) \mid f \in X\}$.

Together, they allow to capture the following property.

Lemma 1.10.2. *For every $f \in \mathcal{F}_A$ and $g \in \mathcal{S}_A$ we have $f\ \mathcal{J}_A\ g$ and $f\ \mathcal{K}_A\ g$ iff $\iota_A(\xi_f) \sqsubseteq g$.*

For a β-normal closed λ-term M, we have the following computable chain of equivalences:

$$\begin{array}{rcll} [\![M]\!]^{\mathcal{F}} = f & \iff & f\ \mathcal{J}_A\ [\![M]\!]^{\mathcal{S}} \text{ and } f\ \mathcal{K}_A\ [\![M]\!]^{\mathcal{S}}, & \text{by Lemma 1.6.3 (twice),} \\ & \iff & \iota_A(\xi_f) \sqsubseteq [\![M]\!]^{\mathcal{S}}, & \text{by Lemma 1.10.2,} \\ & \iff & \vdash_\wedge M : \xi_f, & \text{by Proposition 1.9.3.} \end{array}$$

Therefore $f \in \mathcal{F}$ is definable if and only if the intersection type ξ_f is inhabited. This yields a reduction of the Definability Problem (resp. DP_n) to the Inhabitation Problem (resp. IHP_n).

Theorem 1.10.3 (Salvati et Al. [M30])**.**

1. *The undecidability of the Inhabitation Problem follows by a reduction from the undecidability of the Definability Problem, Theorem 1.7.1(1).*
2. *The undecidability of IHP_n for all $n > 1$ follows by a reduction from the undecidability of DP_n for all $n > 1$, Theorem 1.7.1(2).*

Problem 5. *While the reduction IHP $\leq_T$ DP presented in [M30] is a proper Turing reduction, DP $\leq_T$ IHP is an "ordinary" (many-one, actually one-to-one) reduction, which is logically simpler. Are IHP and DP equivalent with respect to the finer structure of many-one degrees?*

1.11 RAISING ML TO THE POWER OF SYSTEM F, WHILE KEEPING NORMALIZATION

One of the most efficient techniques for assuring the "good behaviour" of a program is *static type-checking*: types are assigned to every subexpression of a program, so that consistency of such an assignment (which is checked at compile time) implies well-behavedness. Such an assignment may be *explicit*, as in C or Java, or *implicit* as in ML where types are inferred by the compiler. In this context *type polymorphism* allows greater flexibility, making it possible to reuse code that works with elements of different types.

Polymorphism vs decidability. The problem of full polymorphism is that it leads to undecidable type systems: this is the case of Girard's system F [Gir72] whose type checking has been proved to be undecidable by Wells [Wel94]. For this reason ML is endowed with the so called second-class polymorphism, more restricted but allowing a type inference procedure [Mil78, Mil84]. Unfortunately, the programmer is also *forced* to use second-class polymorphism only. To solve this problem Le Botelan and Rémy designed ML^F [LBR03] an extension of ML providing a partial type annotation mechanism with an automatic type reconstructor. This extension allows to write system F programs, which is not always possible in ML, while remaining conservative: ML programs still type-check without needing any annotation. An important feature of ML^F are principal type schemas which are obtained by employing a downward bounded quantification $\forall(\alpha \geq \sigma)\tau$, called a *flexible* quantifier. Such a type intuitively denotes that τ may be instantiated to any type $\tau\{\sigma'/\alpha\}$, *provided that* σ' *is an instantiation of* σ.

Strong normalization. As discussed in Section 1.6, one of the properties of well-behavedness that a type system can assure is *strong normalization*, that is the termination of all typable programs whatever execution strategy is used. An important result is the fact that system F is strongly normalizing, which was proved by Girard in [Gir72] using his famous reducibility candidates. In collaboration with Tranquilli, we proved that also ML^F enjoys the strong normalization property, a problem raised in [RY10].

Our proof is achieved in several steps. The starting point is xML^F [RY12], the Church-style version of ML^F: here type inference is omitted, with the aim of providing an internal language to which a compiler might map the surface language briefly presented above. Compared to Church-style system F, the type reduction $\to_\iota$ of xML^F is more complicated, and may *a priori* cause unexpected glitches: namely, it could cause non-termination, or block the reduction of a β-redex. To prove that none of this happens, we define a translation from xML^F into a *coercion calculus* Fc. Indeed, xML^F has syntactic entities (the *instantiations*) which testify an instance relation between types, and it is quite natural to regard them as coercions. Hence our translation sends xML^F terms to Fc terms and xML^F instantiations to Fc coercions. The strong normalization of xML^F follows from the same property of Fc, which is easy to show since Fc can be seen as a decorated version of system F.

We then transfer the result from xML^F to all other versions of ML^Fthrough suitable bisimulation theorems.

Theorem 1.11.1 (Manzonetto, Tranquilli [M28])**.**
The system ML^F *is strongly normalizable.*

Note that, while the strong normalization property for β-reduction is a rather theoretical problem, that of ι-reductions is needed to safely implement xML^F interpreters and compilers. Moreover, we think that the coercion calculus Fc has its own interest as it might be used to provide an abstract characterization of coercions.

2 *The Resource Calculus and its Denotational Semantics*

In which we show that the theory of analytical differentiation can be fruitfully applied to the study of λ-calculus, and opens the way to capture intensional properties of programs.

- **Categorical Models for Simply Typed Resource Calculi.**
  A. Bucciarelli, T. Ehrhard and G. Manzonetto.
  26th Conference on Mathematical Foundations of Programming Semantics, MFPS'10, Electronic Notes in Theoretical Computer Science, Volume 265, pages 213-230, 2010.
- **Böhm's Theorem for Resource Lambda Calculus through Taylor Expansion.**
  G. Manzonetto and M. Pagani.
  Typed Lambda Calculi and Applications (TLCA'11), Lecture Notes in Computer Science, Volume 6690, pages 153-168, 2011.
- **Full Abstraction for Resource Calculus with Tests.**
  A. Bucciarelli, A. Carraro, T. Ehrhard and G. Manzonetto.
  In M. Bezem, editor, Computer Science Logic (CSL'11), volume 12 of Leibniz International Proceedings in Informatics (LIPIcs), pages 97-111, Dagstuhl, Germany, 2011.
- **What is a Categorical Model of the Differential and the Resource Lambda Calculi?**
  G. Manzonetto.
  Mathematical Structures in Computer Science, Volume 22(3), pages 451-520, 2012.
- **Full Abstraction for the Resource Lambda Calculus with Tests, through Taylor Expansion.** A. Bucciarelli, A. Carraro, T. Ehrhard and G. Manzonetto.
  Logical Methods in Computer Science, Volume 8, Issue 4, Paper 3, pp. 1-44, 2012.

In 2009, I have studied the resource calculus introduced in Tranquilli's PhD thesis. The strong intuitions I acquired analyzing that calculus allowed me to better understand also Ehrhard's differential λ-calculus which is tightly connected.

During my postdoc at the University Paris-Nord, I launched a research programme in collaboration with Bucciarelli and Ehrhard to investigate the denotational models of resource calculus, whose theory was still at the beginning. On the one side, starting from the notion of a differential category, we were able to provide an abstract categorical definition of models of the resource calculus. On the other sider we have built concrete examples of models in the relational semantics of differential linear logic. Unfortunately, none of the models introduced in this semantics turned out to be fully abstract for this calculus.

During my postdoc at the Radboud University (2010), in a larger collaboration including Carraro, we decided to extend the resource calculus with convergency tests to give the contexts additional discriminating power. We then proved that the relational graph model $\mathcal{D}_\omega$ constitutes a fully abstract model of this extended calculus. In that period Pagani and Ronchi della Rocca undertook a quest for characterizing the solvability in the resource calculus both in terms of a terminating reduction strategy and in terms of typability with non-idempotent intersection types. Building on our complementary competencies, Pagani and I have formulated the analogue of the Böhm Theorem in the context of resource calculus. In a joint collaboration, we proved that all extensionally distinct normal resource terms having different Taylor expansion can be separated by a suitable applicative context.

2.1 INTRODUCTION

The λ-calculus is a very simple and powerful paradigmatic programming language but it is not resource sensitive in the sense that a program is allowed to erase its argument or duplicate it an arbitrary number of times. This becomes a problem when the program runs in environments with bounded resources (like integrated systems) or in presence of depletable arguments (like quantum data that cannot be duplicated for physical reasons). In these contexts one would like to be able to verify at compile time the amount of resources needed by a program during its execution, and to express the fact that it *actually consumes* its arguments. This idea of 'resource consumption' is central in Girard's quantitative semantics of linear logic [Gir88] which establishes an analogy between linearity in the sense of computer science (programs using arguments exactly once) and algebraic linearity.

This connection found by Girard is interesting because the mainstream denotational semantics tends to interpret programs as partial continuous functions defined on domains. The partiality is necessary to model divergent computations, like the unsolvable terms, while the continuity takes into account the finite nature of computation — a program only needs a finite piece of the input in order to produce a finite piece of the output. In the model, this is expressed by the fact that a continuous function can be seen as the limit of a sequence of compact functions, therefore the notion of approximation arising in this context is given by finite functions. This point of view is reflected in the syntax of λ-calculus by the theory of Böhm trees and of their finite approximants. Such an approach is however orthogonal to the one followed in analysis where the functions are total, defined on vector spaces and approximated by (multi)linear maps through (iterated) differentiation.

Drawing on these insights, Ehrhard was able to conciliate these points of view by introducing a syntactic notion of derivative of a program in the programming languages discipline. This opens the way for the application of powerful results developed in the study of analytical differentiation to this field of computer science. Indeed, as in analysis the derivative $\mathsf{D}(f)$ gives information on the function f (since it is a measure of how f changes as its input changes) in the theory of programming languages it is possible to infer quantitative properties of a program P by studying its derivative.

The differential λ-calculus. In 2003, Ehrhard, in collaboration with Regnier, designed a resource sensitive paradigmatic programming language called *the differential λ-calculus* [ER03], extending the regular λ-calculus with differential and linear constructions. In this language, there are two different operators that can be used to apply a program to its argument: the usual application and a *linear application*. This last one defines a syntactic derivative operator which is an excellent candidate to increase control over programs executed in environments with bounded resources. Indeed, the evaluation of $\mathsf{D}(\lambda x.M)\cdot N$ (the derivative of a program $\lambda x.M$ on the input N) has a precise operational meaning: it captures the fact that the argument N is available for $\lambda x.M$ "exactly once". Therefore, the differential substitution $\frac{\partial M}{\partial x}\cdot N$ forces the presence of non-determinism in the system: if M contains several subroutines asking for the argument N the system needs to choose non-deterministically which subroutine is actually fed with the only available copy of N. As a consequence, the differential λ-calculus constitutes a useful framework for studying the notions of linearity and non-determinism, and the relation between them.

Approximation by Taylor expansion. As expected, iterated differentiation yields a natural notion of linear approximation of the ordinary application of an input to a program. This notion relates to ordinary application through the *Taylor formula* [ER08]:

$$MN = \sum_{n=0}^{\infty} \frac{1}{n!} \left(\mathsf{D}^n(M) \cdot (\underbrace{N, \ldots, N}_{n \text{ times}})\right)0$$

where $1/n!$ is a numerical coefficient and $\mathsf{D}^n(M) \cdot (N, \ldots, N)$ represent the n-th derivative of M, which is an n-linear symmetric function, applied to n copies of N. More generally, if one fully develops each application occurring in a program into its corresponding Taylor expansion, one expresses the program as an infinite sum of purely "differential programs" all of which contain only linear applications and regular applications to 0. The study of the relationship between a program and its full Taylor expansion opens the way to the renewal of the theory of approximations usually based on domains and Böhm trees.

The resource calculus. In the nineties Boudol introduced the *λ-calculus with multiplicities* [Bou93], an extension of call-by-value λ-calculus where arguments may come in limited availability and are mixed together. His aim was to analyze finer notions of observational equivalences, like the one given by the π-calculus via Milner's translation [BL00]. Inspired by the work of Boudol, Ehrhard and Regnier, Tranquilli introduced in his Phd thesis the *resource calculus* [Tra09] and proved in [Tra11] that it corresponds to the intuitionistic minimal fragment of differential nets with promotion [ER05], exactly as λ-calculus corresponds to the intuitionistic minimal fragment of linear logic proof-nets [Gir87].

The resource calculus has depletable and reusable arguments that come in multisets, like in the λ-calculus with multiplicities. However, it is a call-by-name language that handles non-determinism via formal sums and its depletable resources are linear rather than affine[1], like in the differential λ-calculus. In other words a term M is not applied to another term N, like in the usual λ-calculus, but rather to a *bag of resources*:

$$M[L_1, \ldots, L_k, (M_1 + \cdots + M_n)^!]$$

where the resources L_i are linear, while the promoted resources M_i are reusable. The resource calculus is intimately related with the differential λ-calculus in the sense that the resource term $M[L_1, \ldots, L_k, (M_1 + \cdots + M_n)^!]$ corresponds to the differential λ-term $(\mathsf{D}^k(M) \cdot (L_1, \ldots, L_k))(M_1 + \cdots + M_n)$. Under this correspondence all the summands in the Taylor expansion of MN can be written as $M[N, \ldots, N]$, therefore the promotion-free resource calculus can be also seen as the *target language* of the Taylor expansion. For this reason, in this chapter we will focus on Tranquilli's resource calculus even if most of the results could be formulated for the differential λ-calculus as well.

Outline. We start by recalling the syntax and operational semantics of resource calculus and surveying the most important results. Subsequently we present an investigation on the denotational semantics of this calculus: on the one side we present Cartesian closed differential categories that constitute the categorical notion of model of the resource calculus, on the other side we study the concrete model $\mathcal{D}_\omega$ living in the relational semantics. We conclude the chapter by showing that, even if $\mathcal{D}_\omega$ is not fully abstract for the resource calculus, it possible to extend the syntax of the calculus with an exception mechanism in order to match the equality induced by this model.

[1] Linear resources must be used exactly once, while affine resources either zero or one times.

2.2 THE RESOURCE CALCULUS

The *resource calculus* was introduced in 2009 by Tranquilli [Tra09] as an alternative syntax for the differential λ-calculus. The calculus has three syntactic categories: *resource terms* that are in functional position, *bags* that are in argument position and represent multisets of resources, and *finite formal sums* that represent the possible results of a computation. The grammar for generating the set Λ^r of resource terms and the set Λ^b of bags is given by:

Λ^r:	M, N, L	$::= x \mid \lambda x.M \mid MP$	resource terms
Λ^b:	P	$::= [L_1, \dots, L_n, \mathbf{M}^!]$	bags
$\mathbb{N}\langle\Lambda^r\rangle$:	$\mathbf{M}, \mathbf{N}$	$::= 0 \mid M \mid M + \mathbf{M}$	sums of terms

A *bag* $[\vec{L}, \mathbf{M}^!]$ is a compound object, consisting of a multiset $[L_1, \dots, L_k]$ of *linear resources* and a multiset $\mathbf{M} = M_1 + \cdots + M_n$, presented in additive notation, representing the *reusable resources*. Roughly speaking, the linear resources $L_1, \dots, L_k$ must be used exactly once during a reduction, while the reusable ones in $\mathbf{M}$ can be used ad libitum (hence, following the linear logic notation, $\mathbf{M}$ is decorated with a ! superscript). We shall deal with bags as if they were multisets presented in multiplicative notation, defining union by

$$[\vec{M}, \mathbf{M}^!] \cdot [\vec{N}, \mathbf{N}^!] := [\vec{M}, \vec{N}, (\mathbf{M} + \mathbf{N})^!].$$

This operation is commutative, associative and has the *empty bag* $1 := [0^!]$ as neutral element. To lighten the notations we write $[L_1, \dots, L_k]$ for the bag $[L_1, \dots, L_k, 0^!]$.

The α-equivalence and the set $\mathrm{FV}(M)$ of free variables of M are defined as in the ordinary λ-calculus. From now on resource terms are considered up to α-equivalence.

Adding terms and bags. Let $(\mathbb{N}, +, \cdot, 0, 1)$ be the semiring of natural numbers. We denote by $\mathbb{N}\langle\Lambda^r\rangle$ the free $\mathbb{N}$-module generated by Λ^r. This amounts to say that $\mathbb{N}\langle\Lambda^r\rangle$ is the set of finite formal sums of resource terms, with 0 referring to the neutral element. Though in practice only sums of terms are considered, in some definitions we exploit the fact that sums can be defined on bags as well, and we denote by $\mathbb{N}\langle\Lambda^b\rangle$ the corresponding module.

The grammar for resource terms and bags does not include any sums, but under the scope of a $(\cdot)^!$. As a syntactic sugar, we extend all the constructors to sums as follows:

$$\lambda x.(\textstyle\sum_i M_i) := \sum_i \lambda x.M_i, \qquad \mathbf{M}(\textstyle\sum_i P_i) := \sum_i \mathbf{M}P_i,$$
$$(\textstyle\sum_i M_i)\mathbf{P} := \sum_i M_i\mathbf{P}, \qquad [(\textstyle\sum_i M_i)] \cdot \mathbf{P} := \sum_i [M_i] \cdot \mathbf{P}.$$

This captures the fact that all constructors except for the $(\cdot)^!$ are (multi)linear, as expected. The intuition is that a reusable sum $(M+N)^!$ represents a resource that can be used several times and each time the system can choose non-deterministically either M or N.

Observe that, in the particular case of the empty sum, we obtain

$$\lambda x.0 := 0, \qquad M0 := 0, \qquad 0P := 0, \qquad [0] := 0, \qquad 0 \cdot P := 0, \qquad [0^!] = 1.$$

Thus 0 annihilates any resource term or bag, except when it lies under a $(\cdot)^!$. As an example of this *extended (meta-)syntax*, we may simply write $(x_1 + x_2)[y_1 + y_2, (z_1 + z_2)^!]$ instead of

$$x_1[y_1, (z_1 + z_2)^!] + x_1[y_2, (z_1 + z_2)^!] + x_2[y_1, (z_1 + z_2)^!] + x_2[y_2, (z_1 + z_2)^!].$$

This kind of meta-syntactic notation is discussed thoroughly in [ER08].

$$y\langle N/x\rangle := \begin{cases} N & \text{if } y = x, \\ 0 & \text{otherwise,} \end{cases} \qquad (\lambda y.M)\langle N/x\rangle := \lambda y.(M\langle N/x\rangle),$$
$$(MP)\langle N/x\rangle := M\langle N/x\rangle P + M(P\langle N/x\rangle),$$
$$[\mathbf{M}^!]\langle N/x\rangle := [\mathbf{M}\langle N/x\rangle, \mathbf{M}^!], \qquad ([M]\cdot P)\langle N/x\rangle := [M\langle N/x\rangle]\cdot P + [M]\cdot P\langle N/x\rangle.$$

Figure 2.1: Linear substitution, in the abstraction case we suppose $y \notin \mathrm{FV}(N) \cup \{x\}$.

Substitutions. Due to the presence of two kinds of resources, we need two different notions of substitutions: the usual λ-calculus substitution and a linear one, which is particular to differential and resource calculi (see [ER08, PT09]).

Definition 2.2.1 (Substitutions)**.** *We define the following substitution operations.*

1. $M\{N/x\}$ *denotes the usual capture-free substitution of* N *for* x *in* M*. It is extended to sums as in* $\mathbf{M}\{\mathbf{N}/x\}$ *by linearity in* $\mathbf{M}$*.*
2. $M\langle N/x\rangle$ *denotes the capture free* linear substitution *defined inductively in Figure 2.1. It is extended to sums as in* $\mathbf{M}\langle\mathbf{N}/x\rangle$ *by bilinearity in both* $\mathbf{M}$ *and* $\mathbf{N}$*.*

Intuitively, linear substitution replaces the given resource for *exactly one* linear free occurrence of the variable. In presence of multiple occurrences, all possible choices are taken into account and the result is the sum of them. For example, we have $(x[x])\langle \mathrm{I}/x\rangle = \mathrm{I}[x] + x[\mathrm{I}]$. Note the peculiar behaviour appearing when N is linearly substituted for x in $[x^!]$. The idea is that, since $x^!$ represents ω-copies of x, one copy is substituted for N while the other ones remain unchanged. In other words, we have $[x^!]\langle x/N\rangle = [N, x^!]$.

Linear substitution $M\langle N/x\rangle$ bears resemblance to differentiation $\frac{\partial M}{\partial x}\cdot N$, as shown clearly in Ehrhard and Regnier's differential λ-calculus [ER03]. For instance, the following commutation lemma corresponds to Schwarz Theorem.

Lemma 2.2.2 (Schwarz Theorem [ER03, ER08])**.**
Given $L, M, N \in \Lambda^r$ *and* $y \notin \mathrm{FV}(M[N])$ *we have*

$$L\langle M/y\rangle\langle N/x\rangle = L\langle N/x\rangle\langle M/y\rangle + L\langle M\langle N/x\rangle/y\rangle.$$

In particular, whenever $x \notin \mathrm{FV}(M)$ *the two substitutions commute.*

Operational semantics. The β_{r}-reduction in the framework of resource calculus is defined as the contextual closure of the following rule:

$$(\lambda x.M)[L_1, \ldots, L_k, \mathbf{N}^!] \to_{\beta_{\mathsf{r}}} M\langle L_1/x\rangle \cdots \langle L_k/x\rangle\{\mathbf{N}/x\}. \qquad (\beta_{\mathsf{r}})$$

Notice that the β_{r}-rule is independent from the order of the linear substitutions, as shown by the Schwarz Theorem above. As usual, we say that $M \in \Lambda^r$ *is in* β_{r}*-normal form* if there is no $\mathbf{M}$ such that $M \to_{\beta_{\mathsf{r}}} \mathbf{M}$. We say that sum of resource terms $\mathbf{M}$ *is in* β_{r}*-normal form* whenever all its summands are, in particular the empty sum 0 is a β_{r}-normal form.

The regular λ-calculus can be embedded into the resource calculus by translating every application MN into $M[N^!]$. In this fragment the β_{r}-reduction defined above coincides with the usual β-reduction. Hence the resource calculus has usual looping terms like

$\mathbf{\Omega} := (\lambda x.x[x^!])[(\lambda x.x[x^!])^!]$, but also terms like I1 or $\mathrm{I}[y, y]$ reducing to 0 because there is a mismatch between the number of linear resources needed by the functional part of the application and the number of resources it actually receives.

Pagani and Tranquilli proved the confluence of the resource calculus through an appropriate Standardization Theorem [PT09].

Theorem 2.2.3 (Confluence, Pagani and Tranquilli [PT09])**.**
The β_{r}-reduction is confluent on Λ^r.

The resource calculus is *intensional*. Indeed, just like in the regular λ-calculus, there are different programs having the same extensional behaviour. The analogue of η-reduction for the resource calculus is defined as the contextual closure of the following rule:

$$\lambda x.M[x^!] \to_{\eta_{\mathsf{r}}} M, \text{ if } x \notin \mathrm{FV}(M). \qquad (\eta_{\mathsf{r}})$$

Solvability has been studied by Pagani and Ronchi della Rocca in [PR10a, PR10b]. As in the resource calculus terms appear in formal sums, various notions of solvability can arise, depending on the meaning associated with the sum. The interpretation of the $+$ as an inner choice operator corresponding to an angelic non-determinism arises naturally, because of the definition of 0 as the neutral element of the sum. However, the sum can be also seen as a parallel composition corresponding to a demonic non-determinism.

Definition 2.2.4. *A closed resource term M is:*

- may-solvable *whenever there are closed bags $P_1, \dots, P_n \in \Lambda^b$ such that $M\vec{P} =_{\beta_{\mathsf{r}}} \mathrm{I} + \mathbf{N}$, for some $\mathbf{N} \in \mathbb{N}\langle\Lambda^r\rangle$ (possibly $\mathbf{N} := 0$).*
- must-solvable *if there are closed bags $P_1, \dots, P_n \in \Lambda^b$ such that $M\vec{P} =_{\beta_{\mathsf{r}}} \mathrm{I} + \cdots + \mathrm{I}$.*

As in the formal sum of resource terms repetitions do matter, other notions of solvability may arise depending on the number of times one gets the identity I.

As in λ-calculus the solvability can be characterized in terms of head-normalization, the may-solvability of the resource calculus can be captured by "outer-normalization". The outer-reduction $\to_o$, first defined in [PT09], corresponds to an evaluation where no reduction is made inside reusable resources. We say that a sum $\mathbf{M}$ of resource terms is in *may-outer normal form* whenever it is of the form $M + \mathbf{N}$ with $M \not\to_o \mathbf{M}'$. Pagani and Ronchi della Rocca proved that the may-outer normal form of $\mathbf{M}$, if it exists, can be reached by outer-reduction and that this is equivalent to ask that $\mathbf{M}$ is may-solvable. Inspired by the relational semantics of λ-calculus, they also introduced a type system $\vdash_{\mathsf{m}}$ based on non-idempotent intersection types in which a resource term M is typable if and only if it possesses a may-outer normal form. Summing up, the following theorem holds.

Theorem 2.2.5 (Pagani, Ronchi della Rocca [PR10a, PR10b])**.**
Given a closed resource term M, the following are equivalent:

1. *M is may-solvable,*
2. *M is reducible to a may-outer-normal form (by outer reduction),*
3. *$\vdash_{\mathsf{m}} M : \sigma$ for some non-idempotent intersection type σ.*

Problem 6. *Is it possible to characterize the must-solvability of the resource calculus in terms of outer-reduction and in terms of typability in a suitable type system?*

A partial answer is given in [PR10a] for the fragment of Λ^r corresponding to the demonic non-deterministic λ-calculus (therefore without linear resources).

2.3 THE TAYLOR EXPANSION

The *finite resource calculus* is the fragment of resource calculus restricted to linear resources. In other words, every bag has the multiset of reusable resources which is empty. The terms (resp. bags) of this fragment are called *finite* and their set is denoted by Λ_{f}^{r} (resp. Λ_{f}^{b}). This sub-calculus is closed under β_{r}-reduction, while η_{r}-reduction does not play any role. As linear resources cannot be duplicated, every reduction strictly decreases the size of a term and the system is strongly normalizable [ER08] even if no typing is involved.

Taylor expansion of ordinary λ-terms. The Taylor expansion was originally introduced in [ER03] in the context of λ-calculus as a translation developing every application as an infinite series of finite applications with rational coefficients. For our purposes, it is enough to present the Taylor expansion as a map $\mathscr{T}(\cdot)$ from Λ to possibly infinite sets of finite resource terms. This simplified version corresponds to the support[2] of the original Taylor expansion. Given $M \in \Lambda$, the *Taylor expansion* $\mathscr{T}(M)$ *of* M is defined by induction as:

$$\begin{array}{rcl} \mathscr{T}(x) & = & \{x\}, \\ \mathscr{T}(\lambda x.M) & = & \{\lambda x.t \mid t \in \mathscr{T}(M)\}, \\ \mathscr{T}(MN) & = & \{tb \mid t \in \mathscr{T}(M), b \in \mathcal{M}_{\mathrm{f}}(\mathscr{T}(N))\}. \end{array}$$

The intuition is that in the λ-term MN the function M can either erase its argument N or use it a certain amount of times during its execution. As we cannot determine this number statically, we consider all possibilities henceforth the infinite set. Intuitively, the Taylor expansion $\mathscr{T}(M)$ can be seen as a resource sensitive version of $\mathsf{App}(M)$, where the approximants keep track of the resources. Note that the resource approximants may not be β_{r}-normal, but are strongly normalizable, hence $\mathrm{NF}_{\beta_{\mathsf{r}}}(\mathscr{T}(M)) = \{\mathrm{NF}_{\beta_{\mathsf{r}}}(t) \mid t \in \mathscr{T}(M)\}$.

The following result formalizes the intuition above.

Theorem 2.3.1 (Ehrhard and Regnier [ER08])**.**
Given $M, N \in \Lambda$, we have $\mathrm{BT}(M) = \mathrm{BT}(N)$ if and only if $\mathrm{NF}_{\beta_{\mathsf{r}}}(\mathscr{T}(M)) = \mathrm{NF}_{\beta_{\mathsf{r}}}(\mathscr{T}(N))$.

Taylor expansion of resource λ-terms. If in the study of usual λ-calculus we can choose to use the Böhm trees or the Taylor expansion, depending on whether we need a grasp on resource usage, the situation is more constrained when studying the resource calculus. Indeed, at the moment, no convincing notion of Böhm tree has been developed for this calculus: first because one should consider forests of trees (to treat the non-determinism), second because one would have an alternation of commutative and non-commutative branches (to treat the multisets). The resulting structure would not be easy to handle.

For this reason, in collaboration with Pagani, in [M19] we generalized the notion of Taylor expansion to terms and bags of the resource calculus. We adopt for sets $X \subseteq \Lambda^{r}$ (resp. $Y, Z \subseteq \Lambda_{\mathrm{f}}^{b}$) the same abbreviations as introduced for finite sums, that is $\lambda x.X := \{\lambda x.M \mid M \in X\}$, $XZ := \{MP \mid M \in X, P \in Z\}$ and $Y \cdot Z := \{P_1 \cdot P_2 \mid P_1 \in Y,\ P_2 \in Z\}$.

Definition 2.3.2. *The* Taylor expansion *of a resource term $M \in \Lambda^{r}$ is the set $\mathscr{T}(M) \subseteq \Lambda_{\mathrm{f}}^{r}$ defined in Figure 2.2. It is extended to sums as follows $\mathscr{T}(\sum_i M_i) = \bigcup_i \mathscr{T}(M_i)$.*

Problem 7. *Is it possible to define a convincing notion of Böhm tree for the full resource calculus?*

[2] I.e., the set of those finite terms appearing in the series with a non-zero coefficient.

$$\mathscr{T}(x) = \{x\}, \qquad \mathscr{T}(\lambda x.M) = \lambda x.\mathscr{T}(M), \qquad \mathscr{T}(MP) = \mathscr{T}(M)\mathscr{T}(P),$$
$$\mathscr{T}([\mathbf{M}^!]) = \mathcal{M}_{\mathrm{f}}(\mathscr{T}(\mathbf{M})), \qquad \mathscr{T}([M] \cdot P) = [\mathscr{T}(M)] \cdot \mathscr{T}(P).$$

Figure 2.2: The Taylor expansion $\mathscr{T}(M)$ of a resource term M.

2.4 BÖHM THEOREM FOR RESOURCE CALCULUS

In collaboration with Pagani, we studied the separability of resource terms in [M19]. Our notion of separability is inspired by the relational semantics of resource calculus (see §2.7). Semantically $M + M$ is indistinguishable from M, so we assume that the $+$ is idempotent.

The ordinary notion of separability is too strong for this calculus, as there are resource terms like $\lambda xy.x[y^!]$ and $\lambda xy.x[y]$ that cannot be separated by sending them to different projections, but they can be separated by sending the former to I and the latter to 0.

Definition 2.4.1 (Separability)**.** *Two closed resource terms M, N are* separable *if there are closed bags $P_1, \dots, P_k \in \Lambda^b$ such that $MP_1 \cdots P_k =_{\beta_{\mathsf{r}}} \mathrm{I}$ while $NP_1 \cdots P_k =_{\beta_{\mathsf{r}}} 0$.*

Following the steps of Böhm in his pioneering work [Böh68], we focus our attention on normal forms, which are easier to separate. However, in contrast with what happens in the λ-calculus, there are β_{r}-normal resource terms that cannot be separated even if they are η_{r}-different. The typical example is given by $\lambda xyz.x[(z[y^!])^!]$ and $\lambda xyz.x[(z1 + z[y, y^!])^!]$, the idea being that the ω-copies of y in the former are simulated in the latter as a erratic choice between *zero* and *at least one* copies. The attentive reader will notice that these terms have the same Taylor expansion, and actually that is the reason why they cannot be separated.

Because of its infinitary nature, the property of having the same Taylor expansion looks rather semantical than syntactical. This property can be however characterized syntactically by proving that $\mathscr{T}(M) = \mathscr{T}(N)$ if and only if $M \equiv^{\mathscr{T}} N$ where $\equiv^{\mathscr{T}}$ is the congruence generated by setting $[x^!] = 1 + [x, x^!]$. We let $\equiv^{\mathscr{T}}_{\eta_{\mathsf{r}}}$ be the congruence generated by $\equiv^{\mathscr{T}} \cup =_{\eta_{\mathsf{r}}}$.

Theorem 2.4.2 (Resource Böhm Theorem, Manzonetto and Pagani [M19])**.**
Let $M, N \in \Lambda^r$ be two closed resource terms in β_{r}-normal form. If $M \not\equiv^{\mathscr{T}}_{\eta_{\mathsf{r}}} N$ then there are closed bags $P_1, \dots, P_k \in \Lambda^b$ such that

$$M\vec{P} =_{\beta_{\mathsf{r}}} \mathrm{I}, \qquad N\vec{P} =_{\beta_{\mathsf{r}}} 0. \qquad \text{(or } \textit{vice versa}\text{)}$$

To obtain such a result we developed in [M19] a refined version of the Böhm-out technique. Note that a crucial ingredient in the classic proof of the Böhm Theorem is the fact that it is possible to erase subterms in order extract from the λ-terms their structural difference. This is not an easy task in Λ^r, because the linear resources must be consumed and cannot be erased. In this respect, our technique has some similarities with the one developed to achieve the separation in the λI-calculus [Bar84, §10.5]. Moreover, since the argument of an application is a bag of resources, comparing a difference between two resource terms may turn into comparing the differences between two multisets of terms, and this problem presents analogies with that of separating a finite set of λ-terms [BDPR79].

2.5 Differential Categories and Cartesian Closedness

Despite the fact that the differential λ-calculus is born from semantical considerations, namely Ehrhard's deep analysis of coherent spaces, it took some years to provide a general categorical definition of its models. Inspired by Ehrhard's work on differential λ-calculus, Blute, Cockett and Seely investigated the problem of axiomatizing a derivative operator $\mathsf{D}(-)$ in the context of category theory. The authors started working in the monoidal setting and proposed a notion of *monoidal differential category* [BCS06]. By applying the co-Keisli construction, one obtains a Cartesian category having a differential operator.

Cartesian differential categories were introduced in [BCS09] to provide a direct axiomatization of the derivative operator in the Cartesian setting. The derivative of a morphism $f : A \to B$ will be a morphism $\mathsf{D}(f) : A \times A \to B$ linear in its first component.

$$\frac{f : A \to B}{\mathsf{D}(f) : A \times A \to B}\ (\mathsf{D})$$

In order to model differentiation in a category, we need to be able to sum its morphisms. Therefore, we consider *left-additive* categories, where each homset $\mathrm{Hom}(A, B)$ has a structure of a commutative monoid and precomposition by any map preserves such a structure:

$$f;(g+h) = (f;g) + (f;h) \qquad\qquad f;0 = 0.$$

A morphism f of a left-additive category is called *additive* if postcomposition by f also preserves the commutative monoid structure.

A left-additive category is *Cartesian* if it has finite products, the projections are additive, and the pairing $\langle -, - \rangle$ of additive morphisms is itself additive.

As expected, a morphism f is called *linear* whenever its derivative is constant, and all linear morphisms are additive (while the converse does not hold in general).

Definition 2.5.1. *A* Cartesian differential category *is a Cartesian left-additive category having an operator* $\mathsf{D}(-)$ *that sends every* $f : A \to B$ *into a morphism* $\mathsf{D}(f) : A \times A \to B$ *and satisfying:*

(D1) $\mathsf{D}(f+g) = \mathsf{D}(f) + \mathsf{D}(g)$ *and* $\mathsf{D}(0) = 0$,

(D2) $\langle h+k, v\rangle; \mathsf{D}(f) = \langle h, v\rangle; \mathsf{D}(f) + \langle k, v\rangle; \mathsf{D}(f)$ *and* $\langle 0, v\rangle; \mathsf{D}(f) = 0$,

(D3) $\mathsf{D}(\mathrm{Id}) = \pi_1$, $\mathsf{D}(\pi_1) = \pi_1;\pi_1$ *and* $\mathsf{D}(\pi_2) = \pi_2;\pi_1$,

(D4) $\mathsf{D}(\langle f, g\rangle) = \langle \mathsf{D}(f), \mathsf{D}(g)\rangle$,

(D5) $\mathsf{D}(g; f) = \langle \mathsf{D}(g), \pi_2; g\rangle; \mathsf{D}(f)$,

(D6) $\langle\langle g, 0\rangle, \langle h, k\rangle\rangle; \mathsf{D}(\mathsf{D}(f)) = \langle g, k\rangle; \mathsf{D}(f)$,

(D7) $\langle\langle 0, h\rangle, \langle g, k\rangle\rangle; \mathsf{D}(\mathsf{D}(f)) = \langle\langle 0, g\rangle, \langle h, k\rangle\rangle; \mathsf{D}(\mathsf{D}(f))$.

Therefore, intuitively the axiom (D1) states that the operator $\mathsf{D}(-)$ is linear; (D2) says that $\mathsf{D}(-)$ is additive in its first coordinate; (D3) and (D4) ask that $\mathsf{D}(-)$ behaves coherently with the product structure; (D5) is the usual chain rule; (D6) requires that $\mathsf{D}(f)$ is linear in its first component. (D7) states the independence of the order of "partial differentiation". Indeed, in these categories *partial derivatives* are obtained from the full ones by "zeroing out" the components on which the differentiation is not required. For example, the partial derivative of a morphism $f : C \times A \to B$ on its first component C can be defined by setting

$$\mathsf{D}_1(f) = C \times (C \times A) \xrightarrow{\langle \mathrm{Id}, 0\rangle \times \mathrm{Id}} (C \times A) \times (C \times A) \xrightarrow{\mathsf{D}(f)} C \times (C \times A) \to B.$$

Cartesian closed structure. In [BCS09], Blute, Cockett and Seely show that Cartesian differential categories are sound and complete models of a suitable term calculus. This calculus has a derivative operator in the spirit of Ehrhard's differential λ-calculus, but possesses no λ-abstraction operator. It is therefore unlikely that these categories are well suited for modeling the resource calculus, that contains the whole λ-calculus as a subsystem. Indeed, in order to provide a categorical interpretation of λ-abstraction we need to have an isomorphism $\mathrm{Hom}(C \times A, B) \cong \mathrm{Hom}(C, [A \to B])$ natural in both C and B, that is to say a Cartesian closed structure. However, it is not enough to have a Cartesian differential category which is moreover Cartesian closed to interpret the resource calculus. For this reason, in collaboration with Bucciarelli and Ehrhard, we introduced the notion of "Cartesian closed differential category" (called *differential λ-category* in [M5]).

The starting point is a Cartesian left-additive category $\mathbf{C}$ which is in addition Cartesian closed and such that the currying $\Lambda(-)$ satisfies the following axioms:

$$(\mathrm{Curry}_+) \qquad \Lambda(f+g) = \Lambda(f)+\Lambda(g) \qquad\qquad \Lambda(0) = 0 \qquad (\mathrm{Curry}_0)$$

In this case, we say that $\mathbf{C}$ is *Cartesian closed left-additive*. The axioms (Curry_+) and (Curry_0) are needed to ensure that currying $\Lambda(-)$ is linear. From these properties, it follows that the evaluation map $\mathrm{Eval} : [A \to B] \times A \to B$ is additive in its left component, as expected.

We ensure that the differential operator behaves correctly with respect to the Cartesian closed structure by requiring the additional axiom $(\mathrm{Curry}_\mathsf{D})$ below.

Definition 2.5.2. *A* Cartesian closed differential category *is a Cartesian differential category which is Cartesian closed left-additive and such that, for all $f : C \times A \to B$:*

$$(\mathrm{Curry}_\mathsf{D}) \quad \mathsf{D}(\Lambda(f)) = \langle \pi_1 \times 0_A, \pi_2 \times \mathrm{Id}_A \rangle ; \Lambda(\mathsf{D}(f)).$$

In a Cartesian closed differential category it is possible to define an operator $\star$ that can be seen as the categorical counterpart of the linear substitution (Definition 2.2.1(2)):

$$\frac{f : C \times A \to B \quad g : C \to A}{f \star g : C \times A \to B} \ (\star)$$

where $f \star g = \langle \langle 0_C^{C \times A}, \pi_1 \rangle ; g, \mathrm{Id}_{C \times A} \rangle ; \mathsf{D}(f)$. Intuitively, the morphism $f \star g$ is obtained by force-feeding the second argument A of f with *one copy* of the result of g. Notice that the type is not modified because the morphism $f \star g$ may still depend on its argument A.

Cartesian closed differential categories provide a sound model of the simply typed resource calculus, in which the sums and the elements of the bags are uniformly typed:

$$\frac{}{\Delta \vdash 0 : A} \qquad \frac{\Delta \vdash M : A \quad \Delta \vdash \mathbf{M} : A}{\Delta \vdash M + \mathbf{M} : A} \qquad \frac{\Delta \vdash M : A \to B \quad \Delta \vdash P : A}{\Delta \vdash MP : B}$$

$$\frac{}{\Delta \vdash 1 : A} \qquad \frac{\Delta \vdash L : A \quad \Delta \vdash P : A}{\Delta \vdash [L] \cdot P : A} \qquad \frac{\Delta \vdash \mathbf{M} : A \quad \Delta \vdash P : A}{\Delta \vdash [\mathbf{M}^!] \cdot P : A}$$

Exploting the left-additive structure of a Cartesian closed differential category and the operator $\star$ above, it is not difficult to define the categorical interpretation of a judgement $x_1 : A_1, \ldots, x_n : A_n \vdash \mathbf{M} : B$ as a morphism $[\![\mathbf{M}]\!]^\Delta : A_1 \times \cdots \times A_n \to B$.

Theorem 2.5.3 (Bucciarelli et Al. [M5])**.** *Cartesian closed differential categories constitute sound models of the simply typed resource calculus.*

2.6 Categorical Models of Resource Calculus

The notion of Cartesian closed differential category is general enough to encompass many examples given in the literature, like Ehrhard's finiteness spaces [Ehr05], Blute, Ehrhard and Tasson's convenient categories [BET12], Kerjean and Tasson's Mackey-complete spaces [KT17], Harmer and McCusker's category of non-deterministic games [HM99] and the relational semantics. However, to interpret the untyped resource calculus we need reflexive objects, while the categories presented in [Ehr05, BET12, KT17] do not possess any.

Linear reflexive objects. The resource calculus can be interpreted in a reflexive object $\mathcal{U}$ of a Cartesian closed differential category provided that $\mathcal{U}$ satisfies some linearity conditions. Indeed, as the λ-abstraction of resource calculus is a linear operator, we need that such a property is preserved by the morphisms performing the retraction.

Definition 2.6.1. *A reflexive object $\mathcal{U} = (U, \mathrm{app}, \mathrm{abs})$ of a Cartesian closed differential category is* linear *whenever the morphisms $\mathrm{abs} : [U \to U] \to U$ and $\mathrm{app} : [U \to U] \to U$ are linear.*

The *interpretation of a resource term M in a linear reflexive object $\mathcal{U}$ with respect to a sequence $\vec{x} \supseteq \mathrm{FV}(M)$ of length n* is a morphism $[\![M]\!]^{\mathcal{U}}_{\vec{x}} : U^n \to U$ defined by extending the usual categorical interpretation of λ-terms as follows:

$$[\![M[L_1, \dots, L_n, \mathbf{N}^!]]\!]^{\mathcal{U}}_{\vec{x}} = \langle \Lambda((\cdots(\Lambda^-([\![M]\!]^{\mathcal{U}}_{\vec{x}}; \mathrm{app}) \star [\![L_1]\!]^{\mathcal{U}}_{\vec{x}}) \cdots) \star [\![L_n]\!]^{\mathcal{U}}_{\vec{x}}), [\![\mathbf{N}]\!]^{\mathcal{U}}_{\vec{x}} \rangle; \mathrm{Eval}.$$

The interpretation is extended to sums by linearity: $[\![M_1 + \cdots + M_k]\!]^{\mathcal{U}}_{\vec{x}} = [\![M_1]\!]^{\mathcal{U}}_{\vec{x}} + \cdots + [\![M_k]\!]^{\mathcal{U}}_{\vec{x}}$.

Theorem 2.6.2 (Manzonetto [M18])**.**
Linear reflexive objects in differential Cartesian closed categories constitute sound models of the resource calculus.

Modeling the Taylor Expansion. Given a linear reflexive object $\mathcal{U}$ living in a Cartesian closed differential category $\mathbf{C}$ it is interesting to check whether all resource terms having the same Taylor expansion get the same interpretation. As an interesting fact, this happens to be a property of the category rather than of the reflexive object under consideration. Since the definition of the Taylor expansion asks for infinite power series, we consider Cartesian closed differential categories $\mathbf{C}$ where it is possible to add infinitely many morphisms. Formally, we require that for every countable set I and every family $\{f_i\}_{i \in I}$ of morphisms $f_i : A \to B$ there exists a morphism $\sum_{i \in I} f_i \in \mathrm{Hom}(A, B)$ and, in this case, we say that $\mathbf{C}$ *has countable sums*.

Remember that Definition 2.3.2 works under the assumption of an idempotent sum, therefore we suppose that also our sums on the morphisms of the category is idempotent.

Definition 2.6.3. *A Cartesian closed differential category* models the Taylor Expansion *if it has countable sums and the following axiom holds (for every $f : C \times A \to B$ and $g : C \to A$):*

$$\mathrm{Eval}; \langle f, g \rangle = \sum_{k \in \mathbb{N}} ((\cdots(\Lambda^-(f) \underbrace{\star\, g) \cdots) \star g}_{k \text{ times}}); \langle \mathrm{Id}, 0 \rangle. \tag{Taylor}$$

As a consequence, all models living in a category modeling the Taylor expansion equate all resource terms having the same Taylor expansion. From Theorem 2.3.1 it follows that all λ-terms having the same Böhm trees get the same interpretation in such models.

Equational Completeness. An important result concerning λ-calculus is the *equational completeness theorem* proved by Scott in [Sco80] and subsequently refined by Koymans in [Koy82]. The theorem states that every λ-theory $\mathcal{T}$ is the theory of a reflexive object $\mathcal{U}$ in a Cartesian closed category $\mathbf{C}$. In other words the completeness theorem guarantees that the definition of model of λ-calculus under consideration is general enough to represent all λ-theories. This classical result is achieved in two steps:

(i) given a λ-theory $\mathcal{T}$ one proves that the set $\Lambda_{\mathcal{T}}$ of λ-terms modulo $\mathcal{T}$ together with the application operator defined between equivalence classes constitutes a λ-model $\mathcal{M}_{\mathcal{T}}$ (called *the term model of* $\mathcal{T}$) having as theory exactly $\mathcal{T}$;

(ii) by applying to $\mathcal{M}_{\mathcal{T}}$ a construction called *Karoubi envelope* [Kar78], which consists in splitting the idempotents, one builds a (very syntactic) Cartesian closed category $\mathbf{C}_{\mathcal{T}}$ in which the identity I is a reflexive object such that $\mathrm{Th}(\mathrm{I}) = \mathcal{T}$.

Summing up, the idea of the proof is to find suitable λ-terms to encode the structure of the category (pairing, currying, evaluation, and the like) and prove that they actually define a category with a Cartesian closed structure.

In [M18], we investigated the question whether the categorical notion of model presented above is complete. Using the resource calculus, it is possible to define a syntactic Cartesian closed differential category. On the one hand, the categorical operator $\mathsf{D}(-)$ is unproblematic since it can be easily defined in terms of linear application. On the other hand, the encoding of categorical pairing $\langle f, g\rangle$ used by Scott is not additive. Indeed, such pairing is defined starting from Church's encoding of the pair in λ-calculus given by $\langle f, g\rangle := \lambda x.xfg$ with projections $\pi_1 := \mathrm{K}$, $\pi_2 := \mathrm{K}'$. Obviously with this definition we have $\langle f_1 + f_2, g_1 + g_2\rangle \neq \langle f_1, g_1\rangle + \langle f_2, g_2\rangle$ since the sums do not occur in linear position[3].

This problem can be circumvented by providing a different encoding of the pairing in the resource calculus:

$$\langle\!\langle M, N\rangle\!\rangle := \lambda x.x[\lambda y.M] + \lambda x.x[\lambda y.y[N]], \text{ for some } y \notin \mathrm{FV}(M, N),$$

with projections $\pi_1 := \lambda x.x1$ and $\pi_2 := \lambda x.x[\mathrm{I}]$. This encoding is inspired by the set-theoretical definition of ordered pair: intuitively the pair of M, N is the set containing M, N (the sum is representing the union) where the elements have been slightly modified in order to distinguish them. In our case, such a distinction consists in the number of linear resources they can receive (zero for the first component and one for the second).

With this encoding we are able to show that every equational theory $\mathcal{T}$ of the resource calculus gives rise to a category $\mathbf{C}_{\mathcal{T}}$ with a structure of differential Cartesian closed category, under the assumption that the sum is idempotent (like set-theoretical union) and that the Karoubi envelope splits idempotents which are linear.

Theorem 2.6.4 (Manzonetto [M18])**.**
For all equational theories $\mathcal{T}$ of the resource calculus satisfying sum idempotency we have:

(i) $\mathbf{C}_{\mathcal{T}}$ *is a differential Cartesian closed category,*

(ii) the identity I *is a linear reflexive object.*

Problem 8. *Is it possible to prove an equational completeness theorem for categorical models of resource calculus without assuming an idempotent sum? (See [CG16] for preliminary investigations, notice however that the authors prove completeness for another* ad hoc *differential calculus.)*

[3] Recall that in λ-calculus the λ-abstraction is linear but the application is only linear in its left-component.

2.7 Relational Models of Resource Calculus

We mentioned in Section 1.3 that **MRel** is a Cartesian closed category. Notice that since the Cartesian structure is given by the disjoint union, which will be denoted by $\&$, there exists a Seely isomorphism between $\mathcal{M}_{\mathrm{f}}(A\&B)$ and $\mathcal{M}_{\mathrm{f}}(A) \times \mathcal{M}_{\mathrm{f}}(B)$. In particular, a morphism $f : A\&B \to C$ will be systematically presented as a relation $f \subseteq \mathcal{M}_{\mathrm{f}}(A) \times \mathcal{M}_{\mathrm{f}}(B) \times C$.

Cartesian differential structure. We are now going to expose the differential structure of **MRel**. As relations are closed under arbitrary union, every homset can be endowed with a structure $(\mathbf{MRel}(A, B), \cup, \emptyset)$ of commutative monoid in such a way that precomposition by any map preserves this structure. It is easy to check that pairing preserves additivity and that currying is linear, therefore the category is Cartesian closed left-additive and has countable sums. Moreover, given a morphism $f : A \to B$ its derivative is given by:

$$\mathsf{D}(f) = \{(([\alpha], a), \beta) \mid ([\alpha] \uplus a, \beta) \in f\} : A\&A \to B.$$

Again, it is not difficult to check that the operator $\mathsf{D}(-)$ satisfies all the axioms of a Cartesian closed differential category. Summing up, the following theorem holds.

Theorem 2.7.1 (Manzonetto [M18])**.**
The category **MRel** *is a Cartesian closed differential category.*

Relational models of resource calculus. In the category **MRel**, a morphism $f : A \to B$ is linear when its behaviour is specified on the singleton multisets of A, that is $(a, \alpha) \in f$ entails that the multiset a is a singleton. Thus, every reflexive object $\mathcal{U}$ having abs and app satisfying this property is linear. As a consequence we get that every rgm $\mathcal{D} = (D, i)$ is linear, since the induced retraction pair is of the following form:

- $\mathrm{abs} = \{([(a, \alpha)], a \to \alpha) \mid a \in \mathcal{M}_{\mathrm{f}}(D), \alpha \in D\} : [D \to D] \to D$,
- $\mathrm{app} = \{([a \to \alpha], (a, \alpha)) \mid a \in \mathcal{M}_{\mathrm{f}}(D), \alpha \in D\} : D \to [D \to D]$,

where $a \to \alpha$ is a notation for $i(a, \alpha)$ following the intuition given in Section 1.3. By Theorem 2.6.2 rgms constitute sound models of the untyped resource calculus.

Definition 2.7.2. *The interpretation of resource terms $M \in \Lambda^r$, sums of terms $\mathbf{M} \in \mathbb{N}\langle\Lambda^r\rangle$ and bags $P \in \Lambda^b$ in an rgm $\mathcal{D}$ (with respect to a sequence $\vec{x}$ including their free variables) are morphisms $[\![M]\!]^{\mathcal{D}}_{\vec{x}}, [\![\mathbf{M}]\!]^{\mathcal{D}}_{\vec{x}} : \mathcal{D}^{\vec{x}} \to \mathcal{D}$, $[\![P]\!]^{\mathcal{D}}_{\vec{x}} : \mathcal{D}^{\vec{x}} \to \mathcal{M}_{\mathrm{f}}(\mathcal{D})$ defined by mutual induction as follows:*

- $[\![M_1 + \cdots + M_k]\!]^{\mathcal{D}}_{\vec{x}} = [\![M_1]\!]^{\mathcal{D}}_{\vec{x}} \cup \cdots \cup [\![M_k]\!]^{\mathcal{D}}_{\vec{x}}$,
- $[\![x_i]\!]^{\mathcal{D}}_{\vec{x}} = \{(([], \ldots, [], [\alpha], [], \ldots, []), \alpha) \mid \alpha \in \mathcal{D}\}$, *where $[\alpha]$ stands in i-th position,*
- $[\![\lambda y.M]\!]^{\mathcal{D}}_{\vec{x}} = \{(\vec{a}, b \to \alpha) \mid ((\vec{a}, b), \alpha) \in [\![M]\!]^{\mathcal{D}}_{\vec{x},y}\}$, *where we suppose that $y \notin \vec{x}$,*
- $[\![MP]\!]^{\mathcal{D}}_{\vec{x}} = \{(\vec{a}_1 \uplus \vec{a}_2, \alpha) \mid \exists b \in \mathcal{M}_{\mathrm{f}}(\mathcal{D})\ (\vec{a}_1, b \to \alpha) \in [\![M]\!]^{\mathcal{D}}_{\vec{x}},\ (\vec{a}_2, b) \in [\![P]\!]^{\mathcal{D}}_{\vec{x}}\}$,
- $[\![[L_1, \ldots, L_k, \mathbf{N}^!]]\!]^{\mathcal{D}}_{\vec{x}} = \{(\uplus_{r=1}^{k+m} \vec{a}_r, [\beta_1, \ldots, \beta_{k+m}]) \mid (\vec{a}_j, \beta_j) \in [\![L_j]\!]^{\mathcal{D}}_{\vec{x}}, 1 \leq j \leq k$ *and* $(\vec{a}_i, \beta_i) \in [\![\mathbf{N}]\!]^{\mathcal{D}}_{\vec{x}},\ k < i \leq k + m\}$,

where $\vec{a}$ denotes a sequence of multisets $(a_1, \ldots, a_n)$ and, given a sequence $\vec{b} = (b_1, \ldots, b_n)$ having the same length, $\vec{a} \uplus \vec{b}$ denotes the pointwise multiset union $(a_1 \uplus b_1, \ldots, a_n \uplus b_n)$.

By structural induction on M, one can verify that the analogue of Theorem 1.3.2 holds.

Theorem 2.7.3 (The Approximation Theorem for Taylor Expansion [M18])**.**
Given a relational graph model $\mathcal{D}$ and a closed resource term M, we have that:

$$\sigma \in [\![M]\!]^{\mathcal{D}} \iff \exists t \in \mathscr{T}(M) \text{ such that } \sigma \in [\![t]\!]^{\mathcal{D}}.$$

2.8 $\mathcal{D}_\omega$ IS NOT FULLY ABSTRACT FOR THE RESOURCE CALCULUS

The original relational analogue of Scott's $\mathcal{D}_\infty$ was defined in [M3] as part of an investigation on non-well-pointed semantics of λ-calculus. The idea is to start building a family of sets $(D_n)_{n\in\mathbb{N}}$ by induction on n:

- $D_0 = \emptyset$,
- $D_{n+1} = \mathcal{M}_{\mathrm{f}}(D_n)^{(\omega)}$,

where $\mathcal{M}_{\mathrm{f}}(A)^{(\omega)}$ denotes the set of infinite sequences $(a_0, a_1, \dots, a_i, \dots)$ of finite multisets of A satisfying the property $a_i \neq []$ only for a finite number of indices i. Since the operation $A \mapsto \mathcal{M}_{\mathrm{f}}(A)^{(\omega)}$ is monotonic on sets, and since $D_0 \subseteq D_1$, we have $D_n \subseteq D_{n+1}$ for all $n \in \mathbb{N}$.

The reflexive object is given by the union $D = \bigcup_{n\in\mathbb{N}} D_n$. Indeed, to define an isomorphism in $\mathbf{MRel}$ between D and $[D \to D]$ it is enough to remark that every element $\alpha = (a_0, a_1, a_2, \dots) \in D$ is canonically associated with the pair $(a_0, (a_1, a_2, \dots)) \in \mathcal{M}_{\mathrm{f}}(D) \times D$ and *vice versa*. It is easy to check that the reflexive object obtained by this construction is isomorphic to the rgm $\mathcal{D}_\omega$ having a single atom ε and generated by the equation $[] \to \varepsilon = \varepsilon$. Under this isomorphism the element ε corresponds to the infinite sequence $([], [], [], \dots)$.

Observational equivalence. By collecting the results in the previous sections we know that the model $\mathcal{D}_\omega$ is extensional and equates all resource terms having the same (normal form of the) Taylor expansion. In particular, $[\![M]\!]^{\mathcal{D}_\omega} \neq \emptyset$ exactly when M is may-solvable. Moreover, when considered as a model of λ-calculus, $\mathcal{D}_\omega$ is fully abstract for $\mathcal{H}^*$ [M17].

These considerations led us to conjecture in [M18] that the equality induced by $\mathcal{D}_\omega$ on resource terms coincides with the following observational equivalence, where we take as observables the may-outer normal forms of [PR10b].

Definition 2.8.1 (Observational Equivalence and Preorder)**.**
Given two closed $M, N \in \Lambda^r$, we set $M \equiv^{\mathrm{monf}} N$, whenever for all closed bags $P_1, \dots, P_n \in \Lambda^b$:

$$MP_1 \cdots P_n \text{ is may-solvable} \iff NP_1 \cdots P_n \text{ is may-solvable}$$

The observational preorder $\sqsubseteq^{\mathrm{monf}}$ *is defined by orienting the equivalence from left to right.*

The failure of full abstraction. Breuvart, during his PhD studies, found a counterexample to the full abstraction of $\mathcal{D}_\omega$ for $\equiv^{\mathrm{monf}}$, a result subsequently published in [Bre13].

The key idea is to exhibit a resource term A that is observationally above the identity I, but whose interpretation in $\mathcal{D}_\omega$ does not contain $[\varepsilon] \to \varepsilon$ (while, obviously, $[\varepsilon] \to \varepsilon \in [\![\mathrm{I}]\!]^{\mathcal{D}_\omega}$). Using a fixed point combinator, it is possible to define a term A such that, for all k:

$$\mathrm{A} =_{\beta_{\mathrm{r}}} \Sigma_{n=1}^{k} \mathrm{A}_n + \mathbf{A}' \qquad \text{where } \mathrm{A}_n = \lambda y_1 \dots y_n x.x[\mathrm{I}[y_1^!] \cdots [y_n^!]].$$

Clearly, if $\mathrm{I}P_1 \cdots P_n$ is solvable then also $\mathrm{A}P_1 \cdots P_n =_{\beta_{\mathrm{r}}} \mathrm{A}_n P_1 \cdots P_n + \mathbf{A}'P_1 \cdots P_n$ is solvable. On the other side, the interpretation $[\![\mathrm{A}]\!]^{\mathcal{D}_\omega} = \bigcup_{n\in\mathbb{N}} [\![\mathrm{A}_n]\!]^{\mathcal{D}_\omega}$ is incomparable with the interpretation of the identity since none of the A_n's contains $[\varepsilon] \to \varepsilon$ in its semantics. A similar reasoning shows that $\mathrm{A} \equiv^{\mathrm{monf}} \mathrm{I}[\mathrm{I}^!, \mathrm{A}^!]$ while $[\![\mathrm{A}]\!] \neq [\![\mathrm{I}[\mathrm{I}^!, \mathrm{A}^!]]\!]$.

Theorem 2.8.2 (Breuvart [Bre13])**.**
The relational model $\mathcal{D}_\omega$ of resource calculus is not fully abstract for $\equiv^{\mathrm{monf}}$ (nor for $\sqsubseteq^{\mathrm{monf}}$).

Problem 9. *Show that the relational semantics does not contain any model fully abstract for the resource calculus.*

2.9 Adding Convergency Tests to Achieve Full Abstraction

As shown by Breuvart, the applicative contexts of resource calculus are not powerful enough to discriminate all the terms which are different in $\mathcal{D}_\omega$. To achieve full abstraction we decided, together with Bucciarelli, Carraro and Ehrhard, to endow the resource calculus with new constructions, to be understood as implementing a very simple exception mechanism, and with a "must" parallel composition [M1]. This is part of a tradition that consists in varying the language to fit an intended model: for instance, Plotkin needed to extend PCF with a *parallel or* in order to find a fully abstract continuous model [Plo77].

Convergency tests. The *resource calculus with tests* [M1, M2] is obtained by extending the syntax of resource calculus as follows:

$$(\Lambda^{\bar\tau}) \qquad M, N, L ::= \cdots \mid \bar\tau(Q) \qquad\qquad (\Lambda^{\tau}) \qquad Q, R ::= \tau[L_1, \ldots, L_k]$$

where $\Lambda^{\bar\tau}$ is the set of terms (which includes Λ^r) and Λ^τ is the set of (convergency) tests.

Tests are "corked" multisets of terms having only two possible outcomes: either *success*, which is given by the empty test $\tau 1$, or *failure*, given by the empty sum 0. Intuitively, the test $\tau[L_1, \ldots, L_k]$ represents the must-parallel composition of $\tau[L_1] \parallel \cdots \parallel \tau[L_k]$ and therefore it succeeds when all of its components succeed. Consequently, we will sometimes use the notation $\tau[L_1, \ldots, L_k] \parallel \tau[L_{k+1}, \ldots, L_n]$ for the test $\tau[L_1, \ldots, L_n]$.

The operator $\bar\tau(\cdot)$ allows to build a term out of a test: intuitively, the term $\bar\tau(Q)$ may be thought of as Q preceded by an infinite sequence of dummy λ-abstractions (like a μ-abstraction of Parigot's $\lambda\mu$-calculus [Par92] over un unnamed variable). Dually, the "cork construction" $\tau[L_1, \ldots, L_k]$ may be thought of as an operator applying to all its arguments an infinite sequence of empty bags. This suggests that it is sound to reduce $\tau[\bar\tau(Q)]$ to Q.

Hence the term $\bar\tau(Q)$ raises an exception encapsulating Q and the test $\tau[L_1, \ldots, L_k]$ catches the exception possibly raised by any of the L_i's and replaces L_i by the multiset of terms encapsulated in that exception. The context of the exception is thrown away by the dummy abstractions of $\bar\tau$ and the dummy applications of τ. A test needs to catch an exception in order to succeed; for instance, $\tau[M]$ fails as soon as M is a $\bar\tau$-free, closed term.

All the notions and notations of resource calculus are extended as follows.

- the *meta-syntax* concerning sums of terms $\mathbf{M}$ and sums of tests $\mathbf{Q}$:

$$\tau[\Sigma_i M_i] := \Sigma_i \tau[M_i], \qquad (\Sigma_i R_i) \parallel \mathbf{Q} := \Sigma_i R_i \parallel \mathbf{Q}, \qquad \bar\tau[\Sigma_i R_i] := \Sigma_i \bar\tau[R_i],$$

- the *substitution* $M\{N/x\}$ (defined as expected) and the *linear substitution* $M\langle N/x\rangle$:

$$\bar\tau(Q)\langle N/x\rangle = \bar\tau(Q\langle N/x\rangle), \quad \tau[L_1, \ldots, L_k]\langle N/x\rangle = \Sigma_{i=1}^k \tau[L_1, \ldots, L_i\langle N/x\rangle, \ldots, L_k],$$

- the reduction semantics extends (β_{r}) with the following rules:

$$\tau[\lambda x.M] \to_\tau \tau[M\{0/x\}], \quad \tau[\bar\tau(Q)] \to_\gamma Q, \quad \bar\tau(Q)[\mathbf{M}^!] \to_{\bar\tau} \bar\tau(Q), \quad \bar\tau(Q)([L]\cdot P) \to_{\bar\tau} 0.$$

In this framework the convergence of a term can be verified through suitable test-contexts. A *test-context* $C(\!|-|\!)$ is a test having a single occurrence of its *hole* $(\!|-|\!)$, appearing in term-position. We write $C(\!|M|\!)$ for the result of substituting M for the hole $(\!|-|\!)$ in C, possibly with capture of free variables. According with the intuition of such a test mechanism, we say that a test Q *converges*, notation $Q\downarrow$, if there exists a sum $\mathbf{Q}$ such that $Q \twoheadrightarrow_{\beta_{\mathsf{r}}\tau\bar\tau\gamma} \tau[] + \mathbf{Q}$.

Definition 2.9.1. *Two resource terms M, N with tests are* observationally equivalent*, written $M \equiv^\tau N$, whenever $\forall C(\!|-|\!)\ .\ C(\!|M|\!)\downarrow \iff C(\!|N|\!)\downarrow$.*

Interpreting the tests. At first sight, these convergency tests may seem an *ad hoc* syntactic tool with no logical content. In reality, this extension already appeared in a differential linear logic setting [EL10]: τ corresponds to the 0-ary tensor and $\bar{\tau}$ to the 0-ary par cells. The parallel composition of tests can be obtained by combining the mix rule of linear logic, if available, with the contraction rule.

The resource calculus with tests has a natural denotational interpretation in $\mathcal{D}_\omega$. Indeed, an element of $\mathcal{D}_\omega$ can be described as a finite tree alternating two kinds of layers:

- *multiplicative layers* where subtrees are indexed by natural numbers,
- *exponential layers* where subtrees are organized as non-empty multisets.

To be more precise, $\parr - ?$ (negative) pairs of layers alternate with $\otimes - !$ (positive) pairs, respecting a strict polarity discipline very much in the spirit of Ludics [Gir03]. The empty positive multiplicative tree corresponds to the empty tensor cell and the negative one to the empty par cell. The corresponding constructions τ, $\bar{\tau}$ are therefore quite easy to interpret.

Definition 2.9.2. *We extend Definition 2.7.2 by setting* $[\![\bar{\tau}(Q)]\!]_{\vec{x}} = \{(\vec{a}, \varepsilon) \mid \vec{a} \in [\![Q]\!]_{\vec{x}}\}$, *where* $[\![Q]\!]_{\vec{x}} \subseteq \mathcal{D}_\omega^{\vec{x}}$ *represents the interpretation of a test* Q *with respect to a sequence* $\vec{x}$ *in* $\mathcal{D}_\omega$ *satisfying:*

$$[\![\tau[M_1, \dots, M_k]]\!]_{\vec{x}} = \{\vec{a}_1 \uplus \dots \uplus \vec{a}_k \mid (\vec{a}_i, \varepsilon) \in [\![M_i]\!]_{\vec{x}}\}.$$

Keeping in mind the logical interpretation of the elements of $\mathcal{D}$, we are able to associate with each $\alpha \in \mathcal{D}_\omega$, a test-context $\alpha^+(\!|\cdot|\!)$ with a hole $(\!|\cdot|\!)$ for a term.

Definition 2.9.3. *For* $\alpha \in \mathcal{D}_\omega$ *of the form* $[\alpha_1^1, \dots, \alpha_{k_1}^1] \to \dots \to [\alpha_1^r, \dots, \alpha_{k_r}^r] \to \varepsilon$, *we define by mutual induction a term* α^- *and a test-context* $\alpha^+(\!|\cdot|\!)$ *by:*

- $\alpha^- = \lambda x_1 \dots x_r.\bar{\tau}(\|_{i=1}^r ((\alpha_1^i)^+(\!|x_i|\!) \parallel \dots \parallel (\alpha_{k_i}^i)^+(\!|x_i|\!)))$,
- $\alpha^+(\!|\cdot|\!) = \tau[(\!|\cdot|\!)[(\alpha_1^1)^-, \dots, (\alpha_{k_1}^1)^-] \cdots [(\alpha_1^r)^-, \dots, (\alpha_{k_r}^r)^-]]$.

The test-context $\alpha^+(\!|-|\!)$ and the term α^- are built in such a way that $[\![\alpha^+(\!|x|\!)]\!]_x = \{[\alpha]\}$ and $[\![\alpha^-]\!] = \{\alpha\}$, from which it follows that $\mathcal{D}_\omega$ has the finite definability property. Moreover, for all $\alpha \in \mathcal{D}_\omega$ we have that $\alpha^+(\!|\alpha^-|\!) \twoheadrightarrow_{\beta_r \tau \bar{\tau} \gamma} \varepsilon$, while $\alpha^+(\!|\beta^-|\!) \twoheadrightarrow_{\beta_r \tau \bar{\tau} \gamma} 0$ for all $\beta \neq \alpha$. From these properties and the fact that !-free terms are strongly normalizable we get:

Lemma 2.9.4. *Given a closed !-free term* M *of resource calculus with tests, we have:*

$$\alpha \in [\![M]\!] \iff \alpha^+(\!|M|\!)\downarrow$$

As a consequence, we derive easily a full abstraction result for the *promotion free* fragment of the resource calculus with tests. To infer the analogous result for the full calculus, we need to extend the Taylor expansion of Section 2.3 to resource terms with tests:

$$\mathscr{T}(\bar{\tau}(Q)) = \{\bar{\tau}(Q') \mid Q' \in \mathscr{T}(Q)\},$$
$$\mathscr{T}(\tau[M_1, \dots, M_k]) = \{\tau[M_1', \dots, M_k'] \mid M_i' \in \mathscr{T}(M_i), \text{ for } 1 \leq i \leq k\}.$$

We then exploit the fact that $\mathcal{D}_\omega$ models this kind of Taylor expansion as well as a simulation lemma which relates the outer-reduction of a term with the outer-reduction of its Taylor expansion to prove that $[\![M]\!] = [\![N]\!]$ if and only if $M \equiv^\tau N$.

Theorem 2.9.5 (Bucciarelli et Al. [M1, M2])**.**
The model $\mathcal{D}_\omega$ *is fully abstract for the resource calculus with tests.*

Problem 10. *Is it possible to find a fully abstract model of the untyped resource calculus in categories of games?*

3

Nondeterminism in the Quantitative Setting

In which we show that not only the relational semantics is well suited for modeling both parallel composition and nondeterministic choice, but also that adding coefficients from semirings opens the way to capture several quantitative properties of programs.

- **A Relational Semantics for Parallelism and Non-Determinism in a Functional Setting.** A. Bucciarelli, T. Ehrhard and G. Manzonetto.
  Annals of Pure and Applied Logic, Vol. 163, issue 7, pp. 918-934, 2012.
- **Call-By-Value Non-Determinism in a Linear Logic Type Discipline.**
  A. Diaz-Caro, G. Manzonetto and M. Pagani.
  Proc. of Symposium on Logical Foundations of Computer Science 2013, Volume 7392 of Lecture Notes in Computer Science, pp. 364-376, 2013.
- **Constructing Differential Categories and Deconstructing Categories of Games.**
  J. Laird, G. Manzonetto and G. McCusker.
  Long version. Information and Computation, Volume 222, Issue C, pp. 247-264, 2013.
- **Constructing Differential Categories and Deconstructing Categories of Games.**
  J. Laird, G. Manzonetto and G. McCusker.
  International Conference on Automata, Languages and Programming - Part II (ICALP 2011), Volume 6756 of LNCS, pages 186-197, 2011.
- **Weighted Relational Models of Typed Lambda-Calculi.**
  J. Laird, G. Manzonetto, G. McCusker and M. Pagani.
  28th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2013), pp. 301-310, 2013.

When I introduced, together with Bucciarelli and Ehrhard, the relational model $\mathcal{D}_\omega$, we quickly realized that the semantic counterparts of *may* and *must* nondeterminism were at hand. During my ATER at the University Paris-Diderot (2008) we investigated the situation in detail and proved that $\mathcal{D}_\omega$ constitutes an adequate model for a λ-calculus endowed with parallel composition and nondeterministic choice. Subsequently, together with Pagani and Díaz-Caro, I showed that an analogous construction can be done in the call-by-name setting, and that an element in the interpretation of a term carries enough information to compute a bound for the number of reduction steps.

During my postdoc in Nijmegen (2011), I visited McCusker and Laird in Bath and we started a collaboration for defining and investigating differential categories based on games. Starting from a work of Melliès, Tabareau and Tasson we were able to provide an abstract canonical categorical construction for building differential categories, and reconstruct the relational semantics and categories of games as instances.

Subsequently (2013), in a larger collaboration including Pagani, we generalized the relational semantics by replacing relations from A and B with matrices indexed by A and B and populated by scalars from a continuous semiring $\mathcal{R}$. This gives a weighted relational semantics which is an adequate model of a nondeterministic extension of PCF, parametrized by $\mathcal{R}$. Specific instances of $\mathcal{R}$ allow us to compare programs not only with respect to "what they can do", but also "in how many steps" or "in how many different ways" (for nondeterministic PCF) or even "with what probability" (for probabilistic PCF).

3.1 NONDETERMINISM IN A FUNCTIONAL SETTING

It is well known that the λ-calculus is a deterministic functional programming language. At each reduction step, one could choose nondeterministically which β-redex to contract, but the confluence of the system assures that the outcome of a program, if any, is unique. Thus, λ-terms can be thought of as specifications of sequential, deterministic processes.

Several extensions of λ-calculus with parallel and nondeterministic constructs have been proposed in the literature. The purpose was either to increase the expressive power of the language, in the typed [Plo77, Pao06, Lai06] and untyped [Bou94, BL96] settings, or to study the interplay between higher-order and parallel or nondeterministic features of the language [Ong93, DdP96, DdP98]. When introducing nondeterminism in a functional setting, the same program can produce different results depending on its evaluation.

As briefly discussed in Section 2.2 (pag. 34), it becomes crucial to specify what notion of convergence is considered. Two widely used notions are:

- the *may* convergency: a nondeterministic choice converges if at least one of its components does. This characterizes the *angelic* nondeterminism;
- the *must* convergency: a nondeterministic choice converges whenever all its components converge. This characterizes the *demonic* nondeterminism.

Capturing nondeterministic behaviour in higher order languages presents a challenge for the usual denotational models based on domains. Consider two programs P_1 and P_2 converging respectively to values v_1 and v_2 having incompatible denotations. What semantic value should take the nondeterministic program $P_1 + P_2$ that *may* converges either to v_1 or to v_2? The resulting value should be some sort of "aggregation" of v_1 and v_2 that could be unavailable in the domain.

The typical techniques for interpreting "multi-valued" programs, like the ones described above, consist in using models based on Plotkin's *powerdomains* [Plo76]. Roughly speaking, a powerdomain $\mathcal{P}(\mathcal{D})$ is a domain whose elements are subsets of a domain $\mathcal{D}$. The denotation of a nondeterministic program P is a *set-valued function* in $\mathcal{D}$, that is a function mapping an input $d \in \mathcal{D}$ into a set $\{d_1, \ldots, d_k\} \in \mathcal{P}(\mathcal{D})$ containing all the possible values produced by P starting from the input d. Similar techniques are also used for interpreting the *must* nondeterminism. In this framework, both kinds of nondeterminism are modelled by some idempotent, commutative and associative operations.

In [FM09], Faure and Miquel defined the *aggregation monad* as a categorical counterpart of parallel execution. Powerdomains, sets with union and multisets with multiset union are all instances of aggregation monads (in categories of domains and of sets, respectively). In general, the notion of parallel composition is modelled by an aggregation monad that is neither idempotent, nor commutative, nor associative. There are however models of multiplicative exponential linear logic, where aggregation can be interpreted by the *mix rule*, if available. This rule allows to "put together" any two proofs whatsoever [DK00]. More precisely, parallel composition is obtained by combining the mix rule with the contraction rule. Indeed, mix can be seen as a linear morphism $X \otimes Y \multimap X \parr Y$, so that there is a morphism $?A \otimes ?A \multimap ?A$, obtained by composing the mix morphism $?A \otimes ?A \multimap ?A \parr ?A$ with the contraction morphism $?A \parr ?A \multimap ?A$. This composite morphism defines a commutative algebra structure on $?A$, which is used to model the "parallel composition" of MELL proofs. Therefore, in order to obtain a categorical model of parallel λ-calculus, it is sufficient to solve the equation $\mathcal{D} \cong \mathcal{D} \to \mathcal{D}$, with an object $\mathcal{D}$ of shape $?A$.

3.2 **MRel** AS QUANTITATIVE SEMANTICS OF NONDETERMINISM

In the relational semantics of λ-calculus, presented in Section 1.3, a program of type $A \to B$ is interpreted as a relation from $\mathcal{M}_{\mathrm{f}}(A)$ to B. The fact that the programs are interpreted as relations rather than functions is related with the intrinsic nondeterminism of the semantics while the multiplicities in the multiset keep track of how many times a resource is needed during the computation. For instance, the program $P := \lambda n.n * n : \mathbb{N} \to \mathbb{N}$ implementing the square product is interpreted in the relational semantics as follows:

$$[\![P]\!] = \{([n_1, n_2], k) \mid n_1 \cdot n_2 = k\}.$$

Indeed, P uses its input n twice and each call can give as answer a different number n_i because of the nondeterminism of the system. Since relations are closed under arbitrary unions, the *may* nondeterminism can be interpreted directly as set-theoretical union, without introducing any additional powerdomain construction or aggregation monad. In the particular case of our relational model $\mathcal{D}_\omega$, whose construction is recalled in Section 2.8, we are considering an object satisfying the recursive equation $\mathcal{D}_\omega = ?A$ where $A = (\mathcal{D}_\omega^{\mathbb{N}})^{\perp}$. Hence, $\mathcal{D}_\omega$ has the commutative algebra structure discussed in Section 3.1. It is precisely this structure that we use for interpreting parallel composition, just like Danos and Krivine did in [DK00] for an extension of $\lambda\mu$-calculus with a parallel composition operation.

More concretely, given two elements $\sigma, \tau \in \mathcal{D}_\omega$ we can always represent them as:

$$\sigma = a_1 \to \cdots \to a_n \to \varepsilon \qquad\qquad \tau = b_1 \to \cdots \to b_n \to \varepsilon$$

where ε is the special element of $\mathcal{D}_\omega$ satifying $\varepsilon = [] \to \varepsilon$. Therefore, it is possible to define a merging operator $\sigma \odot \tau = (a_1 \uplus b_1) \to \cdots \to (a_n \uplus b_n) \to \varepsilon$ using mix and contraction.

As a consequence, in the model $\mathcal{D}_\omega$, the semantic counterparts of *may* and *must* nondeterminism are at hand: they are the set-theoretic union and the mix-based operation $\odot$. In this framework, parallel composition is no longer idempotent: this is quite natural if we consider each component of a parallel composition as the specification of a process whose execution requires the consumption of some resources.

Non-determinism in call-by-name. In [M6], together with Bucciarelli and Ehrhard, we introduced the *$\lambda_{+\parallel}$-calculus*, an untyped λ-calculus whose terms are extended with *may* nondeterministic choice and *must* parallel composition:

$$\Lambda_{+\parallel} : \quad M, N, P, Q ::= x \mid \lambda x.M \mid MN \mid M + N \mid M \parallel N$$

The elements of $\Lambda_{+\parallel}$ are called *$\lambda_{+\parallel}$-terms*. This calculus is endowed with a pretty standard call-by-name operational semantics, defined in terms of head reduction $\to_{\mathtt{h}}$ in Figure 3.1.

The intuitive idea behind this notion of head-reduction is the following:

- when a $\lambda_{+\parallel}$-term of the form $M_1 + M_2$ gets in head position, either of the alternatives may be chosen to pursue the head reduction. If either M_1 or M_2 converges, then also the global program converges.
- when a $\lambda_{+\parallel}$-term of the form $M_1 \parallel M_2$ gets in head position, the head reduction continues the computations in parallel. In particular, if either M_1 or M_2 does not terminate, then also the global program diverges.

The *operational value* of a $\lambda_{+\parallel}$-term M is therefore the set of all head-normal forms that can possibly be reached starting from M. Accordingly, we say that a closed $\lambda_{+\parallel}$-term M *converges* whenever there is a reduction $M \to_{\mathtt{h}} N_1 \parallel \cdots \parallel N_k$ where each N_i is head normal.

β_h-reduction
$(\lambda x.M)N \to_{\mathtt{h}} M\{N/x\}$

$+$-reductions
$M + N \to_{\mathtt{h}} M$
$M + N \to_{\mathtt{h}} N$

$\|$-reductions
$(M \parallel N)P \to_{\mathtt{h}} MP \parallel NP$

Contextual rules

$$\frac{M \to_{\mathtt{h}} M'}{\lambda x.M \to_{\mathtt{h}} \lambda x.M'} \qquad \frac{N \to_{\mathtt{h}} N'}{N \parallel M \to_{\mathtt{h}} N' \parallel M} \qquad \frac{M \to_{\mathtt{h}} M'}{N \parallel M \to_{\mathtt{h}} N \parallel M'} \qquad \frac{M \to_{\mathtt{h}} M' \quad M \neq \lambda x.Q,\ Q_1 \parallel Q_2}{MN \to_{\mathtt{h}} M'N}$$

Figure 3.1: Operational semantics of the call-by-name $\lambda_{+\parallel}$-calculus (head-reduction).

The $\lambda_{+\parallel}$-calculus can be interpreted in the model $\mathcal{D}_\omega$ as expected.

Definition 3.2.1. *The interpretation $[\![M]\!]^{\mathcal{D}_\omega}_{\vec{x}} \subseteq \mathcal{M}_{\mathrm{f}}(\mathcal{D}_\omega)^{\vec{x}} \times \mathcal{D}_\omega$ of a $\lambda_{+\parallel}$-term M in $\mathcal{D}_\omega$ with respect to a sequence of variables $\vec{x} \supseteq \mathrm{FV}(M)$ extends the usual definition as follows:*

- $[\![M_1 + M_2]\!]^{\mathcal{D}_\omega}_{\vec{x}} = [\![M_1]\!]^{\mathcal{D}_\omega}_{\vec{x}} \cup [\![M_2]\!]^{\mathcal{D}_\omega}_{\vec{x}}$,
- $[\![M_1 \parallel M_2]\!]^{\mathcal{D}_\omega}_{\vec{x}} = \{(\vec{a} \uplus \vec{b}, \sigma \odot \tau) \mid (\vec{a}, \sigma) \in [\![M_1]\!]^{\mathcal{D}_\omega}_{\vec{x}},\ (\vec{b}, \tau) \in [\![M_2]\!]^{\mathcal{D}_\omega}_{\vec{x}}\ \}$,

where $\vec{a} \uplus \vec{b}$ is the componentwise union and $\odot$ denotes the mix-based operation defined previously.

By generalising Girard's reducibility candidates to the framework of $\lambda_{+\parallel}$-calculus, we were able to prove that the model $\mathcal{D}_\omega$ is *adequate*. This means that the $\lambda_{+\parallel}$-term M converges exactly when its interpretation in $\mathcal{D}_\omega$ is non-empty.

Theorem 3.2.2 (Bucciarelli et Al. [M6])**.**
For all $M \in \Lambda_{+\parallel}$, $[\![M]\!]^{\mathcal{D}_\omega} \neq \emptyset$ if and only if $M \twoheadrightarrow_{\mathtt{h}} N_1 \parallel \cdots \parallel N_k$ and each N_i is head-normal.

This model is not fully abstract for the observational equivalence which is associated with our notion of convergency. The counterexample is very simple, just consider the identity I and I $\parallel$ I, that are observationally indistinguishable, and verify that their interpretations $[\![\mathrm{I}]\!]^{\mathcal{D}_\omega} = \{[\sigma] \to \sigma \mid \sigma \in \mathcal{D}_\omega\}$ and $[\![\mathrm{I} \parallel \mathrm{I}]\!]^{\mathcal{D}_\omega} = \{[\sigma, \sigma] \to (\sigma \odot \sigma) \mid \sigma \in \mathcal{D}_\omega\}$ are different. The lack of full abstraction is related with the fact that parallel composition is idempotent from the operational point of view, whilst it is not idempotent from the denotational one.

Notice that, until now, we have used the logical structure of the model $\mathcal{D}_\omega$ but we did not really exploit its quantitative properties. In [dC09] de Carvalho has shown that Engeler's rgm $\mathcal{E}$ can be presented as an intersection type system, called *system R*, where the intersection is not idempotent. The lack of idempotency is the key ingredient to model the resource sensitiveness: while in the usual intersection type systems $\vdash M : \sigma \wedge \tau$ stands for "M can be used either as data of type σ or as data of type τ", when the intersection is not idempotent the meaning becomes "M will be called *once* as data of type σ and *once* as data of type τ". Hence, types should no longer be understood as *sets of terms*, but rather as *sets of calls* to terms. Moreover, de Carvalho proved that system R, beyond characterising converging terms, carries information on the evaluation sequence as well — the size of a derivation tree typing a term gives a bound on the number of steps needed to reach a head normal form. Similar results are obtained in [BL11] for a variant of system R characterising strong normalisation and giving a bound to the longest β-reduction sequence.

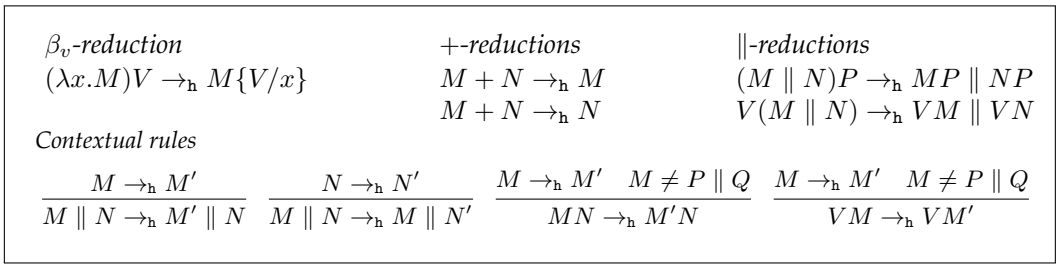


Figure 3.2: Operational semantics for the call-by-value $\lambda_{+\parallel}$-calculus (weak reduction).

Non-determinism in call-by-value. Together with Díaz-Caro and Pagani, we have shown in [M12] that the approach followed in call-by-name, also works in the call-by-value setting [RP04]. The terms of the call-by-value $\lambda_{+\parallel}$-calculus are the same as in call-by-name, but we also consider a subset $\mathrm{V}_{+\parallel} \subseteq \Lambda_{+\parallel}$ of *values* generated by:

$$\mathrm{V}_{+\parallel}: \quad V ::= x \mid \lambda x.M$$

This calculus is endowed with a call-by-value operational semantics, defined in terms of weak head reduction in Figure 3.2. The terminology "call-by-value" refers to the fact that a head-redex $(\lambda x.M)N$ can only be contracted when N is a value, while "weak" mans that we do not reduce under a λ-abstraction. In this context we say that a closed $\lambda_{+\parallel}$-term M *converges* whenever there is a reduction $M \twoheadrightarrow_{\mathtt{h}} V_1 \parallel \cdots \parallel V_n$ for some $V_i \in \mathrm{V}_{+\parallel}$.

In [Ehr12], Ehrhard introduced a non-idempotent intersection type system characterising the convergence of the deterministic call-by-value λ-calculus. This system is based on the so-called *second Girard's translation* of intuitionistic logic into linear logic $A \to B = !(A \multimap B)$ [Gir87, MOTW99]. In order to deal with the must nondeterminism, namely the parallel composition, we modify the translation by mapping the call-by-value λ-calculus into the polarised fragment of linear logic, as described by Laurent in [Lau02]. Then, our typing system is describing an object $\mathcal{V}$ in **Rel** satisfying the equation $\mathcal{V} \simeq\ !\mathcal{V} \multimap ?!\mathcal{V}$, where the connectives $?$ and $!$ are interpreted by the finite multiset comonad. Following the approach used in call-by-name, the composition of the mix rule and the contraction rule yields an operation $?!\mathcal{V} \otimes ?!\mathcal{V} \multimap ?!\mathcal{V}$ which is used to interpret the parallel composition.

To avoid a clumsy notation with multisets of multisets, we prefer to denote a !-multiset $[\rho_1, \ldots, \rho_m]$ (the type of a computation) with the linear logic multiplicative conjunction $\rho_1 \otimes \cdots \otimes \rho_m$ and a $?$-multiset $[\sigma_1, \ldots, \sigma_n]$ (the type of a parallel composition) with the multiplicative disjunction $\sigma_1 \parr \cdots \parr \sigma_n$. Such a notation stresses the fact that the non-idempotent intersection type systems issued from **Rel** are essentially contained in the multiplicative fragment of linear logic.

The set $\mathbb{T}^{\parallel}$ of *(parallel) types* and the set $\mathbb{C}$ of *computational types* are generated by the following grammar:

parallel-types	$\mathbb{T}^{\parallel}$*:*	$\sigma, \tau ::= \sigma \parr \tau \mid \rho$
computational-types	$\mathbb{C}$*:*	$\rho, \nu ::= 1 \mid \rho \otimes \nu \mid \rho \multimap \sigma$

$$\frac{}{x : \rho \vdash_{\mathcal{V}} x : \rho}\ (ax) \qquad \frac{\Gamma_i, x : \rho_i \vdash_{\mathcal{V}} M : \sigma_i \quad 1 \leq i \leq n \geq 0}{\otimes_{i=1}^{n} \Gamma_i \vdash_{\mathcal{V}} \lambda x.M : \otimes_{i=1}^{n}(\rho_i \multimap \sigma_i)}\ \multimap_I$$

$$\frac{\Gamma \vdash_{\mathcal{V}} M : \parr_{i=1}^{k}(\otimes_{j=1}^{n_i}(\rho_{ij} \multimap \sigma_{ij})) \quad \Gamma_i \vdash_{\mathcal{V}} N : \parr_{j=1}^{n_i}\rho_{ij} \quad 1 \leq i \leq k \quad k \geq 1 \quad n_i \geq 1}{\Gamma \otimes (\otimes_{i=1}^{k}\Gamma_i) \vdash_{\mathcal{V}} MN : \parr_{i=1}^{k}(\parr_{j=1}^{n_i}\sigma_{ij})}\ (\multimap_E)$$

$$\frac{\Gamma \vdash_{\mathcal{V}} M : \sigma}{\Gamma \vdash_{\mathcal{V}} M + N : \sigma}\ (+_\ell) \qquad \frac{\Gamma \vdash_{\mathcal{V}} N : \sigma}{\Gamma \vdash_{\mathcal{V}} M + N : \sigma}\ (+_r) \qquad \frac{\Gamma_1 \vdash_{\mathcal{V}} M : \sigma_1 \quad \Gamma_2 \vdash_{\mathcal{V}} N : \sigma_2}{\Gamma_1 \otimes \Gamma_2 \vdash_{\mathcal{V}} M \parallel N : \sigma_1 \parr \sigma_2}\ (\parallel_I)$$

Figure 3.3: Type system for the call-by-value $\lambda_{+\parallel}$-calculus.

For the sake of simplicity, types are considered up to associativity and commutativity of the tensor $\otimes$ and the par $\parr$. The type 1, which is the only atomic type, represents the empty tensor and is therefore its neutral element (i.e. $\rho \otimes 1 = \rho$). Accordingly, we write $\otimes_{i=1}^{n}\rho_i$ for $\rho_1 \otimes \cdots \otimes \rho_n$ when $n \geq 1$, and for 1 when $n = 0$. Similarly, when $n \geq 1$, $\parr_{i=1}^{n}\alpha_i$ stands for $\sigma_1 \parr \cdots \parr \sigma_n$. We do not allow the empty par as it would correspond to an empty sum of terms, that would be delicate to treat operationally [AD08]. An *environement* Γ is handled as a total map from Var to $\mathbb{C}$ such that $\Gamma(x) \neq 1$ for finitely many variables x. Given two environments Γ_1, Γ_2 their *tensor product* is defined by setting $(\Gamma_1 \otimes \Gamma_2)(x) = \Gamma_1(x) \otimes \Gamma_2(x)$.

The type system $\vdash_{\mathcal{V}}$ for the call-by-value $\lambda_{+\parallel}$-calculus is presented in Figure 3.3.

As mentioned above, we exploit the resource consciousness of our system for extracting from the typing tree of a term enough information to give a bound on the length of its reduction. In other words, we define the *measure* $|\pi|$ of a derivation tree π as follows:

$$\begin{array}{ll}
\pi = \dfrac{}{S}\ (ax) & |\pi| = 0, \\
\pi = \dfrac{\pi_1 \ \cdots \ \pi_n}{S}\ (\multimap_I) & |\pi| = \sum_{i=1}^{n} |\pi_i|, \\
\pi = \dfrac{\pi_0 \quad \pi_1 \cdots \pi_k \quad k \geq 1 \quad n_i \geq 1}{S}\ (\multimap_E) & |\pi| = \sum_{i=0}^{k} |\pi_i| + (\sum_{i=1}^{k} 2n_i) - 1, \\
\pi = \dfrac{\pi'}{S}\ (+^\ell) \quad \text{or} \quad \pi = \dfrac{\pi'}{S}\ (+^r) & |\pi| = |\pi'| + 1, \\
\pi = \dfrac{\pi_1 \quad \pi_2}{S}\ (\parallel_I) & |\pi| = |\pi_1| + |\pi_2|.
\end{array}$$

When checking that the subject reduction holds we verify not only that the type of a term is preserved along the head reduction, with the notable exception of the rules (+) that need to be treated more carefully, but also that the measure associated with the typing trees strictly decreases.

Theorem 3.2.3 (Subject reduction)**.** *Let π be the derivation tree of $\Gamma \vdash_{\mathcal{V}} M : \sigma$.*

- *If $M \to_{\mathtt{h}} N$ without using $+$-reductions, then there is a derivation π' of $\Gamma \vdash_{\mathcal{V}} N : \sigma$.*
- *If $M \to_{\mathtt{h}} N_1$ and $M \to_{\mathtt{h}} N_2$ using $+$-reductions, then there is a derivation π' of either $\Gamma \vdash_{\mathcal{V}} N_1 : \sigma$ or $\Gamma \vdash_{\mathcal{V}} N_2 : \sigma$.*

In both cases the measure has decreased: $|\pi'| = |\pi| - 1$.

This gives a combinatorial proof of weak normalization for the call-by-value $\lambda_{+\|}$-calculus. Moreover, when the type derivation is of the form $\vdash_{\mathcal{V}} M : 1 \parr \cdots \parr 1$, it is possible to prove that the associated measure provides *the exact length* of the reduction of M.

Theorem 3.2.4 (Díaz-Caro et Al. [M12])**.**
Given a closed $\lambda_{+\|}$-term M and a number $k > 0$, there is a typing tree π for $\vdash_{\mathcal{V}} M : \parr^k 1$ if and only if there are values $V_1, \ldots, V_k$ and a head reduction sequence $M \twoheadrightarrow_{\mathtt{h}} V_1 \parallel \cdots \parallel V_k$ of length $|\pi|$.

As mentioned earlier and discussed thoughtfully in [PPR15], the choice of presenting a model through a type discipline or a categorical object satisfying the equation $\mathcal{V} \simeq !\mathcal{V} \multimap ?!\mathcal{V}$ is more a matter of taste rather than a technical decision. For instance, the interpretation of a $\lambda_{+\|}$-term M which is induced by our type system $\vdash_{\mathcal{V}}$ can be defined as:

$$[\![M]\!]^{\mathcal{V}} = \{(\Gamma, \sigma) \mid \Gamma \vdash_{\mathcal{V}} M : \sigma\}.$$

Accordingly, the interpretation of a closed term is identified with the set of its types. The adequacy of the model $\mathcal{V}$ follows easily from the subject reduction (Theorem 3.2.4) and the monotonicity of the interpretation.

Corollary 3.2.5 (Adequacy)**.**
For all closed $M, N \in \Lambda_{+\|}$, if $[\![M]\!]^{\mathcal{V}} \subseteq [\![N]\!]^{\mathcal{V}}$ then for all closed $\vec{P} \in \Lambda_{+\|}$, we have that $M\vec{P}$ converges implies that $N\vec{P}$ converges.

Also in this case the identity I and its parallel composition I $\parallel$ I are operationally indistinguishable but have different interpretations in $\mathcal{V}$. However, the model $\mathcal{V}$ is not fully abstract even in the deterministic case. Indeed, the call-by-value λ-calculus admits the creation of an *ogre* that is capable of 'eating' any finite sequence of values and still converge, constituting therefore a top of the call-by-value observational preorder, denoted here by $\sqsubseteq^{\mathsf{cbv}}$. Following Boudol [Bou94], we define the ogre as follows:

$$\mathrm{O} := (\lambda xy.xx)(\lambda xy.xx).$$

On the one hand we have that the ogre O converges since $\mathrm{O} \to \lambda y.\mathrm{O}$ and remains convergent when applied to every sequence of values, by discarding them one at a time.

Lemma 3.2.6. *For all closed $\lambda_{+\|}$-terms M we have $M \sqsubseteq^{\mathsf{cbv}} \mathrm{O}$.*

On the other hand, we have that $\sigma \in [\![\mathrm{O}]\!]^{\mathcal{V}}$ if and only if $\sigma = \bigotimes_{i=0}^{n}(\mathbf{1} \multimap \sigma_i)$ with $n \geq 0$ and $\sigma_i \in [\![\mathrm{O}]\!]^{\mathcal{V}}$ for all $i \leq n$. In particular, it follows that $(\mathbf{1} \multimap \mathbf{1}) \multimap (\mathbf{1} \multimap \mathbf{1}) \in [\![\mathrm{I}]\!]^{\mathcal{V}} - [\![\mathrm{O}]\!]^{\mathcal{V}}$, hence $[\![\mathrm{I}]\!]^{\mathcal{V}} \not\subseteq [\![\mathrm{O}]\!]^{\mathcal{V}}$. Summing up, we have $\mathrm{I} \sqsubseteq^{\mathsf{cbv}} \mathrm{O}$, but $[\![\mathrm{I}]\!]^{\mathcal{V}} \not\subseteq [\![\mathrm{O}]\!]^{\mathcal{V}}$, therefore the model $\mathcal{V}$ is not inequationally full abstraction for the call-by-value λ-calculus. To conclude that also the equational fully abstraction fails, one needs to find a more subtle counterexample.

Problem 11. *Show that the model $\mathcal{V}$ introduced in [M12] (equivalently, Ehrhard's models of [Ehr12]) is not equationally fully abstract for the call-by-value λ-calculus.*

Hint: Following [EHR92], define the λ-terms:

$$M_1 := \lambda x.(\lambda xyz.\Omega)(x(\lambda x.\Omega)(\lambda xy.\Omega))(x(\lambda xy.\Omega)(\lambda x.\Omega))$$
$$M_2 := \lambda x.(\lambda xy.\Omega)(x(\lambda x.\Omega)(\lambda x.\Omega))$$

Show that $M_1 \equiv^{\mathsf{cbv}} M_2$, but $[\![M_1]\!]^{\mathcal{V}} \neq [\![M_2]\!]^{\mathcal{V}}$. The intuition is that the observational equivalence satisfies a property of stability which is not satisfied by the interpretation in $\mathcal{V}$.

3.3 Constructing Differential Categories

Differential categories are particularly interesting models of nondeterministic languages, as they keep track of the resource usage. However, it can be difficult to define such categories by hand, because they possess a quite rich logical structure. Together with Laird and McCusker [M13, M14], we found a general construction of models of intuitionistic linear logic that are also differential categories. The main ingredient is the construction of a category which possesses a comonad delivering cofree cocommutative comonoids.

The construction. Let $\mathbf{C}$ be a symmetric monoidal category enriched over sup-lattices, that is, over *idempotent* commutative monoids with *all* sums. Any product $A \times B$ in $\mathbf{C}$ is necessarily a *biproduct*, that is, it is also a coproduct and the canonical map $[\langle \mathrm{Id}_A, 0\rangle, \langle 0, \mathrm{Id}_B\rangle] : A \oplus B \to A \times B$ is an isomorphism. Similarly, every coproduct is a biproduct. Suppose that $\mathbf{C}$ has all *countable* biproducts, and that the monoidal structure distributes over them. We construct a differential structure on the Karoubi envelope $\mathcal{K}(\mathbf{C})$ of $\mathbf{C}$, which has:

- as objects the pairs (A, f) where A is an object of $\mathbf{C}$ and $f : A \to A$ is an idempotent,
- as maps $(A, f) \to (B, g)$, those maps $h : A \to B$ from $\mathbf{C}$ such that $h = f; h; g$.

$\mathcal{K}(\mathbf{C})$ inherits the monoidal structure, sup-lattice enrichment and biproducts from $\mathbf{C}$.

First, for any object A of $\mathbf{C}$, write $A^{\otimes n}$ to denote the n-fold tensor power of A. The *symmetric* tensor power A^n, if it exists, is the equaliser of the diagram

$$(A^{\otimes n}, f^{\otimes n}) \;\overrightarrow{\vdots\ n!\text{ permutations}}\; (A^{\otimes n}, f^{\otimes n})$$

consisting of all $n!$ permutations from $A^{\otimes n}$ to itself.

In $\mathcal{K}(\mathbf{C})$ we can readily construct symmetric tensor powers, as follows. Given an object A of $\mathbf{C}$, define $\Theta_{A,n} : A^{\otimes n} \to A^{\otimes n}$ to be the sum of the $n!$ permutation maps. Straightforward calculation establishes the following.

Lemma 3.3.1. *For any object (A, f) of $\mathcal{K}(\mathbf{C})$, the following diagram is an equalizer.*

$$(A^{\otimes n}, f^{\otimes n}; \Theta_{A,n}) \xrightarrow{f^{\otimes n}; \Theta_{A,n}} (A^{\otimes n}, f^{\otimes n}) \;\overrightarrow{\vdots\ n!\ \textit{permutations}}\; (A^{\otimes n}, f^{\otimes n})$$

Moreover, these equalizer diagrams are preserved by tensor products.

One consequence of this is that there are maps $(A, f)^{m+n} \to (A, f)^m \otimes (A, f)^n$ whose underlying maps are given by $f^{\otimes m+n}; \Theta_{A,m+n}$, as one might expect.

These symmetric tensor powers allow to construct a coalgebra modality on $\mathcal{K}(\mathbf{C})$ as the free commutative comonoid. Recall that a *commutative comonoid* in a symmetric monoidal category is an object A together with maps $c : A \to A \otimes A$ and $w : A \to 1$ satisfying the obvious commutativity, associativity and unit diagrams; morphisms of comonoids are morphisms between the underlying objects that preserve the comonoid structure. Let $\mathcal{K}^{\otimes}(\mathbf{C})$ be the category of commutative comonoids and comonoid morphisms in $\mathcal{K}(\mathbf{C})$.

Lemma 3.3.2. *The forgetful functor $U : \mathcal{K}^{\otimes}(\mathbf{C}) \to \mathcal{K}(\mathbf{C})$ has a right adjoint, whose action on objects takes (A, f) to the biproduct $\bigoplus_{n\in\mathbb{N}}(A, f)^n$, which we call $!(A, f)$.*

Composing these two adjoint functors yields a comonad $(!, \mathrm{der}, \mathrm{dig})$ on $\mathcal{K}(\mathbf{C})$.

Lemma 3.3.3. *The comonad $(!, \mathrm{der}, \mathrm{dig})$ is a coalgebra modality. In fact, it is a linear exponential comonad (also known as a storage modality).*

The fact that the cofree commutative comonoid provides a linear exponential comonad is attributed to Lafont. The construction of this comonad along the lines given above follows the recipe in [MTT09], though the use of Karoubi envelope to generate a category possessing the required equalizers is our original contribution.

We are now in a position to construct a differential operator on $\mathcal{K}(\mathbf{C})$, making it into a differential category. The differential operator is given by precomposition with the *deriving transformation* $d : (A, f) \otimes !(A, f) \to !(A, f)$ defined as follows.

For each n, the map $f^{\otimes n+1}; \Theta_{A,n+1}$ in $\mathbf{C}$ gives us a morphism

$$f^{\otimes n+1}; \Theta_{A,n+1} : (A, f) \otimes (A, f)^n \to (A, f)^{n+1}$$

and hence we obtain maps $\cong; \pi_n; f^{\otimes n+1}; \Theta_{A,n+1} : (A, f) \otimes !(A, f) \to (A, f)^{n+1}$ where $\cong$ is the distributivity map. Tupling all these gives us a morphism $(A, f) \otimes !(A, f) \to \bigoplus_n (A, f)^{n+1}$, and finally pairing this with $0 : (A, f) \otimes !(A, f) \to I$ gives the required map.

Theorem 3.3.4 (Laird et Al. [M13])**.**
Let $\mathbf{C}$ be a sup-lattice-enriched symmetric monoidal category with countable distributive products. Then $\mathcal{K}(\mathbf{C})$ is a sup-lattice-enriched differential category, and the Kleisli category $\mathcal{K}_!(\mathbf{C})$ a cpo-enriched Cartesian differential category. If $\mathbf{C}$ is monoidal closed (in the sup-lattice-enriched sense) then $\mathcal{K}_!(\mathbf{C})$ is a cpo-enriched Cartesian-closed differential category.

In the presentation given in [M14], we asked for n-ary sum operators $\Sigma_n[f_1, \ldots, f_n]$ needed to produce the symmetric tensors and *a priori* separated from the sums provided by commutative monoid enrichment. This is because there are models, considered in Section 3.7, in which these sums are indeed different (but related); in particular, the commutative monoid structure does not need to be idempotent while the n-ary sums must be.

Preliminary steps. To apply the construction above, we need a sup-lattice enriched symmetric monoidal category with countable distributive biproducts. Such categories can readily be manufactured via a series of free constructions.

1) Sup-lattice enrichment. Beginning with a symmetric monoidal category $\mathbf{C}$, one can construct its *sup-lattice-completion* $\mathbf{C}^+$ as the category with the same objects, but whose maps $A \to B$ are sets of maps in the original category. This is a sup-lattice enriched category, with joins of maps given by unions, and monoidal structure inherited from the original category. The monoidal closed structure is also inherited, if it exists.

2) Biproduct completion. Given a sup-lattice-enriched symmetric monoidal category $\mathbf{C}$, its *biproduct completion* $\mathbf{C}^\oplus$ has as objects indexed sets $\{A_i \mid i \in I\}$ of objects in the original category, and as morphisms $\{A_i \mid i \in I\} \to \{B_j \mid j \in J\}$ *matrices* of morphisms, i.e., for all indices i and j, a morphism $A_i \to B_j$. Composition is (potentially infinite) matrix multiplication — the infinite sums required for composition are the reason we require sup-lattice enrichment. The biproduct of a set of objects is given by the disjoint union of families.

We will be interested in some categories which arise by performing these two constructions in sequence. Given a category $\mathbf{C}$, we denote by $\mathbf{FamRel}(\mathbf{C})$ the category whose objects are families $\{A_i \mid i \in I\}$ of objects of $\mathbf{C}$, and whose morphisms $\{A_i \mid i \in I\} \to \{B_j \mid j \in J\}$ are given by sets of triples (i, j, f) where $i \in I$, $j \in J$ and $f : A_i \to B_j$ in $\mathbf{C}$. Note that for a given i and j there might be no such triples in a morphism, or one, or many. It is easy to check that $\mathbf{FamRel}(\mathbf{C})$ is isomorphic to the category $(\mathbf{C}^+)^\oplus$.

A simple but central example begins with the terminal category $\mathbf{1}$. $\mathbf{FamRel}(\mathbf{1})$ is the category $\mathbf{Rel}$ of sets and relations. On the image of $\mathbf{Rel}$ in $\mathcal{K}(\mathbf{Rel})$, ! is the finite-multiset comonad, so we find $\mathbf{MRel}$ embedded in $\mathcal{K}_!(\mathbf{Rel})$ as a sub-Cartesian-differential-category.

3.4 A Differential Category of Games

We present a category of games $\mathbf{G}^{\otimes}$ introduced by Harmer and McCusker at the end of the '90s as a fully abstract model of *Erratic Idealized Algol* (EIA) [Har99, HM99]. EIA is a typed λ-calculus with constants, including an erratic choice operator, making it a higher-order imperative programming language with variables in which natural numbers can be stored. We show that $\mathbf{G}^{\otimes}$ is a actually a Cartesian closed differential category, which demonstrates that this kind of categories arise naturally when modelling the nondeterminism.

Arenas, moves and sequences. An *arena* A is a finite bipartite forest over two sets of moves, M_A^P and M_A^O with edge relation $\vdash$. We say that a move is *enabled* by its parent in the forest, and that root moves are *initial*. A *QA-arena* is an arena equipped with a function labelling each move as a question (Q) or answer (A), such that every answer is the child of a question. We assume the standard notions of *justified sequence*, *views*, *P-* and *O-visibility* from the game semantics literature (see, e.g., [McC98]). Given a justified sequence s, we say that an answer-move occurrence a *answers* the question occurrence q that justifies it. A justified sequence s satisfies *P-well-bracketing* if, for every prefix $s'a$ with a an answer move by P, the question that a answers is the rightmost O-question in the player view $\ulcorner s' \urcorner$; we call this the *pending question* at s'. A justified sequence is *complete* if every question is answered exactly once; we write $\mathsf{comp}(A)$ for the set of complete justified sequences of A.

Lemma 3.4.1. *If s is a complete justified sequence that satisfies P-visibility (resp. O-visibility), then s satisfies P-well-bracketing (resp. O-well-bracketing).*

The monoidal structure. A sequence is called *well-opened* if it contains exactly one initial O-move. A *strategy* for an arena A is a set of complete sequences in which O plays first, satisfying P-visibility (and, by Lemma 3.4.1, P-well-bracketing). Given a strategy σ, $\mathsf{wv}(\sigma)$ is the set of sequences in σ that are well-opened and satisfy O-visibility. Given arenas A and B, we write $A \uplus B$ for the arena arising as the disjoint union of A and B, and $A^{\perp}$ for the arena A with O and P-moves interchanged. We can define a category $\mathbf{G}$ in which objects are arenas whose roots are all O-moves, and morphisms $\sigma : A \to B$ are strategies on the arena $A^{\perp} \uplus B$. Composition of strategies is the usual "parallel composition plus hiding" construction, and identities are copycat strategies. As proved in [Har99, HM99], this category is monoidal closed: disjoint union of arenas gives a tensor product, and exponentials are given by the arena $A \multimap B$, which consists of the arena B with a copy of $A^{\perp}$ attached below each initial move; duplication of $A^{\perp}$ is required to maintain the forest structure. Every object of $\mathbf{G}$ possesses a canonical comonoid structure, and the subcategory of comonoid homomorphisms is a Cartesian closed category $\mathbf{G}^{\otimes}$. These maps are those whose choice of move at any stage depends only on the current thread, that is, the subsequence of moves hereditarily justified by the initial O-move currently in view. It follows that such strategies are completely determined by the well-opened plays they contain.

The differential structure of $\mathbf{G}^{\otimes}$. Let s be a complete, well-opened play in $A^{\perp} \uplus B$ which contains at least one initial A-move. We say that a complete play s' in $A^{\perp} \uplus A^{\perp} \uplus B$ is a *derivative* of s if $\mathrm{contr}; \{s'\} = \{s\}$ and s' contains one initial move in the left occurrence of $A^{\perp}$. We then define $\mathsf{D}(\sigma)$ as the strategy whose well-opened plays are

$$\{s' \in \mathsf{comp}(A^{\perp} \uplus A^{\perp} \uplus B) \mid s' \text{ is a derivative of some well-opened } s \in \sigma\}.$$

It is possible to verify directly that this makes $\mathbf{G}^{\otimes}$ a Cartesian closed differential category, but we now show that this follows from the general constructions of Section 3.3.

3.5 RECONSTRUCTING CATEGORIES OF GAMES

We apply some of the constructions developed above to reconstruct the category $\mathbf{G}^{\otimes}$ of Section 3.4 and discover its differential structure as an instance.

Exhausting games. Given a finite arena A, a *path* is a non-repeating enumeration of all its moves, respecting the order given by the edge relation in the arena — that is, a traversal of the forest — such that the first move is by O and moves alternate polarity thereafter. Note that every move in a path has a unique justifier earlier in the path. An *exhausting strategy* on the arena A is a set of even-length paths that satisfy P-visibility. In particular, if A has an odd number of moves, the only exhausting strategy is the empty set.

Definition 3.5.1. *The category* $\mathbf{EG}$ *of exhausting games has:*

- *finite O-rooted arenas as objects,*
- *exhausting strategies on* $A^{\perp} \uplus B$ *as maps from* A *to* B*, with the usual composition.*

The monoidal structure is given by the disjoint union of arenas. It is clear that $\mathbf{EG}$ is sup-lattice enriched: unions of strategies are strategies, and composition preserves unions. We may therefore form its biproduct completion, to obtain the structure we require to construct a differential category as in Section 3.3. We write $\mathcal{K}(\mathbf{EG}^{\oplus})$ for the differential category so constructed, and $\mathcal{K}_!(\mathbf{EG}^{\oplus})$ for its Kleisli category, which is a Cartesian differential category. Thought $\mathbf{EG}$ is not monoidal closed, it has all R-exponentials, that is exponentials of the form $A \multimap R$ where R is the arena with a single move belonging to O.

The full subcategory of R-exponentials of $\mathcal{K}_!(\mathbf{EG}^{\oplus})$ is therefore a Cartesian closed differential category that contains the original category $\mathbf{G}^{\otimes}$, as we have shown in [M13].

Proposition 3.5.2. *There is a full and faithful product-preserving functor from* $\mathbf{G}^{\otimes}$ *to* $\mathcal{K}_!(\mathbf{EG}^{\oplus})$.

Some refined categories of games. We refine the category $\mathbf{G}^{\otimes}$ by considering strategies satisfying a notion of causal independence similar to that of Melliès [Mel06]. Let us define a relation $\sim$ on the paths of an arena A as the smallest equivalence such that:

$$s \cdot o \cdot p \cdot o' \cdot p' \cdot t \sim s \cdot o' \cdot p' \cdot o \cdot p \cdot t$$

where o, o' are O-moves and p, p' are P-moves. We say that a path is *safe* if, whenever $s = s' \cdot o \cdot p \cdot o' \cdot p' \cdot t$ and o justifies p', p' justifies o'.

A $\sim$*-strategy* σ on an arena A is a set of safe paths that is $\sim$-closed, that is, if $s \in \sigma$ and $s \sim t$ then $t \in \sigma$. A $\sim$-strategy σ is *deterministic* if it is non-empty, and the longest common prefix of any $s, t \in \sigma$ has even length.

We define the category $\mathbf{G}_{\sim}$ as the subcategory of $\mathbf{G}$ consisting of $\sim$-closed strategies. Again taking the subcategory of comonoid homomorphisms, we arrive at a Cartesian closed differential category $\mathbf{G}^{\otimes}_{\sim}$. Also in this case the category $\mathbf{G}^{\otimes}_{\sim}$ can be reconstructed starting from the category $\mathbf{EG}_{\sim}$ having O-rooted arenas as objects and deterministic $\sim$-strategies on $A^{\perp} \uplus B$ as maps $A \to B$. Indeed, our construction gives us a comonad ! on $\mathcal{K}(\mathbf{FamRel}(\mathbf{EG}_{\sim}))$ such that the Kleisli category $\mathcal{K}_!(\mathbf{FamRel}(\mathbf{EG}_{\sim}))$ is a Cartesian differential category. Also in this case $\mathbf{EG}_{\sim}$ is not monoidal closed, but it has all R-exponentials, so the full subcategory of $\mathcal{K}_!(\mathbf{FamRel}(\mathbf{EG}_{\sim}))$ comprising the arenas with a single root is a Cartesian closed differential category and there is a full and faithful functor from $\mathbf{G}^{\otimes}_{\sim}$ into $\mathcal{K}_!(\mathbf{FamRel}(\mathbf{EG}_{\sim}))$ which preserves the Cartesian closed differential structure.

3.6 A Fully Abstract Model of Resource PCF

The categories $\mathbf{G}^{\otimes}$, $\mathbf{G}^{\otimes}_{\sim}$ and $\mathbf{MRel}$ constitute sound models of *Resource* PCF [M13], a simply typed resource calculus which incorporates the constants of PCF [Plo77], making it a prototypical resource-sensitive programming language. Its operational semantics is given in [M13] in terms of a nondeterministic linear-head reduction. Our constructions show that each of $\mathcal{K}_!(\mathbf{FamRel}(\mathbf{EG}_\sim))$, $\mathcal{K}_!(\mathbf{BP}(\mathbf{EG}))$ and $\mathcal{K}_!(\mathbf{FamRel}(\mathbf{1}))$ is a cpo-enriched differential Cartesian category with enough exponentials to interpret Resource PCF; and indeed we have identified full subcategories which are cpo-enriched Cartesian closed differential categories containing all the objects needed to interpret Resource PCF.

For $\mathbf{G}^{\otimes}_{\sim}$ and $\mathbf{MRel}$, there is more to be said. Consider those arenas for which there exists a Q/A-labelling such that every question enables a unique answer — this is a constraint on the shapes of the trees, rather than additional structure. Write $\mathbf{EG}^{QA}_{\sim}$ for the full subcategory of $\mathbf{EG}_\sim$ consisting of such arenas, and note that $\mathbf{G}^{\otimes}_{\sim}$ embeds in $\mathcal{K}_!(\mathbf{FamRel}(\mathbf{EG}^{QA}_{\sim}))$ by construction: indeed, for every such arena, the set of safe paths is non-empty.

The unique functor $\top : \mathbf{EG}^{QA}_{\sim} \to \mathbf{1}$ is full and extends through our constructions to a full functor from $\mathcal{K}_!(\mathbf{FamRel}(\mathbf{EG}^{QA}_{\sim}))$ to $\mathcal{K}_!(\mathbf{FamRel}(\mathbf{1}))$. Moreover, the only idempotents we make use of in the Karoubi envelope have the form $\sum_{f\in G} f$ where G is some group of automorphisms. In the case of $\mathbf{Rel}$, these idempotents are equivalence relations, and an object $(A, \simeq)$ in $\mathcal{K}(\mathbf{Rel})$ is isomorphic to $(A/\simeq, =)$. The part of the Karoubi envelope that is used in our constructions is therefore equivalent to $\mathbf{Rel}$ itself, with the comonad being the usual finite-multiset comonad, and the Kleisli category being $\mathbf{MRel}$. We obtain a full functor from $\mathbf{G}^{\otimes}_{\sim}$ to $\mathbf{MRel}$ which preserves all the relevant structure. This is essentially the "time-forgetting" map of [BDER98], which is functorial because of the $\sim$-closure.

Theorem 3.6.1. *The models of Resource* PCF *in* $\mathbf{G}^{\otimes}_{\sim}$ *and* $\mathbf{MRel}$ *have the finite definability property: every finite element of the model is the denotation of some term of Resource* PCF.

From this result it follows that $\mathbf{MRel}$ is fully abstract for Resource PCF. Note however that the notion of full abstraction changes substantially for PCF-like languages because it is customary to observe the behaviour of programs at *ground type*. Indeed, we say that a closed term M of type int *converges*, written $M \Downarrow$, if M reduces to some numeral k.

Theorem 3.6.2 (Laid et Al. [M13, M14])**.**
The model of Resource PCF *in* $\mathbf{MRel}$ *is fully abstract, that is, for any terms* M *and* N, $[\![M]\!] \subseteq [\![N]\!]$ *if and only if whenever* $C(\!|M|\!) \Downarrow$ *for some context* C*, we also have* $C(\!|N|\!) \Downarrow$.

Proof sketch. ($\Rightarrow$) By monotonicity of interpretation $[\![M]\!] \subseteq [\![N]\!]$ and $C(\!|M|\!) \Downarrow$ entail $C(\!|N|\!) \Downarrow$. ($\Leftarrow$) Let M, N be closed terms of type A such that $[\![M]\!] \not\subseteq [\![N]\!]$. There must be some $a \in [\![M]\!] - [\![N]\!]$. By finite definability, the relation $\{(a, 0)\} : A \to \mathsf{int}$ is the denotation of some term $x : A \vdash C(\!|x|\!) : \mathsf{int}$. Thus $[\![C(\!|M|\!)]\!] = [\![\mathsf{0}]\!]$ while $[\![C(\!|N|\!)]\!] = \emptyset$, so $C(\!|M|\!) \Downarrow$ but $C(\!|N|\!) \not\Downarrow$. $\square$

The above argument does not transfer to $\mathbf{G}^{\otimes}_{\sim}$ because the game-semantic equivalent of $\{(a, 0)\}$ would not in general be $\sim$-closed. Indeed $\mathbf{G}^{\otimes}_{\sim}$ is not fully abstract, for similar reasons to the failure of full abstraction of the innocent strategy model of PCF [HO00]. For instance, the model contains strategies for both left-to-right and right-to-left evaluation of the addition function, but no σ-closed strategy (and no context) can distinguish them.

Problem 12. *Provide a direct proof of the fact that every finite element of the relational semantics is the denotation of some term of Resource* PCF*, and conclude that it is fully abstract.*

3.7 THE WEIGHTED RELATIONAL SEMANTICS

We have shown that the relational semantics is quantitative in the sense that the elements living in the interpretation of a term M carry as information the amount of resources used by M during its execution. However, we have also seen that having an idempotent sum entails an unrecoverable loss of information, namely the number of different ways a nondeterministic program can reduce to a certain result. In collaboration with Laird, McCusker and Pagani, we were able to provide in [M15] a uniform denotational account of a range of quantitative notions, using a simple refinement of the relational model.

The underlying idea is that a relation between two sets A and B can be seen as matrix indexed by A and B, and populated by Boolean values. Replacing the Booleans by elements of an arbitrary continuous semiring $\mathcal{R}$, we arrive at a new *weighted relational semantics* embodying some quantitative information. The weighted relational interpretation of a program M of type $A \to B$ is an infinite matrix

$$[\![M]\!] \in \mathcal{R}^{\mathcal{M}_{\mathrm{f}}(A) \times B}$$

whose lines are indexed by finite multisets of inputs, whose columns are indexed by the corresponding outputs and whose entries are the coefficients from $\mathcal{R}$. For instance, a program M of type $\mathsf{bool} \to \mathsf{bool}$ is interpreted by a matrix of the form:

$$[\![M]\!] = \begin{array}{r} \\ \\ [] \mapsto \\ [\mathsf{t}] \mapsto \\ [\mathsf{f}] \mapsto \\ [\mathsf{t},\mathsf{f}] \mapsto \\ \vdots \end{array} \begin{array}{c} \mathsf{t} \quad \mathsf{f} \\ \downarrow \quad \downarrow \\ \left[\begin{array}{cc} p_1 & q_1 \\ p_2 & q_2 \\ p_3 & q_3 \\ p_4 & q_4 \\ \vdots & \vdots \end{array}\right. \end{array}$$

Like in the relational semantics the finite multisets represent the different calls needed to produce the corresponding output, but what is the meaning of the associated scalars?

We consider $\mathsf{PCF}^{\mathsf{or}}$, the extension of Plotkin's PCF with a nondeterministic choice operator which can naturally be interpreted in our models by addition of matrices. The interpretation of a closed term of ground type is then a vector of scalars from $\mathcal{R}$. To understand their meaning, we consider a further extended language $\mathsf{PCF}^{\mathcal{R}}$, in which terms can be instrumented with elements of $\mathcal{R}$. We demonstrate that our weighted relations correctly model execution in this language, and go on to use $\mathsf{PCF}^{\mathcal{R}}$ as a metalanguage for quantitative modelling of the execution of programs in $\mathsf{PCF}^{\mathsf{or}}$. By varying our choice of $\mathcal{R}$, and of how terms are instrumented, we show in Section 3.11 that our models can capture, for instance, may- and must- convergence for nondeterministic programs, the probability of convergence; and the minimum and maximum number of reduction steps to convergence.

The models we describe in this section are in some sense the simple cousins of a range of models studied by Ehrhard and co-authors: finiteness spaces, Köthe spaces, as well as probabilistic coherence spaces [Ehr05, Ehr02, DE11]. In all cases, the coherence structure serves to constrain morphisms so that the quantities in the model can remain finite. Our models sacrifice this property in return for simplicity and generality.

3.8 THE CATEGORY $\mathcal{R}^{\oplus}$ AND ITS KLEISLI $\mathcal{R}^{\oplus}_{!}$

We show that constructions similar to the ones in Section 3.3 can be applied to build a cartesian closed differential category starting from any commutative continuous semiring.

Continuous semirings have been introduced in [DK09] and are instances of continuous algebras (see e.g. [Gue81]). Intuitively, continuous semirings are semirings having a structure of *complete partial order* (*cpo*). Recall that a cpo is a partially ordered set $(X, \preceq)$ having a bottom element and such that any directed subset $D \subseteq X$ has a supremum $\bigvee D$.

Definition 3.8.1 (Continuous semiring)**.** *A* continuous semiring $\mathcal{R}$ *is a semiring*

$$(|\mathcal{R}|, +, \cdot, \mathbf{0}, \mathbf{1})$$

equipped with a partial order $\preceq$ such that:

- $(|\mathcal{R}|, \preceq)$ *is a cpo having* $\mathbf{0}$ *as bottom element,*
- *the operators $+$ and $\cdot$ are continuous.*

We will often confuse $\mathcal{R}$ with its underlying set $|\mathcal{R}|$. Since the addition is continuous, in any continuous semiring $\mathcal{R}$ we can define infinite sums by directed suprema:

$$\sum_{p \in I} p := \bigvee_{F \subseteq_{\mathrm{f}} I} \left(\sum_{p \in F} p \right).$$

In particular, every continuous semiring $\mathcal{R}$ has a top element $\boldsymbol{\infty} := \sum_{p \in \mathcal{R}} p$ satisfying $p + \boldsymbol{\infty} = \boldsymbol{\infty}$ for every $p \in \mathcal{R}$. Before exhibiting the most famous examples of continuous semiring, we fix some notations. Given a set X, we write $\overline{X}$ for $X \cup \{\infty\}$ and $X_{\perp}$ for $X \cup \{-\infty\}$, where ∞, $-\infty$ are fresh elements.

Example 3.8.2. *The following semirings, endowed with the natural ordering, are continuous.*

1. Boolean semiring: $\mathcal{B} := (\{\mathsf{t}, \mathsf{f}\}, \vee, \wedge, \mathsf{f}, \mathsf{t})$ *where* $\mathsf{f} < \mathsf{t}$.
2. $\mathbb{N}$ completed: $\mathcal{N} := (\overline{\mathbb{N}}, +, \cdot, 0, 1, \leq)$ *where $+$, $\cdot$ are defined in the obvious way (in particular $0 \cdot \infty = 0 = \infty \cdot 0$). Note that for every infinite $S \subseteq \overline{\mathbb{N}}$ we have $\bigvee S = \infty$.*
3. Tropical semiring: $\mathcal{T} := (\overline{\mathbb{N}}, \min, +, \infty, 0, \geq)$. *Note that the order is reversed so that 0 is the top element.*
4. Arctic semiring: $\mathcal{A} := (\overline{\mathbb{N}}_{\perp}, \max, +, -\infty, 0, \leq)$ *where $\max, +$ are extended as usual (e.g. $(-\infty) + \infty = -\infty$).*
5. $\mathbb{R}^{+}$ completed: $\mathcal{P} := (\overline{\mathbb{R}^{+}}, +, \cdot, 0, 1, \leq)$.

A *continuous module* $(\mathcal{M}, +, 0)$ *over a continuous semiring* $\mathcal{R}$ is a module over $\mathcal{R}$ having a structure of complete partial order such that 0 is the bottom and addition and scalar multiplication are continuous.

The category $\mathcal{R}^{\oplus}$. From now on, and until the end of the section, we consider fixed an arbitrary continuous semiring $\mathcal{R} = (|\mathcal{R}|, \mathbf{0}, \mathbf{1}, +, \cdot, \preceq)$, whose product $\cdot$ is commutative. Notice that $\mathcal{R}$ can be seen as a one-object category whose morphisms are the elements of $\mathcal{R}$, whose composition is the product $\cdot$, and whose identity is given by $\mathbf{1}$. Therefore, it makes sense to consider the biproduct completion $\mathcal{R}^{\oplus}$ of $\mathcal{R}$ that we defined in Section 3.3.

Given a set A and $\alpha, \alpha' \in A$, define the *Kronecker symbol* $\delta_{\alpha,\alpha'} \in \mathcal{R}$ which takes value $\mathbf{1}$ if $\alpha = \alpha'$ and $\mathbf{0}$ if $\alpha \neq \alpha'$.

Definition 3.8.3 (The category $\mathcal{R}^{\oplus}$)**.**

- *The objects of $\mathcal{R}^{\oplus}$ are sets and the morphisms from A to B are the matrices in $\mathcal{R}^{A\times B}$.*
- *Identity over A is the diagonal matrix defined as $\mathrm{Id}^{A}_{\alpha,\alpha'} := \delta_{\alpha,\alpha'}$ for all $a, a' \in A$.*
- *The composition of $f \in \mathcal{R}^{\oplus}(A,B)$ and $g \in \mathcal{R}^{\oplus}(B,C)$ is the morphism $f;g$ given by the usual matrix composition $(f;g)_{\alpha,\gamma} := \sum_{\beta\in B} f_{\alpha,\beta}\cdot g_{\beta,\gamma}$ for all $\alpha \in A$, $\gamma \in C$.*

By construction, the category $\mathcal{R}^{\oplus}$ has (countable) biproducts, represented by disjoint union and indicated as $\oplus$. Indeed, given a (possibly infinite) set I of indices we have:

$$\bigoplus_{i\in I} A_i := \bigcup_{i\in I}\{i\}\times A_i, \quad \pi^{j}_{(i,\alpha),\alpha'} := \iota^{j}_{\alpha,(i,\alpha')} := \delta_{(i,\alpha),(j,\alpha')}$$

where π^j (resp. ι^j) stands for the canonical projection on A_j (resp. injection from A_j). Moreover, given $f_j \in \mathcal{R}^{\oplus}(B, A_j)$ and $g_j \in \mathcal{R}^{\oplus}(A_j, B)$ we have that

$$(\langle f_i\rangle_{i\in I})_{\beta,(j,\alpha)} := (f_j)_{\beta,\alpha}, \qquad ([g_i]_{i\in I})_{(j,\alpha),\beta} := (g_j)_{\alpha,\beta},$$

are the unique morphisms satisfying $\langle f_i\rangle_{i\in I};\pi^j = g_j$ and $\iota^j;\,[g_i]_{i\in I} = f_j$. The terminal (actually null) object T is $\emptyset$. The hom-sets of $\mathcal{R}^{\oplus}$ inherit from $\mathcal{R}$ the structure of a continuous module: given two sets A, B, we define for all matrices $f, g \in \mathcal{R}^{A\times B}$ and scalars $p \in \mathcal{R}$:

$$0_{\alpha,\beta} := \mathbf{0}, \quad (f+g)_{\alpha,\beta} := f_{\alpha,\beta} + g_{\alpha,\beta}, \quad (pf)_{\alpha,\beta} := p\cdot f_{\alpha,\beta}.$$

Moreover, we set $f \preceq g$ if and only if $f_{\alpha,\beta} \preceq g_{\alpha,\beta}$ for all $\alpha \in A$, $\beta \in B$.

The monoidal structure. We briefly present the monoidal structure of $\mathcal{R}^{\oplus}$, showing that it is a $\star$-autonomous category (actually, compact closed).

The bifunctor $\otimes : \mathcal{R}^{\oplus}\times\mathcal{R}^{\oplus} \to \mathcal{R}^{\oplus}$ acts on objects like the cartesian product and on morphisms like the Kronecker product, that is (for every $f \in \mathcal{R}^{\oplus}(A,B)$, $g \in \mathcal{R}^{\oplus}(C,D)$):

$$A\otimes B := A\times B, \qquad (f\otimes g)_{(\alpha,\gamma),(\beta,\delta)} := f_{\alpha,\beta}\cdot g_{\gamma,\delta}$$

Bifunctoriality of this operation follows from commutativity of the $\mathcal{R}$-product $\cdot$. The unit of the tensor is the singleton set $1 := \{*\}$. The category $\mathcal{R}^{\oplus}$ is monoidal closed: the monoidal exponential object and the monoidal evaluation are defined as:

$$A \multimap B := A\times B, \qquad \mathrm{ev}^{A,B}_{((\alpha,\beta),\alpha'),\beta'} := \delta_{(\alpha,\beta),(\alpha',\beta')}, \qquad \lambda(f)_{\gamma,(\alpha,\beta)} := f_{(\gamma,\alpha),\beta}.$$

The object $\bot := \{*\}$ is dualizing, therefore $\mathcal{R}^{\oplus}$ is $\star$-autonomous.

Lafont exponentials. It turns out that $\mathcal{R}^{\oplus}$ has all symmetric tensor powers A^n, so we do not need to apply the Karoubi envelope. Indeed, for every $n \in \mathbb{N}$ and object A, the equalizer (A^n, eq^{A^n}) of the symmetries of $A^{\otimes n}$ exists and is defined by:

$$A^n := \mathcal{M}_n(A), \qquad \mathsf{eq}^{A^n}_{a,(\alpha_1,\ldots,\alpha_n)} := \delta_{a,[\alpha_1,\ldots,\alpha_n]},$$

where $\mathcal{M}_n(A)$ is the set of all multisets over A of cardinality n. These equalizers are preserved by the tensor products. Hence, we can build the exponential as in as in Section 3.3:

$$!A := \bigoplus_{n\in\mathbb{N}} A^n \cong \mathcal{M}_{\mathrm{f}}(A), \qquad \mathrm{der}^{A}_{a,\alpha} := \delta_{a,[\alpha]},$$

$$\mathrm{contr}^A_{a,(a_1,a_2)} := \delta_{a,a_1 \uplus a_2}, \qquad \mathrm{weak}^A_{a,*} := \delta_{a,[]}.$$

Given $f \in \mathcal{R}^{\oplus}(A,B)$, its promotion is the matrix $!f$ ($\forall a \in !A,\ \forall [\beta_1,\dots,\beta_n] \in !B$):

$$!f_{a,[\beta_1,\dots,\beta_n]} = \sum_{\substack{(\alpha_1,\dots,\alpha_n)\ \text{s.t.} \\ a=[\alpha_1,\dots,\alpha_n]}} \prod_{i=1}^{n} f_{\alpha_i,\beta_i}.$$

The concrete presentation of the digging is given by ($\forall a \in !A, \forall [a_1,\dots,a_n] \in !!A$):

$$\mathrm{dig}^A_{a,[a_1,\dots,a_n]} = \delta_{a,a_1 \uplus \cdots \uplus a_n}.$$

This matrix is actually the digging, since it is the unique comonoid morphism satisfying $\mathrm{dig}^A ; \mathrm{der}^{!A} = \mathrm{Id}^{!A}$. The Seely isomorphism $\mathrm{m}^{A,B}$ between $!A \otimes !B$ and $!(A \,\&\, B)$ maps the pair $([\alpha_1,\dots,\alpha_n],[\beta_1,\dots,\beta_k])$ to the multiset $[(1,\alpha_1),\dots,(1,\alpha_n),(2,\beta_1),\dots,(2,\beta_k)]$. Analogously, the isomorphism m^{T} between 1 and $!\mathsf{T}$ sends $*$ to the multiset $[]$. We treat these bijections as equalities, so we still denote by (m_1,m_2) the corresponding element of $!(A \,\&\, B)$.

The Kleisli Category. We denote by $\mathcal{R}^{\oplus}_!$ the Kleisli category of $\mathcal{R}^{\oplus}$ over the comonad $!$. The objects of $\mathcal{R}^{\oplus}_!$ are all the sets, a morphism from A to B is a matrix in $\mathcal{R}^{\mathcal{M}_{\mathrm{f}}(A)\times B}$, that is $\mathcal{R}^{\oplus}_!(A,B) := \mathcal{R}^{\oplus}(\mathcal{M}_{\mathrm{f}}(A),B)$. The composition of morphisms $f \in \mathcal{R}^{\oplus}_!(A,B)$ and $g \in \mathcal{R}^{\oplus}_!(B,C)$ is given by:

$$(f \mathbin{;_!} g)_{a,\gamma} := \sum_{[\beta_1,\dots,\beta_n] \in !B} \sum_{\substack{(a_1,\dots,a_n)\ \text{s.t.} \\ a=a_1 \uplus \cdots \uplus a_n}} g_{[\beta_1,\dots,\beta_n],\gamma} \cdot \prod_{i=1}^{n} f_{a_i,\beta_i}.$$

The identity on A is given by $\mathrm{Id}^A_{a,\alpha} := \delta_{a,[\alpha]}$. For the sake of simplicity the points of A, which are the morphisms in $\mathcal{R}^{\oplus}_!(\mathsf{T},A)$, will be represented as vectors in $\mathcal{R}^A$.

Each homset $\mathcal{R}^{\oplus}_!(A,B)$ inherits the structure of a continuous module over $\mathcal{R}$, moreover the composition is continuous and *post-linear*, that is:

$$f \mathbin{;_!} 0 = 0, \qquad f \mathbin{;_!} (g+h) = f \mathbin{;_!} g + f \mathbin{;_!} h \qquad f \mathbin{;_!} pg = p(f \mathbin{;_!} g)$$

In particular, every $f \in \mathcal{R}^{\oplus}_!(A,B)$ can be seen as a continuous map from $\mathcal{R}^A$ to $\mathcal{R}^B$ by setting $f(\mathsf{v}) := \mathsf{v} \mathbin{;_!} f$ for all vectors $\mathsf{v} \in \mathcal{R}^A$. This will be useful for interpreting fixed points.

The cartesian structure of $\mathcal{R}^{\oplus}$ is preserved in $\mathcal{R}^{\oplus}_!$, therefore the product $\&_{i\in I} A_i$ of an indexed family $(A_i)_{i\in I}$ is still the disjoint union, while the j-th projection is $\pi^j_{a,\alpha} = \delta_{a,[(j,\alpha)]}$. The exponential object $A \to B$ is given by $\mathcal{M}_{\mathrm{f}}(A) \times B$, the evaluation morphism $\mathrm{Eval} \in \mathcal{R}^{\oplus}_!((A \to B) \,\&\, A, B)$ is defined as $\mathrm{Eval}_{(a,a'),\beta} = \delta_{a,[(a',\beta)]}$ and the currying $\Lambda(f) \in \mathcal{R}^{\oplus}_!(C, A \to B)$ of a morphism $f \in \mathcal{R}^{\oplus}_!(C \,\&\, A, B)$ is given by $\Lambda(f)_{a,(a',\beta)} = f_{(a,a'),\beta}$. Notice that $\mathcal{R}^{\oplus}_!$ is a Cartesian closed differential category, where the derivative of a morphism $f : A \to B$ is defined as follows (for all $a,a' \in \mathcal{M}_{\mathrm{f}}(A)$ and $\beta \in B$):

$$\mathsf{D}(f)_{a,a',\beta} = \delta_{[\alpha],a} \cdot f_{a' \uplus [\alpha],\beta}.$$

Problem 13. *In [Lai16], Laird has shown that it is enough to start with a complete commutative semiring $\mathcal{R}$, and still obtain fixed points in $\mathcal{R}^{\oplus}_!$ without requiring any order-theoretic structure. These fixed points corresponding to infinite sums of finitary approximants indexed over the nested finite multisets, each representing a unique call-pattern for computation of the fixed point. It would be interesting to see whether the commutativity can be also released, working on the premonoidal setting, and what kind of languages it is possible to model in the coKleisli.*

Typing Rules of PCF$^{\mathcal{R}}$

$$\frac{}{\Delta, x : A \vdash x : A} \qquad \frac{\Delta, x : A \vdash M : B}{\Delta \vdash \lambda x^A.M : A \to B} \qquad \frac{\Delta \vdash M : A \to B \quad \Delta \vdash P : A}{\Delta \vdash MP : B} \qquad \frac{\Delta \vdash M : A}{\Delta \vdash pM : A}$$

$$\frac{\Delta \vdash M : A \quad \Delta \vdash P : A}{\Delta \vdash M \texttt{ or } P : A} \qquad \frac{}{\Delta \vdash \underline{0} : \mathsf{int}} \qquad \frac{\Delta \vdash M : \mathsf{int}}{\Delta \vdash \texttt{pred } M : \mathsf{int}} \qquad \frac{\Delta \vdash M : \mathsf{int}}{\Delta \vdash \texttt{succ } M : \mathsf{int}}$$

$$\frac{\Delta \vdash M : \mathsf{int} \quad \Delta \vdash P : \mathsf{int} \quad \Delta \vdash L : \mathsf{int}}{\Delta \vdash \texttt{ifz}(M, P, L) : \mathsf{int}} \qquad \frac{\Delta \vdash M : A \to A}{\Delta \vdash \texttt{fix}(M) : A}$$

Figure 3.4: Simple type system for PCF$^{\mathcal{R}}$.

3.9 PCF$^{\mathcal{R}}$: NONDETERMINISTIC PCF WITH SCALARS

Let PCF$^{\texttt{or}}$ be the extension of PCF [Plo77] with a nondeterministic choice operator "or". We now present PCF$^{\mathcal{R}}$, a prototypical programming language introduced in [M15], which extends PCF$^{\texttt{or}}$ with scalars from a continuous semiring $\mathcal{R}$. In Section 3.11 we will see that PCF$^{\mathcal{R}}$ is particularly useful as a target language in which PCF$^{\texttt{or}}$ can be interpreted. Moreover, by varying such an interpretation and the semiring $\mathcal{R}$ we are able to characterize semantically several quantitative features of the nondeterministic executions in PCF$^{\texttt{or}}$.

The language PCF$^{\mathcal{R}}$ is simply typed and has constants for representing the natural numbers, so the set of its *types* contains all arrow types that are built from the ground type int. The *terms* of PCF$^{\mathcal{R}}$ are generated by the following grammar (where $p \in \mathcal{R}$):

$$\begin{aligned} L, M, P ::= \; & x \mid \lambda x^A.M \mid MP \mid \texttt{fix}(M) \mid \underline{0} \mid \texttt{pred } M \mid \texttt{succ } M \mid \texttt{ifz}(M, P, L) \\ & \mid M \texttt{ or } P \mid pM \end{aligned}$$

Like in the language PCF$^{\texttt{or}}$ we have the simply typed λ-calculus, a fixed point operator fix for expressing recursion, an if-then-else operator $\texttt{ifz}(M, P, L)$ which tests whether the first argument M reduces to zero and chooses accordingly wither P or L, a Peano-style representation of natural numbers and an erratic choice operator or. Concerning natural numbers, for every $n \in \mathbb{N}$, we denote by $\underline{n}$ the corresponding numeral $\texttt{succ}^n(\underline{0})$.

Moreover, terms of PCF$^{\mathcal{R}}$ can be instrumented with elements of $\mathcal{R}$. Depending on the continuous semiring under consideration, the scalar p in pM can represent some sort of cost, or weight or probability associated with M.

Since the language is simply typed, the type environments Δ are handled like in the simply typed λ-calculus (see Section 1.6). As usual, type judgements are denoted by $\Delta \vdash M : A$ and can be inferred using the typing rules of Figure 3.4.

The type annotation on the λ-abstraction is not strictly necessary, but it ensures the unicity of the derivation of a valid judgement.

Lemma 3.9.1. *Given an environment Δ and a term M, there exists at most one type A such that $\Delta \vdash M : A$, and the corresponding derivation is unique.*

	Reduction Rules	Contextual Rules
β :	$(\lambda x.M)P \xrightarrow{\mathbf{1}} M[P/x]$	$\dfrac{M \xrightarrow{\mathtt{p}}_\ell M'}{MP \xrightarrow{\mathtt{p}}_\ell M'P}$
`fix` :	$\mathtt{fix}(M) \xrightarrow{\mathbf{1}} M(\mathtt{fix}(M))$	
`scal` :	$pM \xrightarrow{\mathtt{p}} M$	$\dfrac{M \xrightarrow{\mathtt{p}}_\ell M'}{\mathtt{pred}\ M \xrightarrow{\mathtt{p}}_\ell \mathtt{pred}\ M'}$
$\mathtt{or}_1$:	$M \ \mathtt{or}\ P \xrightarrow{\mathbf{1}} M$	
$\mathtt{or}_\mathtt{r}$:	$M \ \mathtt{or}\ P \xrightarrow{\mathbf{1}} P$	$\dfrac{M \xrightarrow{\mathtt{p}}_\ell M'}{\mathtt{ifz}(M,P,L) \xrightarrow{\mathtt{p}}_\ell \mathtt{ifz}(M',P,L)}$
`pred` :	$\mathtt{pred}\ \underline{n} \xrightarrow{\mathbf{1}} \underline{n-1}$	
$\mathtt{if}_0$:	$\mathtt{ifz}(\underline{0},P,L) \xrightarrow{\mathbf{1}} P$	$\dfrac{M \xrightarrow{\mathtt{p}}_\ell M'}{\mathtt{succ}\ M \xrightarrow{\mathtt{p}}_\ell \mathtt{succ}\ M'}$
$\mathtt{if}_\mathtt{s}$:	$\mathtt{ifz}(\underline{n+1},P,L) \xrightarrow{\mathbf{1}} L$	

Figure 3.5: The operational semantics of $\mathsf{PCF}^{\mathcal{R}}$. In the rule `pred` we suppose that $0-1=0$. We write $M \xrightarrow{\mathtt{p}}_\ell P$ to mean that M reduces to P using the rule (ℓ).

The operational semantics of $\mathsf{PCF}^{\mathcal{R}}$is defined in Figures 3.5.

- The reduction rules on the left are treated as relations between terms, decorated with a weight $p \in \mathcal{R}$ and a label $\ell \in \{\beta, \mathtt{fix}, \mathtt{scal}, \mathtt{or}_1, \mathtt{or}_\mathtt{r}, \mathtt{pred}, \mathtt{if}_0, \mathtt{if}_\mathtt{s}\}$.
- The *elementary reduction step* (ers) $M \xrightarrow{\mathtt{p}}_\ell P$ is the smallest relation closed under the above reduction rules and the contextual rules.

The operational semantics implements the leftmost-outermost reduction strategy. The label ℓ is needed in the elementary reduction steps to ensure that there are two distinct reductions from $M \ \mathtt{or}\ M$ to M. When writing $M \xrightarrow{\mathtt{p}} P$ we mean $M \xrightarrow{\mathtt{p}}_\ell P$ for some label ℓ. Remark that every term has at most one redex that reduces, moreover the reduction is deterministic except for the `or`-constructor. The system clearly enjoys the subject reduction.

A *reduction sequence π from M to P* is a finite sequence $(M_i \xrightarrow{p_i} M_{i+1})_{i<k}$ of elementary reduction steps such that $M_0 = M$ and $M_k = P$. In particular, for all M, there is an empty reduction sequence from M to itself. The *set of all reduction sequences from M to P of length at most k* is denoted by $M \Rightarrow^{\leq k} P$. The *set $M \Rightarrow P$ of all reduction sequences from M to P* is defined as $\bigcup_{k\in\mathbb{N}}(M \Rightarrow^{\leq k} P)$.

We have seen that elementary reduction steps are weighted, therefore it makes sense to define the weight of a (set of) reduction sequence(s).

Definition 3.9.2. *Let M, P be two terms.*

- *The* weight of a reduction sequence $\pi \in M \Rightarrow P$ *of the form* $(M_i \xrightarrow{p_i} M_{i+1})_{i<k}$ *is defined as* $\mathrm{weight}(\pi) := \prod_{i<k} p_i \in \mathcal{R}$. *In particular, the weight of the empty sequence is* $\mathbf{1}$.
- *The above operation is extended to a subset $X \subseteq M \Rightarrow P$ by setting*

$$\mathrm{weight}(X) := \sum_{\pi \in X} \mathrm{weight}(\pi)$$

$$
\begin{aligned}
[\![x_i]\!]^{\Delta}_{\vec{a},\beta} &= \delta_{a_i,[\beta]} \cdot \textstyle\prod_{j\neq i} \delta_{a_j,[]}, & [\![\lambda x^C.M]\!]^{\Delta}_{\vec{a},(c,\beta)} &= [\![M]\!]^{\Delta,x:C}_{(\vec{a},c),\beta},\\
[\![MP]\!]^{\Delta}_{\vec{a},\beta} &= \sum_{a'=[\alpha_1,\ldots,\alpha_k]} \sum_{\substack{(\vec{a}_0,\ldots,\vec{a}_k)\\ a_0 \uplus \cdots \uplus a_k = \vec{a}}} [\![M]\!]^{\Delta}_{\vec{a}_0,(a',\beta)} \cdot \prod_{i=1}^{k} [\![P]\!]^{\Delta}_{\vec{a}_i,\alpha_i},\\
[\![\underline{0}]\!]^{\Delta}_{\vec{a},n} &= \delta_{0,n} \cdot \prod_i \delta_{[],a_i}, & [\![\mathtt{pred}\ M]\!]^{\Delta}_{\vec{a},n} &= \delta_{n,0} \cdot [\![M]\!]^{\Delta}_{\vec{a},0} \uplus [\![M]\!]^{\Delta}_{\vec{a},n+1},\\
[\![\mathtt{succ}\ M]\!]^{\Delta}_{\vec{a},0} &= \mathbf{0}, & [\![\mathtt{succ}\ M]\!]^{\Delta}_{\vec{a},n+1} &= [\![M]\!]^{\Delta}_{\vec{a},n},\\
[\![\mathtt{ifz}(M,P,L)]\!]^{\Delta}_{\vec{a},n} &= \sum_{(\vec{a}_0,\vec{a}_1)\ \text{s.t.}\ \vec{a}_0+\vec{a}_1=\vec{a}} \left([\![M]\!]^{\Delta}_{\vec{a}_0,0} \cdot [\![P]\!]^{\Delta}_{\vec{a}_1,n} + \left(\sum_{k=1}^{\infty} [\![M]\!]^{\Delta}_{\vec{a}_0,k} \right) \cdot [\![L]\!]^{\Delta}_{\vec{a}_1,n} \right),\\
[\![pM]\!]^{\Delta} &= p[\![M]\!]^{\Delta}, \quad [\![M\ \mathtt{or}\ P]\!]^{\Delta} = [\![M]\!]^{\Delta} + [\![P]\!]^{\Delta}, \quad [\![\mathtt{fix}(M)]\!]^{\Delta} = \bigvee_{n\in\mathbb{N}} \mathrm{fix}^n([\![M]\!]^{\Delta}),
\end{aligned}
$$

where $\mathrm{fix}^n(\varphi)$ is defined by setting $\mathrm{fix}^0(\varphi) := 0$ and $\mathrm{fix}^{n+1}(\varphi) := \langle \varphi, \mathrm{fix}^n(\varphi)\rangle \,;_! \,\mathrm{Eval}$.

Figure 3.6: The interpretation of $\mathsf{PCF}^{\mathcal{R}}$ programs in $\mathcal{R}_!^{\oplus}$.

3.10 Denotational Semantics in $\mathcal{R}_!^{\oplus}$

Given an environment $\Delta = x_1 : A_1, \ldots, x_n : A_n$, the *interpretation of* $\Delta \vdash M : B$ *in* $\mathcal{R}_!^{\oplus}$ is a morphism $[\![M]\!]^{\Delta} \in \mathcal{R}_!^{\oplus}(A_1 \,\&\, \cdots \,\&\, A_n, B)$ that is, up to isomorphism, a matrix

$$[\![M]\!]^{\Delta} \in \mathcal{R}^{\mathcal{M}_{\mathrm{f}}(A_1)\times\cdots\times\mathcal{M}_{\mathrm{f}}(A_n)\times B}.$$

which is defined in Figure 3.6. When the underlying continuous semiring is not clear from the context we write $[\![M]\!]^{\mathcal{R},\Delta}$ to emphasize that $[\![M]\!]^{\Delta}$ lives in $\mathcal{R}_!^{\oplus}$. The continuity of $\mathcal{R}$ is exploited for interpreting the fixed point operator: indeed, for every closed term M of type A, $[\![\mathtt{fix}(M)]\!]$ is the least fixed point of $[\![M]\!]$ seen as a continuous map from $\mathcal{R}^A$ to itself.

Proposition 3.10.1 (Soundness)**.** *For every term M which is not a normal form, we have:*

$$[\![M]\!]^{\Delta} = \textstyle\sum_{M \to_{\mathrm{p}}^{\ell} L} p[\![L]\!]^{\Delta}.$$

As a consequence, we get $\mathrm{weight}(M \Rightarrow \underline{n}) \preceq [\![M]\!]_n$ for every closed term M of type int, a result relating the denotational and operational semantics of M. The adequacy for $\mathcal{R}_!^{\oplus}$ is a strengthening of such a result, saying that $[\![M]\!]_n$ and $\mathrm{weight}(M \Rightarrow \underline{n})$ are actually equal. The adequacy can be achieved following the lines of [DE11], by using suitable logical relations. More precisely, we define a relation $\lhd^A$ between vectors in $\mathcal{R}^A$ and closed terms of type A:

$$
\begin{aligned}
f \lhd^{\mathsf{int}} M &\iff \forall n \in \mathbb{N},\ f_n \preceq \mathrm{weight}(M \Rightarrow \underline{n}),\\
f \lhd^{B\to C} M &\iff \forall g, P,\ g \lhd^B P \text{ entails } \langle f, g\rangle \,;_! \,\mathrm{Eval} \lhd^C MP.
\end{aligned}
$$

By applying the fundamental lemma of logical relations (adapted to this context) we conclude $[\![M]\!] \lhd^{\mathsf{int}} M$ for all closed term M of type int.

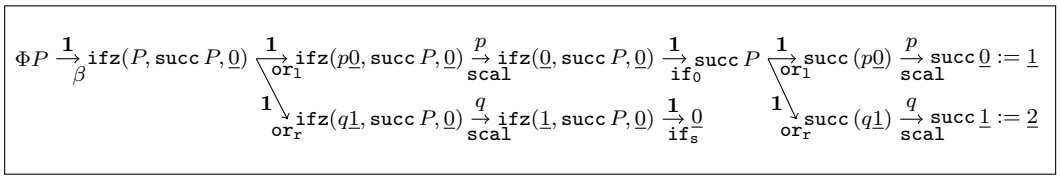


Figure 3.7: Reduction sequences starting from ΦP, where P is the weighted nondeterministic numeral $p\underline{0}$ or $q\underline{1}$.

Theorem 3.10.2 (Adequacy, Laird et Al [M15])**.**
For every closed term M of type int *and $n \in \mathbb{N}$ we have $[\![M]\!]_n = \mathrm{weight}(M \Rightarrow \underline{n})$*

Example 3.10.3. *Let us analyze the operational behaviour and denotational semantics of some specific* $\mathsf{PCF}^{\mathcal{R}}$ *terms.*

1. *The first example we consider is the fixed point of the identity $\Omega^{\mathsf{int}} := \mathtt{fix}(\lambda x^{\mathsf{int}}.x)$ which is the paradigmatic looping term of type* int*, indeed $\Omega^{\mathsf{int}} \xrightarrow{\mathbf{1}}_{\mathtt{fix}} (\lambda x^{\mathsf{int}}.x)\Omega^{\mathsf{int}} \xrightarrow{\mathbf{1}}_{\beta} \Omega^{\mathsf{int}} \xrightarrow{\mathbf{1}} \cdots$. Since the least fixed point of the identity is 0 we obtain $[\![\Omega^{\mathsf{int}}]\!] = 0$, thus $[\![M \mathtt{ or } \Omega^{\mathsf{int}}]\!] = [\![M]\!]$. It follows that, for all $n \in \mathbb{N}$, we have $\Omega \Rightarrow \underline{n} = \emptyset$ and therefore $\mathrm{weight}(\Omega \Rightarrow \underline{n}) = \mathbf{0}$.*
2. *Let us consider now the term $\Phi := \lambda x^{\mathsf{int}}.\mathtt{ifz}(x, \mathtt{succ}\, x, \underline{0})$, which is of type $\mathsf{int} \to \mathsf{int}$. The operational behaviour of Φ on numerals is easy to determine*

$$\Phi\underline{0} \xrightarrow{\mathbf{1}}_{\beta} \mathtt{ifz}(\underline{0}, \underline{1}, \underline{0}) \xrightarrow{\mathbf{1}}_{\mathtt{if_0}} \underline{1} \text{ and, for all } n > 0,\ \Phi\underline{n} \xrightarrow{\mathbf{1}}_{\beta} \mathtt{ifz}(\underline{n}, \underline{n+1}, \underline{0}) \xrightarrow{\mathbf{1}}_{\mathtt{if_s}} \underline{0}$$

   Therefore we have that $\mathrm{weight}(\Phi\underline{n} \Rightarrow \underline{k})$ is equal to $\mathbf{1}$ if either $n = 0$ and $k = 1$, or $n > 0$ and $k = 0$. Otherwise $\mathrm{weight}(\Phi\underline{n} \Rightarrow \underline{k})$ it is equal to $\mathbf{0}$.
   The reduction of Φ is more interesting when it is applied to weighted nondeterministic numerals, like $P := p\underline{0}$ or $q\underline{1}$. As shown in Figure 3.7 weights can be used to carry information on resource consumption: for instance $\mathrm{weight}(\Phi P \Rightarrow \underline{1}) = p^2$, $\mathrm{weight}(\Phi P \Rightarrow \underline{0}) = q$ and $\mathrm{weight}(\Phi P \Rightarrow \underline{2}) = p \cdot q$. The degree of the parameter p (resp. q) corresponds to the number of times the term ΦP uses the resource $p\underline{0}$ (resp. $q\underline{1}$) during the reduction to a numeral.
   Easy calculations show that the interpretation of Φ in $\mathcal{R}_!^{\oplus}$ is given by the following matrix

$$[\![\Phi]\!]_{m,n} = \begin{cases} \mathbf{1} & \text{if either } m = [0,k] \text{ and } n = k+1 \\ & \text{or } m = [k+1] \text{ and } n = 0, \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

3. *Let $\Psi := \mathtt{fix}(\lambda x^{\mathsf{int}}.(x \mathtt{ or } \underline{0}))$, which is a term of type* int*. The reduction of Φ is*

$$\Psi \xrightarrow{\mathbf{1}}_{\mathtt{fix}} (\lambda x^{\mathsf{int}}.(x \mathtt{ or } \underline{0}))\Psi \xrightarrow{\mathbf{1}}_{\beta} \Psi \mathtt{ or } \underline{0}$$

   which in its turn reduces with weight $\mathbf{1}$ using the or*-rules either to Ψ itself or to $\underline{0}$.*
   The interpretation $[\![\Psi]\!]$ can be computed starting from $[\![\lambda x^{\mathsf{int}}.(x \mathtt{ or } \underline{0})]\!]_{m,n} = \delta_{m,[n]} + \delta_{(m,n),([],0)}$ by taking the supremum for all $n \in \mathbb{N}$ of $\mathrm{fix}^n([\![\lambda x^{\mathsf{int}}.(x \mathtt{ or } \underline{0})]\!]) = [\![\underline{0}]\!] + \cdots + [\![\underline{0}]\!]$ (n times).

Failure of Full Abstraction We now show that, for every choice of $\mathcal{R}$, the model $\mathcal{R}_!^{\oplus}$ is not fully abstract for $\mathsf{PCF}^{\mathcal{R}}$ — it does not capture exactly the observational pre-order on terms induced by $\mathcal{R}$. Let us denote by $\mathscr{C}_B^{\Delta,A}$ the set of $\mathsf{PCF}^{\mathcal{R}}$ contexts $Q(\!|-|\!)$ mapping terms M of type A in Δ, into terms $Q(\!|M|\!)$ of type B in the empty environment.

Definition 3.10.4 (Observational pre-order)**.** *Given $\Delta \vdash M : A$ and $\Delta \vdash P : A$, define*

$$M \sqsubseteq^{\Delta} P \iff \forall Q \in \mathscr{C}_{\mathsf{int}}^{\Delta,A}, \mathrm{weight}(Q(\!|M|\!) \Rightarrow \underline{0}) \preceq \mathrm{weight}(Q(\!|P|\!) \Rightarrow \underline{0}).$$

Let $\equiv^{\Delta}$ be the equivalence induced by $\sqsubseteq^{\Delta}$.

Remark that the numeral $\underline{0}$ chosen for testing the equality is not really significant. Indeed, from a context $Q(\!|-|\!)$ such that $\mathrm{weight}(Q(\!|M|\!) \Rightarrow \underline{0}) \not\preceq \mathrm{weight}(Q(\!|P|\!) \Rightarrow \underline{0})$, one can define the context $Q'(\!|-|\!) := \mathtt{succ}^n(Q(\!|-|\!))$ satisfying $\mathrm{weight}(Q'(\!|M|\!) \Rightarrow \underline{n}) \not\preceq \mathrm{weight}(Q'(\!|P|\!) \Rightarrow \underline{n})$.

The model $\mathcal{R}_!^{\oplus}$ would be *(inequationally) fully abstract* if, for all terms M and P, we have $[\![M]\!]^{\Delta} \preceq [\![P]\!]^{\Delta}$ if and only if $M \sqsubseteq^{\Delta} P$. As a corollary of the adequacy, we get the 'only if' direction, that is the fact that $[\![M]\!]^{\Delta} \preceq [\![P]\!]^{\Delta}$ entails $M \sqsubseteq^{\Delta} P$. We now show that the other implication does not hold. Let us consider the following $\mathsf{PCF}^{\mathcal{R}}$ programs:

$$\Xi := \lambda y^{\mathsf{int}}.\boldsymbol{\infty}\underline{0}, \qquad \Upsilon := \lambda y^{\mathsf{int}}.(\boldsymbol{\infty}\underline{0} \ \mathtt{or}\ \mathtt{ifz}(y, \underline{0}, \Omega^{\mathsf{int}})). \tag{3.1}$$

where Ω^{int} is defined in Example 3.10.3 and $\boldsymbol{\infty} = \sum_{p \in \mathcal{R}} p$ as in Section 3.8. It is easy to check that both terms have type $\mathsf{int} \to \mathsf{int}$. By using the rules of Figure 3.6 one can easily compute their interpretations:

- $[\![\Xi]\!]_{[],0} = \boldsymbol{\infty}$ and $[\![\Xi]\!]_{m,n} = \mathbf{0}$ otherwise,
- $[\![\Upsilon]\!]_{[],0} = \boldsymbol{\infty}$, $[\![\Upsilon]\!]_{[0],0} = \mathbf{1}$ and $[\![\Upsilon]\!]_{m,n} = \mathbf{0}$ otherwise.

Note that $[\![\Xi]\!] \prec [\![\Upsilon]\!]$, indeed $[\![\Xi]\!]_{[0],0} = \mathbf{0} \prec \mathbf{1} = [\![\Upsilon]\!]_{[0],0}$. However, the two terms are observationally equivalent — this can be proved by a standard reasoning using the logical relation $\lhd^A$ to shrink the set of the contexts observing the operational behaviour of Ξ and Υ. Indeed, it is possible to show that $[\![\Upsilon]\!] \lhd^{\mathsf{int}\to\mathsf{int}} \Xi$ holds.

Proposition 3.10.5. *Υ and Ξ are observationally equivalent.*

Proof. For any context $Q(\!|-|\!) \in \mathscr{C}_{\mathsf{int}}^{\mathsf{int}\to\mathsf{int}}$ and closed term M of type $\mathsf{int} \to \mathsf{int}$, we have $(\lambda x^{\mathsf{int}\to\mathsf{int}}.Q(\!|x|\!))M \xrightarrow{\mathbf{1}} Q(\!|M|\!)$. Therefore we have $M \sqsubseteq M'$ if and only if $\mathrm{weight}(LM \Rightarrow \underline{0}) \preceq \mathrm{weight}(LM' \Rightarrow \underline{0})$, for every closed term $L : (\mathsf{int} \to \mathsf{int}) \to \mathsf{int}$. From the fundamental lemma of logical relations we get $[\![L]\!] \lhd^{(\mathsf{int}\to\mathsf{int})\to\mathsf{int}} L$, hence from $[\![\Upsilon]\!] \lhd^{\mathsf{int}\to\mathsf{int}} \Xi$ and Theorem 3.10.2, we obtain $\mathrm{weight}(L\Upsilon \Rightarrow \underline{0}) = [\![L\Upsilon]\!]_0 = (\langle [\![L]\!], [\![\Upsilon]\!] \rangle ;_! \mathrm{Eval})_0 \preceq \mathrm{weight}(L\Xi \Rightarrow \underline{0})$. This gives $\Upsilon \sqsubseteq \Xi$, the converse follows from $[\![\Xi]\!] \preceq [\![\Upsilon]\!]$ and the adequacy. □

This is even a counterexample to *equational* full abstraction as we found two terms Ξ, Υ such that $[\![\Xi]\!] \neq [\![\Upsilon]\!]$ but $\Xi \equiv \Upsilon$. Notice that Counterexample (3.1) can be rephrased without using scalar multiplication as soon as $\mathcal{R}$ is such that $\boldsymbol{\infty} = \sum_{n \in \mathbb{N}} \mathbf{1} + \cdots + \mathbf{1}$. (This is the case for all semirings in Example 3.8.2.) Indeed, under this hypothesis, the term Ψ of Example 3.10.3(3) has the same observational and denotational semantics of $\boldsymbol{\infty}\underline{0}$.

Problem 14. *Show that $\mathcal{R}_!^{\oplus}$ is a fully abstract model for resource* PCF *extended with scalars from $\mathcal{R}$. Is the model fully abstract for an Erratic Idealized Algol with scalars?*

3.11 CHARACTERIZING QUANTITATIVE PROPERTIES

In this section we show how, choosing appropriate continuous semirings $\mathcal{R}$, it is possible to capture semantically several quantitative operational properties of programs.

We analyse $\mathsf{PCF}^{\mathtt{or}}$, the restriction of $\mathsf{PCF}^{\mathcal{R}}$ obtained by forbidding the rule pM in the grammar of $\mathsf{PCF}^{\mathcal{R}}$ given in Section 3.9, so that the weight of any reduction sequence is $\mathbf{1}$. This has a natural translation into $\mathsf{PCF}^{\mathcal{R}}$, of course, since it is merely a restriction of that language. Here we shall see that other translations, obtained by *instrumenting* $\mathsf{PCF}^{\mathtt{or}}$ terms with elements of $\mathcal{R}$ using the pM rule, allow us to refine the semantics to various quantitative purposes. Thus $\mathsf{PCF}^{\mathcal{R}}$ is used as a semantic metalanguage, capable of describing a range of different quantitative models of $\mathsf{PCF}^{\mathtt{or}}$.

May/Must Nondeterministic Convergence

The most basic behaviour to observe is whether a $\mathsf{PCF}^{\mathtt{or}}$ program (closed term of type int) M *may-converges* to a numeral $\underline{n}$, that is whether there exists a reduction sequence from M to $\underline{n}$. For instance the term Ψ may-converges to $\underline{0}$, while the term Ω^{int} does not. To observe such a behaviour it is enough to consider the simplest (non-trivial) continuous semiring, that is the Boolean semiring $\mathcal{B}$ of Example 3.8.2(1). Theorem 3.10.2 specializes to the following characterization of may-convergence.

Corollary 3.11.1. *For every program M of $\mathsf{PCF}^{\mathtt{or}}$, $[\![M]\!]^{\mathcal{B}}_n = \mathbf{t}$ if and only if M may-converges to $\underline{n}$.*

Note that $\mathcal{B}^{\oplus}_!$ is isomorphic to the category **MRel**, known as the *relational semantics*. Therefore, this first result is not very surprising as **MRel** has been proved to characterize may-convergence for a resource sensitive extension of $\mathsf{PCF}^{\mathtt{or}}$ [M14].

Starting from the standard semiring $\mathcal{N}$ of Example 3.8.2(2) we already get a much finer observation on programs. Indeed $\mathrm{weight}(M \Rightarrow \underline{n})$ becomes equal to the number of paths in $M \Rightarrow \underline{n}$. This means that $\mathcal{N}^{\oplus}_!$ is able to compare programs depending on how many reduction sequences lead to a certain numeral.

Corollary 3.11.2. *For every program M of $\mathsf{PCF}^{\mathtt{or}}$, $[\![M]\!]^{\mathcal{N}}_n$ is the number of reduction sequences from M to $\underline{n}$.*

For instance, we have $[\![\Psi]\!]^{\mathcal{N}}_0 = \infty$ and $[\![\Phi(\underline{1}\ \mathtt{or}\ \underline{1})]\!]^{\mathcal{N}}_0 = 2$, so $\mathcal{N}^{\oplus}_!$ separates the two terms, while $\mathcal{B}^{\oplus}_!$ gives the same interpretation to both.

The characterization of *must-convergence* (i.e. the convergence to a numeral $\underline{n}$ regardless of the erratic choices taken during the evaluation) requires a more complex translation of $\mathsf{PCF}^{\mathtt{or}}$ into $\mathsf{PCF}^{\mathcal{N}}$, allowing detection of potentially infinite reductions. For instance, as shown in Example 3.10.3, the programs $\Phi\underline{1}\ \mathtt{or}\ \Omega^{\mathsf{int}}$ and $\Phi\underline{1}$ have the same interpretation for any choice of $\mathcal{R}$, but the first term is not must-convergent while the second is.

Let us consider the translation $(-)^{\circ}_{\Gamma}$ mapping judgments $\Gamma \vdash_{\mathsf{PCF}^{\mathtt{or}}} - : A$ into judgments $\Gamma \vdash_{\mathsf{PCF}^{\mathcal{N}}} - : A$ which is generated by (assuming M of type $B \to B$ and L of type B, with $B = B_1 \to \cdots \to B_k \to \mathsf{int}$):

$$(\mathtt{fix}(M))^{\circ}_{\Gamma} := \mathtt{fix}(\lambda x^{B}.((M)^{\circ}_{\Gamma} x\ \mathtt{or}\ \lambda y_1^{B_1} \ldots \lambda y_k^{B_k}.\underline{0})),$$
$$(\lambda x^{C}.L)^{\circ}_{\Gamma} := \lambda x^{C}.((L)^{\circ}_{\Gamma;x:C}\ \mathtt{or}\ \lambda y_1^{B_1} \ldots \lambda y_k^{B_k}.\underline{0}),$$

where *generated by* means that $(-)^{\circ}_{\Gamma}$ commutes with all other constructors of $\mathsf{PCF}^{\mathtt{or}}$. From now on we will consider $\mathsf{PCF}^{\mathtt{or}}$ programs, so the environment will be omitted.

For all programs M, P of $\mathsf{PCF}^{\mathtt{or}}$, we have $M \to_\ell P$ if and only if one of the following conditions holds:

- $\ell = \mathtt{fix}$ and $M^\circ \xrightarrow{1}_{\mathtt{fix}} \xrightarrow{1}_{\beta} \xrightarrow{1}_{\mathtt{or_1}} P^\circ$,
- $\ell = \beta$ and $M^\circ \xrightarrow{1}_{\beta} \xrightarrow{1}_{\mathtt{or_1}} P^\circ$,
- $\ell \notin \{\mathtt{fix}, \beta\}$ and $M^\circ \xrightarrow{1}_{\ell} P^\circ$.

Lemma 3.11.3. *For every* $\mathsf{PCF}^{\mathtt{or}}$ *program* M*, there exists a reduction sequence from* M° *to* $\underline{n}$*, for some* $n \in \mathbb{N}$*.*

As a first corollary we obtain a characterization of strong convergence — a $\mathsf{PCF}^{\mathtt{or}}$ program M is *strongly converging* if there is no infinite reduction sequence starting from M.

Corollary 3.11.4. *A* $\mathsf{PCF}^{\mathtt{or}}$ *program* M *is strongly converging if and only if* $\sum_{n\in\mathbb{N}} [\![M^\circ]\!]^{\mathcal{N}}_n < \infty$*.*

For instance, $(\Omega^{\mathsf{int}})^\circ = \mathtt{fix}(\lambda x^{\mathsf{int}}.((\lambda x^{\mathsf{int}}.(x \ \mathtt{or}\ \underline{0}))x \ \mathtt{or}\ \underline{0}))$, and $\sum_{n\in\mathbb{N}} [\![(\Omega^{\mathsf{int}})^\circ]\!]^{\mathcal{N}}_n = \infty$ as $[\![(\Omega^{\mathsf{int}})^\circ]\!]^{\mathcal{N}}_0 = \infty$. Finally, from Corollaries 3.11.1 and 3.11.4, we obtain the following characterization of must-convergence.

Corollary 3.11.5. *A* $\mathsf{PCF}^{\mathtt{or}}$ *program* M *must-converges to a numeral* $\underline{n}$ *if and only if* $\sum_{k\in\mathbb{N}} [\![M^\circ]\!]^{\mathcal{N}}_k < \infty$*,* $[\![M]\!]^{\mathcal{N}}_n > 0$ *and* $[\![M]\!]^{\mathcal{N}}_k = 0$ *for all* $k \neq n$*.*

Probabilistic Convergence

Let us now determine the probability that a $\mathsf{PCF}^{\mathtt{or}}$ program reduces to a numeral $\underline{n}$, supposing that the probability of applying $\mathtt{or_l}$ or $\mathtt{or_r}$ when firing an $\mathtt{or}$-redex is uniformly distributed. In the spirit of [DE11], this amounts to define its operational semantics through a Markov system having the terms as states, and the normal forms as absorbing states.

The Markov matrix describing such a process is given by:

$$\mathrm{Red}_{M,P} := \begin{cases} 1 & \text{if } P = M \text{ is a normal form,} \\ 1 & \text{if } M \to_\ell P \text{ with } \ell \notin \{\mathtt{or_l}, \mathtt{or_r}\}, \\ 1 & \text{if } M \to_{\mathtt{or_l}} P \text{ and } M \to_{\mathtt{or_r}} P, \\ 0.5 & \text{if } M \to_{\mathtt{or_l}} P \text{ but } M \not\to_{\mathtt{or_r}} P \text{ or viceversa,} \\ 0 & \text{otherwise.} \end{cases}$$

Note that Red is a stochastic matrix (i.e. $\sum_P \mathrm{Red}_{M,P} = 1$), and that $\mathrm{Red}_{M,P}$ describes the probability of evolving from M to P in one ers. Similarly, the k-th fold matrix product Red^k, which is still a stochastic matrix, gives the evolution of the system after k steps. Since $\underline{n}$ is absorbing, $\mathrm{Red}^k_{M,\underline{n}}$ is monotone in k and bounded by 1, so $\mathrm{Red}^\infty_{M,\underline{n}} := \sup_{k\in\mathbb{N}} \mathrm{Red}^k_{M,\underline{n}}$ is well-defined and gives the probability that M reduces to $\underline{n}$ in finitely many elementary reduction steps.

To capture this probabilistic feature in our semantic framework, consider the semiring $\mathcal{P}$ of Example 3.8.2(5) and the translation $(-)^\circ : \mathsf{PCF}^{\mathtt{or}} \to \mathsf{PCF}^{\mathcal{P}}$ generated by:

$$(M \ \mathtt{or}\ P)^\circ := (0.5\ M^\circ) \ \mathtt{or}\ (0.5\ P^\circ).$$

Note that a reduction step $M \to_\ell P$ can be simulated by $M^\circ \xrightarrow{1}_\ell P^\circ$ when ℓ is not an $\mathtt{or}$-rule, otherwise we need two steps $M^\circ \xrightarrow{1}_\ell \xrightarrow{0.5}_{\mathtt{scal}} P^\circ$.

Lemma 3.11.6. *For every program M of* $\mathsf{PCF}^{\mathtt{or}}$ *and $n \in \mathbb{N}$, we have* $\mathrm{weight}(M^\circ \Rightarrow \underline{n}) = \mathrm{Red}^\infty_{M,\underline{n}}$.

As a consequence we get the following result, restating for $\mathcal{P}^\oplus_!$ the adequacy theorem proved in [DE11] for the category $\mathbf{PCoh}_!$ of probabilistic coherence spaces and entire functions.

Corollary 3.11.7. *For every program M of* $\mathsf{PCF}^{\mathtt{or}}$, $[\![M^\circ]\!]^{\mathcal{P}}_n = \mathrm{Red}^\infty_{M,\underline{n}}$ *which is the probability that M reduces to $\underline{n}$.*

For example, $[\![(\Phi\underline{1})^\circ]\!]^{\mathcal{P}} = [\![\Psi^\circ]\!]^{\mathcal{P}}$, both giving 1 on the web element 0. Notice also that, omitting the translation, $[\![\Phi\underline{1}]\!]^{\mathcal{P}}_0 = 1$ while $[\![\Psi]\!]^{\mathcal{P}}_0 = \infty$.

The two models $\mathcal{P}^\oplus_!$ and $\mathbf{PCoh}_!$ share the same interpretations on probabilistic programs (i.e. on the image of the translation), since there is a faithful forgetful functor from $\mathbf{PCoh}_!$ to $\mathcal{P}^\oplus_!$ which acts like the identity on morphisms. These categories however differ in a crucial property, namely the fact that $\mathbf{PCoh}_!$ is well-pointed, while $\mathcal{P}^\oplus_!$ is not (the counterexample being given by the maps $[\![\Xi]\!]$ and $[\![\Upsilon]\!]$).

Resource Analysis.

We wish now to determine the minimum number of times that a β- or a `fix`-redex is contracted during an evaluation of a $\mathsf{PCF}^{\mathtt{or}}$ program M (*best case analysis*), or the maximum number (*worst case analysis*). These are indeed the two most critical redexes from the point of view of resource consumption, as their contraction may increase the size of M.

The model built from the tropical semiring $\mathcal{T}$ of Example 3.8.2(3) computes the best case analysis, through the translation $(-)^\circ : \mathsf{PCF}^{\mathtt{or}} \to \mathsf{PCF}^{\mathcal{T}}$ generated by:

$$(\lambda x^A.M)^\circ := \lambda x^A.1M^\circ, \qquad (\mathtt{fix}(M))^\circ := \mathtt{fix}(1M^\circ).$$

Recall that in $\mathcal{T}$ the product is $+$ and $\mathbf{1} := 0$, so $1 \neq \mathbf{1}$.

Lemma 3.11.8. *For all* $\mathsf{PCF}^{\mathtt{or}}$ *terms M, P we have $M \to_\ell P$ if and only if either $\ell \in \{\beta, \mathtt{fix}\}$ and $M^\circ \xrightarrow{0}_\ell \xrightarrow{1}_{\mathtt{scal}} P^\circ$ or $\ell \notin \{\beta, \mathtt{fix}\}$ and in that case $M^\circ \xrightarrow{0}_\ell P^\circ$.*

Therefore, given a reduction sequence $\pi \in M^\circ \Rightarrow \underline{n}$, its weight $\mathrm{weight}(\pi)$ gives the number of β- and `fix`-redexes contracted in π. Since the addition of $\mathcal{T}$ is $\min$ (with respect to the standard order on $\overline{\mathbb{N}}$), we have the following result.

Corollary 3.11.9. *For every program M of* $\mathsf{PCF}^{\mathtt{or}}$, $[\![M^\circ]\!]^{\mathcal{T}}_n$ *is the minimum number of β- and* `fix`*-redexes reduced in a reduction sequence from M to $\underline{n}$.*

For the worst case analysis, consider the model built from the arctic semiring $\mathcal{A}$ (Example 3.8.2.4), where the addition is $\max$, and the translation $(-)^\circ : \mathsf{PCF}^{\mathtt{or}} \to \mathsf{PCF}^{\mathcal{A}}$ is defined as before. An analogous reasoning gives the next corollary.

Corollary 3.11.10. *For every program M of* $\mathsf{PCF}^{\mathtt{or}}$, $[\![M^\circ]\!]^{\mathcal{A}}_n$ *is the maximum number of β- and* `fix`*- redexes reduced in a reduction sequence from M to $\underline{n}$.*

For instance, we have $[\![(\Phi((\lambda x^{\mathsf{int}}.x)\underline{0}))^\circ]\!]^{\mathcal{T}} > [\![(\mathtt{succ}\,\Psi)^\circ]\!]^{\mathcal{T}}$, namely $[\![(\Phi((\lambda x^{\mathsf{int}}.x)\underline{0}))^\circ]\!]^{\mathcal{T}}_1 = 3$ and $[\![(\mathtt{succ}\,\Psi)^\circ]\!]^{\mathcal{T}}_1 = 2$. On the other hand, we have $[\![(\Phi((\lambda x^{\mathsf{int}}.x)\underline{0}))^\circ]\!]^{\mathcal{A}} < [\![(\mathtt{succ}\,\Psi)^\circ]\!]^{\mathcal{A}}$, in fact $[\![(\Phi((\lambda x^{\mathsf{int}}.x)\underline{0}))^\circ]\!]^{\mathcal{A}}_1 = 3$ and $[\![(\mathtt{succ}\,\Psi)^\circ]\!]^{\mathcal{A}}_1 = \infty$.

4 *Factor Algebras and Symbolic Computation*

In which we show that the theory of decomposition operator can be fruitfully applied to the study of λ-calculus, multi-valued matrix logics, and first-order classical logic.

- **From Lambda Calculus to Universal Algebra and Back.**
  G. Manzonetto and A. Salibra.
  33rd International Symposium on Mathematical Foundations of Computer Science (MFCS'08), volume 5162 of LNCS, pages 479-490, 2008.
- **Lattices of Equational Theories as Church Algebras.**
  G. Manzonetto and A. Salibra.
  In C. Drossos, P. Peppas, and C. Tsinakis, eds, 7th Panhellenic Logic Symposium, pages 117-121. Patras University Press, 2009.
- **Applying Universal Algebra to Lambda Calculus.**
  G. Manzonetto and A. Salibra.
  Special issue of Journal of Logic and Computation on Logic and Algebra, Volume 20, Number 4, pages 877-915, 2010.
- **Factor Varieties and Symbolic Computation.**
  A. Salibra, G. Manzonetto and G. Favro.
  ACM/IEEE Symposium on Logic in Computer Science (LICS'16), pages 738-747, New York, USA, 2016.

Universal algebra has many applications in computer science: indeed, whenever a specification can be modelled by sets and functions, there is an algebra. However, combinatory algebras, that constitute the models of λ-calculus, were considered algebraically pathological as they are never commutative, associative, finite or recursive.

In my PhD thesis (2005-2008) and subsequently during my ATER (2008) I fruitfully applied, in collaboration with Salibra, techniques of universal algebra based on decomposition operators to study models and theories of λ-calculus, thus showing that this belief was not well-founded. The insights gained from the analysis of combinatory algebras led us to define the more general class of Church algebras that also encompass important algebraic structures like Boolean algebras, Heyting algebras and rings with unit. Church algebras enjoy many properties, including a Stone representation theorem having interesting consequences in the study of λ-calculus and of lattices of equational theories.

In 2015, Salibra has spent one month at the laboratory LIPN as an invited professor, together with his PhD student Favro. In this occasion we discovered that the theory of decomposition operators can also be applied to study classical logic and, mode generally, to any finite multi-valued matrix logic. We developed a uniform method for extracting the logical content of a formula and determining whether the formula is a tautology. We have shown that this method can be automatized using a confluent and terminating term rewriting system, thus creating a bridge towards proof assistants and SAT-solvers. Our approach also suggests a new notion of logical circuit and a procedure to reduce the satisfiability problem of first-order logic to an equational problem in a suitable variety of algebras that deserve to be investigated further.

4.1 ALGEBRAS AND FACTORIZATIONS

In this technical section, we recall some concepts of universal algebra that will be useful in the sequel. We mainly use the terminology and notations from [MMT87] and [BS81].

Algebras and varieties. An *algebraic type* τ is constituted by a non-empty set of function symbols together with their *arity*. We write $f \in \tau_n$ if the function symbol $f \in \tau$ has arity n. An *algebra* $\mathbf{A}$ *of type* τ, or *a* τ*-algebra*, is determined by a set $A \neq \emptyset$ together with a function $f^{\mathbf{A}} : A^n \to A$ for every $f \in \tau_n$. An algebra $\mathbf{A}$ is *trivial* if $|A| = 1$, otherwise $\mathbf{A}$ is *proper*.

A compatible equivalence relation φ on a τ-algebra $\mathbf{A}$ is called a *congruence*. As a matter of notation, we write $a \,\varphi\, b$ for $(a, b) \in \varphi$ and $[a]_\varphi$ for the equivalence class $\{a' \in A \mid a \,\varphi\, a'\}$. We denote by $\mathrm{Con}(\mathbf{A})$ the algebraic complete lattice of all congruences on $\mathbf{A}$, ordered by inclusion. The congruences $\Delta = \{(a, a) \mid a \in A\}$ and $\nabla = A \times A$ constitute the bottom and the top elements of $\mathrm{Con}(\mathbf{A})$, respectively. A congruence φ is called: *trivial* if it is equal to Δ or ∇; *consistent* if it is different from ∇; *compact* if it is finitely generated.

Given $a, b \in A$, we write $\vartheta(a, b)$ for the *principal congruence generated by equating* a *and* b, that is for the smallest congruence relating them. Given two congruences φ and ϑ on $\mathbf{A}$, we can form their *relative product* by setting $\varphi \circ \vartheta = \{(a, c) \mid \exists b \in A,\ a \,\vartheta\, b \,\varphi\, c\}$.

A class V of τ-algebras is a *variety* if it is closed under subalgebras, direct product and homomorphic images. By Birkhoff theorem [Bir35] a class of algebras is variety if and only if it is an *equational class*, which means that it is axiomatizable by a set of equations.

Factorization of algebras. An algebra $\mathbf{A}$ is *directly decomposable* if there exist two non-trivial algebras $\mathbf{B}, \mathbf{C}$ such that $\mathbf{A} \cong \mathbf{B} \times \mathbf{C}$, otherwise it is called *directly indecomposable*.

An algebra $\mathbf{A}$ is a *subdirect product* of the algebras $(\mathbf{B}_i)_{i\in I}$, written $\mathbf{A} \leq \Pi_{i\in I}\mathbf{B}_i$, if there exists an embedding f of $\mathbf{A}$ into the direct product $\Pi_{i\in I}\mathbf{B}_i$ such that the projection $\pi_i \circ f : \mathbf{A} \to \mathbf{B}_i$ is surjective for every $i \in I$.

Definition 4.1.1. *A family* $(\varphi_i)_{i\in I}$ *of congruences on* $\mathbf{A}$ *is a* family of complementary factor congruences *if the function*

$$f : \mathbf{A} \to \textstyle\prod_{i\in I}(\mathbf{A}/\varphi_i)$$

defined by $f(a) = (a/\varphi_i)_{i\in I}$ *is an isomorphism. When* $|I| = 2$*, we say that* (φ_1, φ_2) *is a* pair of complementary factor congruences.

A *factor congruence* is any congruence which belongs to a family of complementary factor congruences.

Proposition 4.1.2. *A family* $(\varphi_i)_{i\in I}$ *of congruences on* $\mathbf{A}$ *is a family of complementary factor congruences exactly when:*

1. $\bigcap_{i\in I} \varphi_i = \Delta$*;*
2. $\forall a \in A^I$*, there is* $u \in A$ *such that* $a_i \,\varphi_i\, u$*, for all* $i \in I$*.*

Therefore (φ_1, φ_2) is a pair of complementary factor congruences if and only if $\varphi_1 \cap \varphi_2 = \Delta$ and $\varphi_1 \circ \varphi_2 = \nabla$. The pair (Δ, ∇) corresponds to the product $\mathbf{A} \cong \mathbf{A} \times \mathbf{1}$, where $\mathbf{1}$ is the trivial algebra having a singleton as carrier set; obviously $\mathbf{1} \cong \mathbf{A}/\nabla$ and $\mathbf{A} \cong \mathbf{A}/\Delta$. The set of factor congruences of $\mathbf{A}$ is not, in general, a sublattice of $\mathrm{Con}(\mathbf{A})$.

We say that an algebra $\mathbf{A}$ is *subdirectly irreducible* if the lattice $\mathrm{Con}(\mathbf{A})$ has a unique atom; *simple* if $\mathrm{Con}(\mathbf{A}) = \{\Delta, \nabla\}$. The algebra $\mathbf{A}$ is directly indecomposable if and only if it admits only the two trivial factor congruences. Any simple algebra is subdirectly irreducible and any subdirectly irreducible algebra is directly indecomposable.

4.2 DECOMPOSITION OPERATORS

Factor congruences can be characterized in terms of certain algebra homomorphisms called *decomposition operators* and acting on sequences. We refer the reader to [MMT87, Def. 4.32] for a thoughtful presentation.

Given a set A and a set of indices I we define an *I-sequence $\vec{x}$ on A* as a map $\vec{x} : I \to A$. For every index $i \in I$ and element $a \in A$ we denote by $\vec{x}\{a/i\}$ the I-sequence which coincides with $\vec{x}$, except on i, where it takes the value a. Given $a \in A$ we let a^I denote the constant sequence taking value a for all indices $i \in I$.

Definition 4.2.1.
A decomposition operator *on an algebra $\mathbf{A}$ is a function $f : A^I \to A$ satisfying the following conditions:*

D1 $f(a^I) = a$, *for all* $a \in A$;

D2 $f(f(a_{ij})_{j\in I})_{i\in I} = f(a_{ii})_{i\in I}$;

D3 f *is an algebra homomorphism from* $\mathbf{A}^I$ *to* $\mathbf{A}$.

Remark 4.2.2. *If I is finite, then the axioms (D1)-(D3) can be expressed equationally. This will always be the case for the decomposition operators that we consider in the following sections.*

There is a bijective correspondence between families of complementary factor congruences and decomposition operators, and thus, between decomposition operators and factorizations.

Proposition 4.2.3.
Any decomposition operator $f : \mathbf{A}^I \to \mathbf{A}$ on an algebra $\mathbf{A}$ induces a family of complementary factor congruences $(\varphi_i)_{i\in I}$ where each $\varphi_i \subseteq A \times A$ is defined by:

$$a \; \varphi_i \; b \text{ if and only if } f(a^I\{b/i\}) = a.$$

Conversely, any family $(\varphi_i)_{i\in I}$ of complementary factor congruences induces a decomposition operator f on $\mathbf{A}$:

$$f(\vec{x}) = u \text{ if and only if } x_i \; \varphi_i \; u, \text{ for all } i \in I.$$

Indeed, it is possible to prove that such an element u is unique.

The Boolean product construction allows to transfer numerous fascinating properties of Boolean algebras into other varieties of algebras (see [BS81, Ch. IV]).

We recall that a *Boolean space* is a compact, Hausdorff and totally disconnected topological space, and that *clopen* means "open and closed".

Definition 4.2.4.
A weak Boolean product *of a family $(\mathbf{A}_i)_{i\in I}$ of algebras is a subdirect product $\mathbf{A} \leq \Pi_{i\in I}\mathbf{A}_i$, where I can be endowed with a Boolean space topology such that:*

(i) the set $\{i \in I \mid a_i = b_i\}$ is open for all $a, b \in A$, and

(ii) if $a, b \in A$ and N is a clopen subset of I, then the element c, defined by $c_i = a_i$ for every $i \in N$ and $c_i = b_i$ for every $i \in I - N$, belongs to A.

It is called a Boolean product *whenever the set $\{i \in I \mid a_i = b_i\}$ is clopen for all $a, b \in A$.*

4.3 CHURCH ALGEBRAS AND VARIETIES

In [M23] we introduced, together with Salibra, the *Church algebras*, that are algebras modelling the "if-then-else" operator of programming languages. Perhaps surprisingly, the variety generated by the following two identities have never been studied before.

Definition 4.3.1. *A τ-algebra $\mathbf{A}$ is called* a Church algebra *if there are two constants $0, 1 \in A$ and a ternary term $\mathsf{ite}(e, x, y)$ such that*

$$\mathsf{ite}(1, x, y) = x, \qquad \mathsf{ite}(0, x, y) = y.$$

A variety V is called a Church variety *if every algebra in V is a Church algebra with respect to the same term $\mathsf{ite}(e, x, y)$ and constants $0, 1$.*

The class of Church algebra is general enough to encompass combinatory algebras, Boolean algebras, Heyting algebras, and rings with unit. For instance, in a combinatory algebra, the constants 1 and 0 correspond to the first and second projection respectively, so the if-then-else operator is simply defined by setting $\mathsf{ite}(e, x, y) := exy$. In a Boolean algebra we can define $\mathsf{ite}(e, x, y) := (e \vee y) \wedge (e^- \vee x)$, in a Heyting algebra $\mathsf{ite}(e, x, y) := (e \vee y) \wedge ((e \to 0) \vee x)$ and in a ring with unit $\mathsf{ite}(e, x, y) := (y + e - ey)(1 - e + ex)$.

Decomposition by central elements. Pierce showed that every idempotent element a of a commutative ring $\mathbf{A}$ induces a pair $(\vartheta(1, a), \vartheta(a, 0))$ of complementary factor congruences [Pie67]. In other words, the ring $\mathbf{A}$ can be decomposed as $\mathbf{A} \cong \mathbf{A}/\vartheta(1, a) \times \mathbf{A}/\vartheta(a, 0)$. Moreover, $\mathbf{A}$ is directly indecomposable if 0 and 1 are its unique idempotent elements. In [Vag96], Vaggione generalized idempotent elements to any algebra whose top congruence ∇ is compact, and called them *central elements*. Central elements were used, among other things, to investigate the closure of varieties of algebras under Boolean products. In the case of Church algebras central elements admit a new equational characterization.

Definition 4.3.2. *An element e of a Church algebra $\mathbf{A}$ is* central *if it satisfies:*

1. $\mathsf{ite}(e, x, x) = x$.
2. $\mathsf{ite}(e, \mathsf{ite}(e, x, y), z) = \mathsf{ite}(e, x, z) = \mathsf{ite}(e, x, \mathsf{ite}(e, y, z))$.
3. $\mathsf{ite}(e, f(x_1, \dots, x_n), f(y_1, \dots, y_n)) = f(\mathsf{ite}(e, x_1, y_1), \dots, \mathsf{ite}(e, x_n, y_n))$, *for every $f \in \tau_n$.*
4. $e = \mathsf{ite}(e, 1, 0)$.

We denote by $\mathrm{Ce}(\mathbf{A})$ the set of central elements of $\mathbf{A}$.

It is easy to check that all elements of a Boolean algebra are central, while an element of a commutative ring with unit is central if and only if it is idempotent.

Every central element e induces a pair of complementary factor congruences given by $\vartheta_e := \vartheta(1, e)$ and $\overline{\vartheta}_e := \vartheta(e, 0)$. The central elements 0 and 1 are called *trivial* since they induce the trivial decomposition $\mathbf{A} \cong \mathbf{A} \times \mathbf{1}$. In other words, in a Church algebra, factor congruences are internally represented as central elements. Moreover, given $e \in \mathrm{Ce}(\mathbf{A})$, the function f_e defined by $f_e(x, y) = \mathsf{ite}(e, x, y)$ is a decomposition operator such that $f_e(1, 0) = e$. Conversely, given a decomposition operator f, the element $f(1, 0)$ is central.

Proposition 4.3.3. *There is a natural bijective correspondence between central elements and decomposition operators (resp. pairs of complementary factor congruences).*

Therefore a Church algebra $\mathbf{A}$ is directly indecomposable whenever $\mathrm{Ce}(\mathbf{A}) = \{0, 1\}$.

Stone representation theorem for Church algebras. The central elements of a Church algebra constitute a Boolean algebra. Indeed, the partial ordering on the central elements given by:

$$e \leq d \text{ if and only if } \overline{\vartheta}_e \subseteq \overline{\vartheta}_d$$

is a Boolean ordering and the meet, join and complementation operations are internally representable. The elements 0 and 1 are respectively the bottom and top of this ordering.

Theorem 4.3.4 (Manzonetto and Salibra [M23])**.**
Let $\mathbf{A}$ *be a Church algebra. The algebra* $(\mathrm{Ce}(\mathbf{A}), \wedge, \vee, ^-, 0, 1)$ *of central elements of* $\mathbf{A}$ *defined by:*

$$e \wedge d = \mathsf{ite}(e, d, 0), \qquad e \vee d = \mathsf{ite}(e, 1, d), \qquad e^- = \mathsf{ite}(e, 0, 1),$$

is a Boolean algebra isomorphic to the Boolean algebra of factor congruences of $\mathbf{A}$*.*

The Stone representation theorem for Church algebras follows from Theorem 4.3.4 and from theorems by Comer [Com71] and by Vaggione [Vag96].

In the statement of the next theorem we use the following notations. If I is a maximal ideal of the Boolean algebra $\mathrm{Ce}(\mathbf{A})$, then φ_I denotes the congruence on $\mathbf{A}$ defined by: $\varphi_I = \cup_{e \in I} \overline{\vartheta}_e$. Moreover, we denote by $\mathcal{X}$ the Boolean space of maximal ideals of $\mathrm{Ce}(\mathbf{A})$.

Theorem 4.3.5 (Stone Representation Theorem, Manzonetto and Salibra [M23])**.**
Let $\mathbf{A}$ *be a Church algebra. Then, for all* $I \in \mathcal{X}$ *the quotient algebra* $\mathbf{A}/\varphi_I$ *is directly indecomposable and the function* $f : A \to \Pi_{I \in X}(A/\varphi_I)$*, defined by*

$$f(x) = ([x]_{\varphi_I} : I \in \mathcal{X}),$$

gives a weak Boolean product representation of $\mathbf{A}$*.*

When $\mathbf{A}$ is a Boolean algebra, we retrieve the classic Stone representation theorem for Boolean algebras. However, in general, Theorem 4.3.5 does not give a Boolean product representation. This was shown in [M22] in the context of combinatory algebras.

Some Church algebras $\mathbf{A}$ contain a set X such that whenever we consider two disjoint subsets $X_1, X_2 \subseteq X$ the elements of X_1, X_2 can be respectively equated with 0 and 1 without creating any inconsistency. As a matter of notation, given an element $a \in A$ and a set $Y \subseteq A$, we write $\vartheta(a, Y)$ for the least congruence equating a with all the elements in Y.

Definition 4.3.6. *We say that a subset* X *of* A *is an* easy set[1] *if, for every* $Y \subseteq X$, $\vartheta(1, Y) \vee \vartheta(0, X - Y) \neq \nabla$ *(by definition* $\vartheta(1, \emptyset) = \vartheta(0, \emptyset) = \Delta$*).*

We say that an element a is *easy* if $\{a\}$ is an easy set. Thus, a is easy if the congruences ϑ_a and $\overline{\vartheta}_a$ are both different from ∇. In [M23], we proved that in presence of an easy set, the congruence lattice of a Church algebras contains finite Boolean lattices as subintervals.

Theorem 4.3.7 (Manzonetto and Salibra [M23])**.**
Let $\mathbf{A}$ *be a Church algebra and* X *be an easy subset of* A*. Then there exists a congruence* φ'_X *satisfying the following conditions:*

1. *The lattice reduct of the free Boolean algebra with a set* X *of generators can be embedded into the lattice interval* $[\varphi'_X) := \{\varphi \in \mathrm{Con}(\mathbf{A}) \mid \varphi'_X \subseteq \varphi\}$*;*
2. *If* X *has finite cardinality* n*, then the above embedding is an isomorphism and* $[\varphi'_X)$ *has* 2^{2^n} *elements.*

[1] The terminology "easy" is borrowed from λ-calculus easy terms [Bar84, Def. 15.3.8].

4.4 CHURCH ALGEBRAS AT WORK

In this section we present some applications of the notion of Church algebra in the context of λ-calculus to infer properties of its models and theories, and in the context of universal algebra to characterise lattices of equational theories.

Applications to λ-calculus. Church algebras constitute a unifying framework that allows to study both the denotational models of λ-calculus [M22, M25] and the lattice of λ-theories [M23]. As mentioned above, every combinatory algebra $\mathbf{C} = (C, \cdot, \mathbf{k}, \mathbf{s})$ is a Church algebra whose trivial central elements $\mathbf{k}$ and $\mathbf{k}'$, where $\mathbf{k}'$ is the combinator interpreting the λ-term K' in $\mathbf{C}$. The representation theorem for Church algebras, instantiated to combinatory algebras, states that the directly indecomposable combinatory algebras constitute the "building blocks" in the variety of combinatory algebras. It is therefore natural to investigate the *indecomposable semantics* of λ-calculus, that is the class of λ-models that are indecomposable as combinatory algebras. Together with Salibra we have shown that the indecomposable semantics is general enough to encompass all the main semantics.

Theorem 4.4.1 (Manzonetto and Salibra [M22])**.**
The models living in the Scott-continuous, stable and strongly stable semantics are simple combinatory algebras, therefore they all belong to the indecomposable semantics.

A natural problem that arises when studying a class $\mathscr{C}$ of models of λ-calculus is the question whether it is *complete* which means that for every λ-theory $\mathcal{T}$ there exists a λ-model $\mathcal{M} \in \mathscr{C}$ such that $\mathrm{Th}(\mathcal{M}) = \mathcal{T}$. In other words, a class $\mathscr{C}$ is complete whenever all λ-theories arise as theories of some models in the class; otherwise $\mathscr{C}$ it is called *incomplete*. The completeness problem was negatively solved by Honsell and Ronchi Della Rocca for the Scott-continuous semantics [HR92], by Bastonero and Gouy for the stable semantics [BG99] and by Bastonero for the strongly stable semantics [Bas96]. By exploiting the existence of *easy* λ-terms (like Ω), that is λ-terms that can be consistently equated with any $M \in \Lambda^o$, we are able to provide a uniform incompleteness proof for all these semantics. The idea is to consider the λ-theories $\mathcal{T}_1, \mathcal{T}_2$ generated by equating Ω with K and K' (respectively) and prove that Ω is a non-trivial central element in the term model of $\mathcal{T} := \mathcal{T}_1 \cap \mathcal{T}_2$. Since directly indecomposable combinatory algebras are closed under subalgebras, all models of $\mathcal{T}$ must be decomposable, and thus omitted by the indecomposable semantics.

Theorem 4.4.2 (Manzonetto and Salibra [M22])**.**
The Scott-continuous, the stable and the strongly stable semantics are all incomplete.

We recall from Section 1.1 that the set of all λ-theories, ordered by inclusion, forms a complete lattice $\lambda\mathcal{T}$ of cardinality $2^{\aleph_0}$ whose least element of is denoted by $\boldsymbol{\lambda}$. The *term algebra* of a λ-theory $\mathcal{T}$, hereafter denoted by $\boldsymbol{\Lambda}_{\mathcal{T}}$, has the equivalence classes of λ-terms modulo $\mathcal{T}$ as elements, and the operations of application and of λ-abstractions as operations on these classes. It turns out that the lattice $\lambda\mathcal{T}$ of λ-theories is isomorphic to the congruence lattice of the term algebra $\boldsymbol{\Lambda}_{\boldsymbol{\lambda}}$, which is a Church algebra. Moreover, every lattice interval $[\mathcal{T}) = \{\mathcal{T}' \in \lambda\mathcal{T} \mid \mathcal{T} \subseteq \mathcal{T}'\}$ is isomorphic to the congruence lattice of $\boldsymbol{\Lambda}_{\mathcal{T}}$.

From the easiness of Ω and a compactness argument, it follows that the set consisting of all λ-terms $\Omega\mathsf{c}_n$, where c_n is the n-th Church numeral, is an easy set in the term algebra of $\boldsymbol{\lambda}$ (cf. [Bar84, Ex. 15.4.3]). More generally, if a λ-theory $\mathcal{T}$ is *r.e.*, which means that its equivalence classes $[M]_{\mathcal{T}}$ are recursively enumerable, $\boldsymbol{\Lambda}_{\mathcal{T}}$ admits an infinite easy set X.

By Theorem 4.3.7 for every finite subset of X of cardinality k, there is a λ-theory $\mathcal{T}'_k$ containing $\mathcal{T}$ and such that the lattice interval $[\mathcal{T}'_k)$ is isomorphic to the finite Boolean lattice with 2^{2^k} elements. The λ-theory $\mathcal{T}_n$ in the next theorem can be defined from $\mathcal{T}'_k$ using the fact that every filter of a finite Boolean algebra is a Boolean lattice and the fact that the free Boolean algebra with 2^{2^k} elements has filters of arbitrary cardinality 2^n for $n \leq 2^k$.

Theorem 4.4.3 (Manzonetto and Salibra [M23])**.**
For every r.e. λ-theory $\mathcal{T}$ and each natural number n, there exists a λ-theory $\mathcal{T}_n \supseteq \mathcal{T}$ such that the lattice interval $[\mathcal{T}_n)$ is isomorphic to the finite Boolean lattice with 2^n elements.

This is the first time that finite subintervals of $\lambda\mathcal{T}$ of cardinality different from 1 are constructed. In particular, notice that the λ-theory $\mathcal{T}_n$ above cannot be r.e., otherwise the lattice interval $[\mathcal{T})$ would have a continuum of elements by [Bar84, Cor. 17.1.11].

Applications to equational theories. We say that Σ is *an equational theory* if and only if Σ is a set of identities closed under the rules of the equational calculus. It is well known that the set $L(\Sigma) = \{T \mid \Sigma \subseteq T, T \text{ is an equational theory}\}$ forms a lattice under inclusion. We say that a lattice L is a *lattice of equational theories* if and only if L is isomorphic to the lattice $L(\Sigma)$ for some equational theory Σ. In 1966 A.I. Malcev [Mal68] posed the question:

which lattices can be represented as lattices of equational theories?

$L(\Sigma)$ is an algebraic and coatomic lattice, possessing a compact top element; but no stronger property was known before Lampe's discovery [Lam86] that any lattice of equational theories satisfies the *Zipper condition*: if $\vee\{a_i : i \in I\} = 1$ and $a \wedge c = z$ then $c = z$. The proof uses the following representation of the lattices of equational theories, which is due to McKenzie [McK83]: if L is a lattice of equational theories, then L is isomorphic to some congruence lattice of groupoids with right unit and right zero. The problem of whether any congruence lattice of groupoids with right unit and right zero is isomorphic to a lattice of equational theories, is still open (see [Lam86]). The representation of the lattices of equational theories by congruence lattices of monoids with one additional unary operation was found by Newrly [New93]. We briefly describe Newrly's construction.

Given a countable set $X = \{x_i : i \in \mathbb{N}\}$ of variables and an algebraic type τ, we denote by T_X the set of all terms over X with operation symbols from τ and by (T_X, τ) the corresponding term algebra. We write End for its set of endomorphisms.

The lattice $L(\tau)$ of all equational theories of the given type τ can be described as $\mathrm{Con}(\mathsf{T}_X, \tau \cup \mathrm{End})$. Newrly [New93] has shown that $\mathrm{Con}(\mathsf{T}_X, \tau \cup \mathrm{End}) = \mathrm{Con}(\mathsf{T}_X, +, 0, \phi)$, where $(\mathsf{T}_X, +, 0)$ is a monoid and ϕ is unary. The operations are defined by setting $0 := x_0$, $s + t := t\{s/x_0\}$, $\phi(x_i) := x_{i-1}$ and $\phi(x_0) := x_0$. In [M24] we modified Newrly's algebra $(\mathsf{T}_X, +, 0, \phi)$, without changing its congruence lattice, to turn it into a Church algebras. The idea is to consider the algebra $\mathbf{C}(\tau) = (\mathsf{T}_X, \mathsf{ite}, \phi, 0, 1)$, where $0 := x_0$, $1 := x_1$, ϕ is defined as in Newrly's algebra and the ternary operation "ite" is defined as follows: $\mathsf{ite}(t, s, u) := t\{u/x_0, s/x_1\}$. Newrly's algebra is a reduct of $\mathbf{C}(\tau)$ because $s + t = \mathsf{ite}(t, 1, s)$.

In conclusion, we have for the lattice $L(\tau)$ of all equational theories of type τ:

$$L(\tau) = \mathrm{Con}(\mathbf{C}(\tau)).$$

Therefore every lattice of equational theories is isomorphic to the congruence lattice of a Church algebra. These investigations opened the way to develop an approach to algebraic logic based on decomposition operators, that we describe in the rest of the chapter.

4.5 ALGEBRAIZING LOGIC THROUGH FACTOR VARIETIES

Algebraic logic investigates the connections between a logic and algebraic properties of its corresponding class of algebras. The origin of modern algebraic logic goes back to Tarski's 1935 paper [Tar35], where he introduced the Tarski-Lindenbaum algebra as a tool for establishing the correspondence between classical propositional logic and Boolean algebras. In this context the tautologies coincide with those formulas equivalent to the truth value "true". Subsequently, a number of different propositional logics were algebraized in this way, the most important being the intuitionistic logic and the multi-valued logics of Gödel [Göd32], of Post [Pos21] and of Łukasiewicz [Lu20].

The problem of algebraizing predicate logics is much more complicated because of the variable binding properties of the quantifiers. On the one hand, the algebraization of classical predicate logic led Tarski to the definition of cylindric algebras [HMT85] and Halmos to the notion of polyadic Boolean algebras [Hal54]. In practice these algebras are difficult to manipulate because they are endowed with operators representing the quantifiers in the algebraic structure and this complicates their theory. On the other hand, much work in computer science has been focused on reducing first-order logic to equational logic and, more recently, to term rewriting systems. In [McK75] McKenzie proved that for every sentence Φ in first-order classical logic there is an equation Φ' in a suitable algebraic language such that Φ has non-trivial models of a given cardinality κ exactly when Φ' does. In his 1992 paper [Bur92], Burris made a substantial advance by using discriminator varieties [Wer78].

A discriminator variety V is characterized by a quaternary term s that realizes the switching function on any subdirectly irreducible member of V [BS81, Def. 7.3]:

$$\mathsf{s}(a,b,c,d) = \begin{cases} c & \text{if } a = b, \\ d & \text{otherwise.} \end{cases}$$

Thanks to this switching function, Burris has shown that discriminator varieties have unitary unification, which is at the basis of resolution theorem provers and of the Knuth-Bendix method for finding rewriting systems. He was also able to combine McKenzie's analysis of satisfiability with a standard reduction of $\Psi_1, \ldots, \Psi_n \models \Phi$ to a set of unsatisfiable sentences in prenex normal form. Indeed, given a formula Φ and a finite set T of formulas, one can prove that $T \vdash \Phi$ holds by showing $T \models \Phi$ which is, in turn, equivalent to showing that $\Sigma := T \cup \{\neg\Phi\}$ has no models. In [Bur92], Burris shows how to define a set E of equations in the equational logic of a given discriminator variety such that Σ has no models of cardinality greater than 1 exactly when E has no non-trivial models. To show that E has no non-trivial models it is enough to derive the identity $x = y$ from E. This approach is however not applicable to propositional logic and the process of deriving $x = y$ is not easily automatable because of the complexity of the axioms in the system.

A different approach. In [M29], together with Salibra and Favro, we developed a uniform method for extracting the logical content of a formula: in particular, it allows to determine whether a propositional formula is a tautology or a contradiction. In our approach, rather than using the switching function of discriminator varieties, we use the decomposition operators characterizing the *factor varieties*. The definition we provide of factor variety generalizes not only the notion of discriminator variety, but also the one of factor variety as it was introduced in [SLP15]. Our method is general enough to be applied to any finite multi-valued matrix logic. The question whether it can be extended to infinite logics, like fuzzy logic [Hàj98] and probabilistic logic [Nil86], is currently under investigation.

As a motivating example, we consider the case of the classical propositional logic $\mathcal{C}$.

Our approach consists of two steps.

Step 1. The first step consists in defining a translation $(\cdot)^*$ sending propositional formulas into algebraic terms. Under this translation, the truth values f, t become fresh algebraic variables $\xi_\mathsf{f}, \xi_\mathsf{t}$. A propositional variable P becomes a binary operator $P(-,-)$. A propositional formulas ϕ is translated inductively into an algebraic term ϕ^* on the variables $\xi_\mathsf{f}, \xi_\mathsf{t}$. To simplify the notations we write $\phi^*(t_0, t_1)$ for the substitution $\phi^*\{t_0/\xi_\mathsf{f}, t_1/\xi_\mathsf{t}\}$.

$$\begin{array}{rcl}
P^* & = & P(\xi_\mathsf{f}, \xi_\mathsf{t}); \\
(\neg\phi)^* & = & \phi^*(\xi_{\neg\mathsf{f}}, \xi_{\neg\mathsf{t}}) = \phi^*(\xi_\mathsf{t}, \xi_\mathsf{f}); \\
(\phi \wedge \psi)^* & = & \psi^*(\phi^*(\xi_{\mathsf{f}\wedge\mathsf{f}}, \xi_{\mathsf{f}\wedge\mathsf{t}}), \phi^*(\xi_{\mathsf{t}\wedge\mathsf{f}}, \xi_{\mathsf{t}\wedge\mathsf{t}})) = \psi^*(\phi^*(\xi_\mathsf{f}, \xi_\mathsf{f}), \phi^*(\xi_\mathsf{f}, \xi_\mathsf{t})); \\
(\phi \vee \psi)^* & = & \psi^*(\phi^*(\xi_{\mathsf{f}\vee\mathsf{f}}, \xi_{\mathsf{f}\vee\mathsf{t}}), \phi^*(\xi_{\mathsf{t}\vee\mathsf{f}}, \xi_{\mathsf{t}\vee\mathsf{t}})) = \psi^*(\phi^*(\xi_\mathsf{f}, \xi_\mathsf{t}), \phi^*(\xi_\mathsf{t}, \xi_\mathsf{t})); \\
(\phi \to \psi)^* & = & (\neg\phi \vee \psi)^* = \psi^*(\phi^*(\xi_\mathsf{t}, \xi_\mathsf{f}), \phi^*(\xi_\mathsf{t}, \xi_\mathsf{t})).
\end{array}$$

Connectives are therefore implemented through substitutions and Boolean operations on the indices of $\xi_\mathsf{f}, \xi_\mathsf{t}$. The above translation determines a congruence $\sim^*$ on the set of propositional formulas by setting $\phi \sim^* \psi$ if and only if $\phi^* = \psi^*$. This defines a non-commutative intermediate logic $\mathcal{C}_{\mathrm{int}}$ strictly weaker than classical logic.

Step 2. To retrieve classical logic, we need to endow each P with the operational behavior of a binary decomposition operator:

D1 $P(x, x) = x$;

D2 $P(P(x, y), P(w, z)) = P(x, z)$;

D3 $P(Q(x, y), Q(w, z)) = Q(P(x, w), P(y, z))$, for every propositional variable Q.

Both truth values and propositional variables, that are static objects in the logic $\mathcal{C}$, become dynamic entities after the translation: indeed the variables $\xi_\mathsf{f}, \xi_\mathsf{t}$ can receive substitutions and the operators $P(-,-)$ induce decompositions. We show that the propositional formula ϕ is a tautology if and only if $\phi^* = \xi_\mathsf{t}$ is provable using the axioms $(D1)$-$(D3)$ above (Corollary 4.9.2). In Section 4.10, we give this process a computational flavor by showing that, by orienting the equations from left to right and splitting $(D2)$-$(D3)$ appropriately, we obtain a confluent term rewriting system. Moreover, by well-ordering the propositional variables we can prevent $(D3)$ from looping and ensure strong normalization.

This approach also suggests a new notion of circuit, described in Section 4.11, which is based on components that we call "decomposition gates" and behave like the decomposition operators of an algebra belonging to a factor variety.

Algebraization of first-order classical logic. The translation above can be also generalized to first-order formulas by transforming an n-ary relation symbol R into an operator $R(-_1, \ldots, -_{n+2})$ of arity $n + 2$ (since there are two truth values), which is a decomposition operator in the last two coordinates. Open formulas can be therefore inductively translated, as in Step 1, into algebraic terms by setting:

$$R(t_1, \ldots, t_n)^* = R(t_1, \ldots, t_n, \xi_\mathsf{f}, \xi_\mathsf{t}).$$

Such a translation provides a bijective correspondence between first-order theories axiomatized by universal sentences without equality and varieties of factor algebras axiomatized by identities such as $\Phi^* = \xi_\mathsf{t}$. In presence of equality, the situation becomes more subtle. Intuitively, the problem is that factor algebras can only capture correctly *proper* structures, therefore to check whether the formula Φ is actually a logical truth, one also need to verify that its propositional translation is a tautology (see Lemma 4.7.5 below).

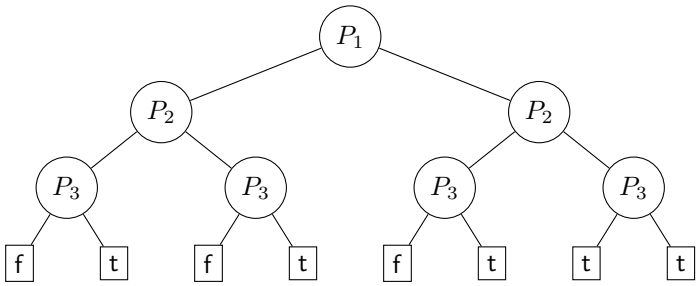


Figure 4.1: The binary decision tree representing $(P_1 \wedge P_2) \vee P_3$.

4.6 A Comparison With Decision Diagrams

While reviewing an earlier version of this manuscript, Goubault-Larrecq pointed out a strong connection between our approach and the theory of binary decision diagrams, introduced by Lee [Lee59] and Akers [Ake78], and refined subsequently by Bryant [Bry86]. We briefly describe these data structures and compare the two methods without the pretence of being exhaustive — as the literature on the subject is very rich, there is certainly more to be said (see, e.g., [Bry92, Weg94]).

Binary Decision Diagrams. Every Boolean function f can be symbolically represented as a binary decision tree by performing its iterated *Shannon expansion* $f(P_1, \ldots, P_n) = P_1 \wedge f(\mathsf{t}, P_2, \ldots, P_n) \vee \neg P_1 \wedge f(\mathsf{f}, P_2, \ldots, P_n)$, like in Figure 4.1. Each nonterminal node is labelled by a propositional variable P and has a left child (corresponding to the assignment $P = 0$) and a right one (corresponding to $P = 1$). The value of f is determined by following a path from the root to a terminal node, which is labelled by a Boolean value. Except for the fact that we label nonterminal vertices by decomposition operators and terminal ones by $\xi_{\mathsf{f}}, \xi_{\mathsf{t}}$, our translation of Step 1 actually computes the binary decision tree of a formula.

A *binary decision diagram* (BDD) is a compact representation of such a tree as a directed acyclic graph, obtained by performing transformation rules to reduce its size: 1) eliminate duplicate terminals; 2) eliminate duplicate non-terminals; 3) eliminate redundant nodes. By imposing an ordering on the propositional variables occurring in the DAG, one ensures its canonicity of the representation. A maximally reduced BDD is called a *Reduced Ordered BDD* (ROBDD). All these ingredients are present in our approach as well, in connection with the term rewriting system of Section 4.10. Indeed, the ordering on the nodes is necessary to ensure strong normalization, and the reduction rules capture the transformation rules of BDD's. From this perspective, Theorem 4.10.5 gives a simple proof of the uniqueness of ROBDD representation. Generalizations of BDD's to multi-valued logics, called *multi-valued decisions graphs*, have also been introduced [SKMB90] and studied [SB96, Sas97, MD02, NS03, KZSS15] and the comparison with our method stands.

Concerning first-order classical logic, the situation is more subtle. On the one side, several techniques to simplify and Skolemize first-order sentences based on BDDs have been proposed [Pos92, PL92, Gou94, GP94, Gou95a]. On the other side, these works do not consider the equality relation and do not try to reduce satisfiability of a sentence to an equational problem. We leave for the future a more precise comparison of the two approaches in this setting.

Łukasiewicz Logic $\mathcal{L}_n$	**Gödel Logic $\mathcal{G}_n$**	**Post Logic $\mathcal{P}_n$**
$\neg a = 1 - a$	$\neg a = \begin{cases} 1 & \text{if } a = 0 \\ 0 & \text{if } a \neq 0 \end{cases}$	$\neg a = \begin{cases} a - \frac{1}{n-1} & \text{if } a \neq 0 \\ 1 & \text{if } a = 0 \end{cases}$
$a \to b = \min(1, 1 - a + b)$	$a \to b = \begin{cases} 1 & \text{if } a \leq b \\ b & \text{if } a > b \end{cases}$	

Figure 4.2: Implication and negation in Łukasiewicz, Gödel and Post Logics.

4.7 MULTI-VALUED MATRIX LOGICS

A matrix logic $\mathcal{L}$ [Got01] is defined by specifying the logical connectives, the set of truth values, among which there is a "designated value" representing the traditional truth value "*verum*", and the truth functions that interpret the logical connectives.

Propositional Matrix Logics. Let us consider an algebraic type τ that represents the set of logical connectives together with their arity.

Definition 4.7.1. *A* logical τ-matrix *is a pair* $(\mathbf{V}, \mathtt{t})$ *where* $\mathbf{V}$ *is a finite* τ*-algebra and* $\mathtt{t} \in V$.

The elements of the universe V are called *truth values* and are denoted by $v_1, \ldots, v_p$, while $\mathtt{t}$ is called the *designated element*. Given a set Pvar of propositional variables, the *propositional formulas* ϕ *of type* τ are defined by induction as follows (for $P \in \mathrm{Pvar}, o \in \tau_n$):

$$\phi, \psi \ ::= \ P \mid o(\phi_1, \ldots, \phi_n)$$

A *truth assignment* is any function $\mathcal{I} : \mathrm{Pvar} \to V$. Given a propositional formula ϕ, its *interpretation in* $\mathbf{V}$ *w.r.t.* $\mathcal{I}$ is the element $[\![\phi]\!]^{\mathcal{I}}$ inductively defined by (for $P \in \mathrm{Pvar}, o \in \tau_n$):

$$[\![P]\!]^{\mathcal{I}} = \mathcal{I}(P), \qquad [\![o(\phi_1, \ldots, \phi_n)]\!]^{\mathcal{I}} = o^{\mathbf{V}}([\![\phi_1]\!]^{\mathcal{I}}, \ldots, [\![\phi_n]\!]^{\mathcal{I}}).$$

A propositional formula ϕ is a *tautology* whenever $[\![\phi]\!]^{\mathcal{I}} = \mathtt{t}$ for all truth assignments $\mathcal{I}$.

Definition 4.7.2. *The* propositional matrix logic $\mathcal{L}$ *induced by a logical* τ*-matrix* $(\mathbf{V}, \mathtt{t})$ *is the logic whose semantics is defined as follows:* $\psi_1, \ldots, \psi_n \models_{\mathcal{L}} \phi$ *if and only if, for every truth assignment* $\mathcal{I}$, $[\![\phi]\!]^{\mathcal{I}} = \mathtt{t}$ *whenever* $[\![\psi_i]\!]^{\mathcal{I}} = \mathtt{t}$ *for all* i.

There are many examples of matrix logics. The most famous is Classical Logic $\mathcal{C}$, corresponding to the algebraic type $\tau = \{\wedge, \vee, \neg, \mathtt{f}, \mathtt{t}\}$ and logical matrix $(\mathbf{2}, \mathtt{t})$, where $\mathbf{2}$ is the two elements Boolean algebra of truth values with $\mathtt{f} < \mathtt{t}$ and $\mathtt{t}$ is the designated element.

Łukasiewicz Logics $\mathcal{L}_n$, Gödel Logics $\mathcal{G}_n$ and Post Logics $\mathcal{P}_n$ are n-valued matrix logics having a totally ordered set $0 < \frac{1}{n-1} < \frac{2}{n-1} < \cdots < \frac{n-2}{n-1} < 1$ of truth values, 1 as designated element, and join and meet defined by $a \vee b = \max\{a, b\}$ and $a \wedge b = \min\{a, b\}$. These logics only differ for the definition of negation and implication (which is not present in Post Logic) that is recalled in Figure 4.2.

The n-valued Gödel logics are *superintuitionistic logics*, which means they are logics between intuitionistic and classical logics. Superintuitionistic logics form a complete lattice whose unique coatom is the 3-valued Gödel Logic $\mathcal{G}_3$. As shown by Gödel in [Göd32], the intuitionistic logic is not definable by a finite logical matrix.

Quantified Matrix Logics. Let us consider fixed a countably infinite set Var of individual variables, an algebraic type τ of logical connectives, a logical τ-matrix $(\mathbf{V}, \mathtt{t})$ and a relational type ν containing both function and relation symbols with arity. We write $f \in \nu$ (resp. $R \in \nu$) to indicate that f is a function symbol (resp. R is a relation symbol) of type ν. Similarly, $f \in \nu_n$ (resp. $R \in \nu_n$) indicates that the symbol f (resp. R) has arity n.

Terms of type ν, or *ν-terms*, are defined as usual from individual variables in Var and function symbols in ν. The set of all ν-terms will be denoted by T_ν and its elements by t, t_1, t_2, etc. *Well formed formulas* are defined by the following grammar, where $R \in \nu_m$ is a relation symbol, $o \in \tau_n$ is a logical connective and $t_1, \ldots, t_m$ are ν-terms:

$$\Phi, \Psi ::= R(t_1, \ldots, t_m) \mid o(\Phi_1, \ldots, \Phi_n) \mid \forall x.\Phi \mid \exists x.\Phi$$

A formula Φ is called: (i) a *sentence* if it has no free variables; (ii) *open* if it is quantifier-free; (iii) *in prenex form* if it has the shape $Q_1 P_1 \ldots Q_n P_n.\Psi$ where $Q_i \in \{\forall, \exists\}$ and Ψ is an open formula (called *the matrix of* Φ); (iv) *universal* if it is in prenex form $Q_1 P_1 \ldots Q_n P_n.\Psi$ and all its quantifiers Q_i are universal.

Definition 4.7.3. *A ν-structure $\mathcal{S}$ on V is given by a collection $(S, g^{\mathcal{S}}, R^{\mathcal{S}})_{g,R\in\nu}$ where S is a set, $g^{\mathcal{S}} : S^k \to S$ is a k-ary operation for any function symbol $g \in \nu_k$ and $R^{\mathcal{S}} : S^n \to V$ is a function for any relation symbol $R \in \nu_n$. We say that the structure $\mathcal{S}$ is* proper *whenever $|S| > 1$.*

The ν-structures having as carrier set a singleton are in some sense *degenerate* models and are handled separately. We denote by $\mathsf{Str}^*_{\nu,V}$ the class of all proper ν-structures on V.

Given a ν-structure $\mathcal{S}$ on V as above, a *valuation* is any function $\rho : \mathrm{Var} \to S$. The interpretation $[\![t]\!]^{\mathcal{S}}_\rho$ of a term t is defined as usual. To interpret the quantifiers we assume the set V of truth values to be a finite lattice, whose top element is the designated element $\mathtt{t}$.

The *interpretation of a formula Φ in $\mathcal{S}$ with respect to a valuation ρ* is then defined inductively as follows (for $R \in \nu_m$, $t_1 \ldots, t_m \in \mathsf{T}_\nu$ and $o \in \tau_n$):

$$[\![R(t_1, \ldots, t_m)]\!]^{\mathcal{S}}_\rho = R^{\mathcal{S}}([\![t_1]\!]^{\mathcal{S}}_\rho, \ldots, [\![t_m]\!]^{\mathcal{S}}_\rho); \quad [\![\forall x.\Phi]\!]^{\mathcal{S}}_\rho = \textstyle\bigwedge_{a\in S} [\![\Phi]\!]^{\mathcal{S}}_{\rho\{a/x\}};$$
$$[\![o(\Phi_1, \ldots, \Phi_n)]\!]^{\mathcal{S}}_\rho = o^{\mathbf{V}}([\![\Phi_1]\!]^{\mathcal{S}}_\rho, \ldots, [\![\Phi_n]\!]^{\mathcal{S}}_\rho); \quad [\![\exists x.\Phi]\!]^{\mathcal{S}}_\rho = \textstyle\bigvee_{a\in S} [\![\Phi]\!]^{\mathcal{S}}_{\rho\{a/x\}}.$$

We write $\mathcal{S} \models_\rho \Phi$ whenever $[\![\Phi]\!]^{\mathcal{S}}_\rho = \mathtt{t}$. We say that a formula Φ is a *logical truth* if $\mathcal{S} \models_\rho \Phi$ for every structure $\mathcal{S}$ and valuation ρ. A class S of ν-structures is called *universal* if it can be axiomatized by universal formulas.

Definition 4.7.4. *The* quantified matrix logic *$\mathcal{QL}$, induced by a logical τ-matrix $(\mathbf{V}, \mathtt{t})$ and a relational type ν, is the logic whose semantics is defined as: $\Psi_1, \ldots, \Psi_n \models_{\mathcal{QL}} \Phi$ if and only if, for every structure $\mathcal{S}$ and valuation ρ, $[\![\Phi]\!]^{\mathcal{S}}_\rho = \mathtt{t}$ whenever $[\![\Psi_k]\!]^{\mathcal{S}}_\rho = \mathtt{t}$ for all k.*

In order to check whether a formula Φ is true in all singleton structures it is enough to consider its *propositional translation* Φ^{p} which is the propositional formula defined by:

- $R(t_1, \ldots, t_m)^{\mathrm{p}} = P_R$, where $P_R \in \mathrm{Pvar}$;
- $o(\Phi_1, \ldots, \Phi_n)^{\mathrm{p}} = o(\Phi_1^{\mathrm{p}}, \ldots, \Phi_n^{\mathrm{p}})$;
- $(\forall x.\Phi)^{\mathrm{p}} = (\exists x.\Phi)^{\mathrm{p}} = \Phi^{\mathrm{p}}$.

In classical logic with equality, there exists an equality symbol which is propositionally translated by setting $(t_1 = t_2)^{\mathrm{p}} = \mathtt{t}$.

Lemma 4.7.5. *A formula Φ is true in all singleton structures if and only if its propositional translation Φ^{p} is a tautology.*

4.8 FACTOR ALGEBRAS AND FACTOR VARIETIES

We present the notions of factor algebras and factor varieties as introduced in [M29].

Factor algebras. We consider fixed a relational type ν and a logical τ-matrix $(\mathbf{V}, \mathsf{t})$ where $V = \{v_1, \ldots, v_p\}$. We denote by $\hat{\nu}$ the smallest algebraic type containing:

- a function symbol $g \in \hat{\nu}_k$ for each function symbol $g \in \nu_k$,
- a function symbol $f_R \in \hat{\nu}_{n+p}$ for each relation symbol $R \in \nu_n$,

i.e. a relation R of arity n is transformed into a function f_R having p additional arguments.

Definition 4.8.1. *A $\hat{\nu}$-factor algebra $\mathbf{A} = (A, g^{\mathbf{A}}, f_R^{\mathbf{A}})_{g,R\in\nu}$ is a $\hat{\nu}$-algebra such that, for all $f_R \in \hat{\nu}_{n+p}$ and $\vec{a} \in A^n$ there exists an index $i \in [1..p]$ such that:*

$$\forall \xi_1 \ldots \xi_p . f_R(\vec{a}, \xi_1, \ldots, \xi_p) = \xi_i. \tag{4.1}$$

The class $\mathsf{FA}_{\hat{\nu}}$ of all $\hat{\nu}$-factor algebras is a universal class, which means that it is closed under subalgebras and ultraproducts. We write $\mathsf{FA}^*_{\hat{\nu}}$ for the class of *proper* factor algebras.

Given a $\hat{\nu}$-factor algebra $\mathbf{A}$, the *algebraic reduct* of $\mathbf{A}$ is the algebra $\mathrm{Alg}(\mathbf{A}) = (A, g^{\mathbf{A}})_{g\in\nu}$.

Definition 4.8.2. *We associate with every proper factor algebra $\mathbf{A}$ a proper structure $\mathrm{Str}(\mathbf{A})$ having the same algebraic reduct, and relations defined by (for all $f_R \in \hat{\nu}_{n+p}$ and $\vec{a} \in A^n$):*

$$R^{\mathrm{Str}(\mathbf{A})}(\vec{a}) = v_k \text{ if and only if } \forall \xi_1, \ldots, \xi_p . f_R^{\mathbf{A}}(\vec{a}, \xi_1, \ldots, \xi_p) = \xi_k.$$

Conversely, we associate with every proper structure $\mathcal{S}$ a proper factor algebra $\mathrm{Fa}(\mathcal{S})$ having the same algebraic reduct as $\mathcal{S}$ and whose functions f_R ($R \in \nu_n$) are defined as follows:

$$f_R^{\mathrm{Fa}(\mathcal{S})}(\vec{a}, \xi_1, \ldots, \xi_p) = \xi_k \text{ if and only if } R^{\mathcal{S}}(\vec{a}) = v_k.$$

In particular, we have $\mathrm{Str}(\mathrm{Fa}(\mathcal{S})) = \mathcal{S}$ and $\mathrm{Fa}(\mathrm{Str}(\mathbf{A})) = \mathbf{A}$.

Note that the above correspondence fails on singleton structures. Let $\mathcal{S}_1, \mathcal{S}_2$ be two structures over $\{*\}$ with a relation symbol R such that $R^{\mathcal{S}_1}(*) = \mathsf{t}$ but $R^{\mathcal{S}_2}(*) \neq \mathsf{t}$. The structures $\mathcal{S}_1$ and $\mathcal{S}_2$ are not isomorphic, but correspond to the same trivial factor algebra.

Factor Varieties are a generalization of a discriminator variety where the set $\{\mathsf{t}, \mathsf{f}\}$ of classical truth values is substituted by the set V and the role of the equality in the definition of the switching term s (page 76) is played by a generic (multi-valued) relation $R : A^n \to V$.

By definition, a *factor variety* is a variety V generated by a class of $\hat{\nu}$-factor algebras.

Proposition 4.8.3. *The variety $\mathsf{V}_{\hat{\nu}}$ generated by the class of all $\hat{\nu}$-factor algebras is axiomatized by (for $f_R \in \hat{\nu}_{n+p}$):*

F1 $f_R(\vec{x}, \xi, \ldots, \xi) = \xi$;

F2 $f_R(\vec{x}, f_R(\vec{x}, \xi_{11}, \ldots, \xi_{1p}), \ldots, f_R(\vec{x}, \xi_{p1}, \ldots, \xi_{pp})) = f_R(\vec{x}, \xi_{11}, \ldots, \xi_{pp})$;

F3 $f_R(\vec{x}, h(\xi_{11}, \ldots, \xi_{1k}), \ldots, h(\xi_{p1}, \ldots, \xi_{pk})) = h(f_R(\vec{x}, \xi_{11}, \ldots, \xi_{p1}), \ldots, f_R(\vec{x}, \xi_{1k}, \ldots, \xi_{pk}))$, *where $h \in \hat{\nu}_k$ is an arbitrary element of $\hat{\nu}$.*

Let us consider an arbitrary algebra $\mathbf{A}$ belonging to $\mathsf{V}_{\hat{\nu}}$ (not necessarily a factor algebra). For every $f_R \in \hat{\nu}_{n+p}$ and $\vec{a} \in A^n$, the p-ary map $f_R(\vec{a}, -, \ldots, -)$ is a decomposition operator on $\mathbf{A}$. By $(F3)$, the decomposition operators f_R ($R \in \nu$) are closed under composition.

Given a factor variety V, we denote by V_{fa} the class of $\hat{\nu}$-factor algebras belonging to V. From Definition 4.8.1 and by [BS81, Ch. 5, Thm. 2.20] it follows that the class V_{fa} is a universal class and every directly indecomposable algebra $\mathbf{A} \in \mathsf{V}$ is a factor algebra.

4.9 ALGEBRAIZATION OF MULTI-VALUED LOGICS

In this section we present the approach introduced in collaboration with Salibra and Favro in the pioneering paper [M29] for the algebraization of quantified matrix logics. We will recover the case of propositional multi-valued matrix logics as a simple instance.

Let us consider fixed a relational type ν and a logical τ-matrix $(\mathbf{V}, \mathtt{t})$, where $V = \{v_1, \ldots, v_p\}$. As announced in Section 4.5, we need to define a translation $(\cdot)^*$ from open ν-formulas into suitable terms of type $\hat{\nu}$, that we call *logical terms*.

Logical terms. First, let us fix a set $\Xi = \{\xi_1, \ldots, \xi_p\}$ of fresh algebraic variables (one for each truth value), called *logical variables*. Recall that T_ν stands for the set of all ν-terms (denoted by t, t_i) over the set Var. The set $\mathsf{LT}_{\hat{\nu}}$ of *logical terms* of type $\hat{\nu}$ (denoted by s, u) is generated by the following grammar (for $\xi_i \in \Xi$, $f_R \in \hat{\nu}$ and $t_1, \ldots, t_n \in \mathsf{T}_\nu$):

$$s, u ::= \ \xi_i \mid f_R(t_1, \ldots, t_n, u_1, \ldots, u_p)$$

Note that $\mathsf{LT}_{\hat{\nu}} \not\subseteq \mathsf{T}_\nu$ since $\nu \neq \hat{\nu}$ and neither the logical variables ξ_i nor the function symbols f_R can occur in t. Let $s, u_1, ..., u_p \in \mathsf{LT}_{\hat{\nu}}$, we write $s\{u_1/\xi_1, \ldots, u_p/\xi_p\}$ for the logical term obtained by substituting simultaneously u_i for each occurrence of ξ_i in s.

From open formulas to logical terms through substitutions. The translation $(\cdot)^*$ we have presented in Section 4.5 for classical logic, can be easily generalized to an arbitrary p-valued matrix logic $\mathcal{L}$. Since the result of the translation can be very verbose, we first introduce some clever notation based on (hyper)matrices.

The intuitive idea is that the truth table of an n-ary logical connective $o \in \tau_n$ will be represented as a hypermatrix M of dimension $p \times \cdots \times p$ (n times).

We recall that hypermatrices are k-dimensional generalizations of the usual bidimensional matrices. Here, we consider hypermatrices of dimension $n_1 \times \cdots \times n_k$ over the set $\mathsf{LT}_{\hat{\nu}}$ of logical terms, that is to say functions

$$M : n_1 \times \cdots \times n_k \to \mathsf{LT}_{\hat{\nu}}.$$

Given a hypermatrix M as above, we write $M_{i_1 \ldots i_k}$ for the logical term $M(i_1, \ldots, i_k)$. A hypermatrix M of dimension p^k is called *cubical*. A *vector* v is any hypermatrix of dimension $p \times 1$ (resp. $1 \times p$) and its *transpose* is a vector of dimension $1 \times p$ (resp. $p \times 1$) denoted by v^T. Given a logical term s, we write $\mathsf{v}(s)$ for the constant vector $[s, \ldots, s]^T$ of dimension $p \times 1$.

Let M be a cubical hypermatrix of dimension p^k such that $M_{i_1 \ldots i_k} \in \mathsf{LT}_{\hat{\nu}}$ and let s be a logical term possibly containing $\xi_1, \ldots, \xi_p$ as variables. The matrix multiplication $M\mathsf{v}(s)$ is a hypermatrix of dimension p^{k-1} defined as follows:

$$(M\mathsf{v}(s))_{i_1 \ldots i_{k-1}} = s\{M_{i_1 \ldots i_{k-1},1}/\xi_1, \ldots, M_{i_1 \ldots i_{k-1},p}/\xi_p\}.$$

As an example, the product between a $p \times p$-matrix and $\mathsf{v}(s)$ is:

$$\begin{bmatrix} u_{11} & \cdots & u_{1p} \\ \vdots & \ddots & \vdots \\ u_{p1} & \cdots & u_{pp} \end{bmatrix} \begin{bmatrix} s \\ \vdots \\ s \end{bmatrix} = \begin{bmatrix} s\{u_{11}/\xi_1, \ldots, u_{1p}/\xi_p\} \\ \vdots \\ s\{u_{p1}/\xi_1, \ldots, u_{pp}/\xi_p\} \end{bmatrix}$$

Hereafter, we assume that matrix multiplication associates to the left. Therefore, we simply write $M\mathsf{v}_1 \cdots \mathsf{v}_k$ for $((\cdots(M\mathsf{v}_1)\cdots)\mathsf{v}_k)$.

The translation. We translate inductively an open formula Φ of a quantified matrix logic $\mathcal{L}$ into a logical term Φ^* as follows:

- $v_i^* = \xi_i$;
- $R(\vec{t})^* = f_R(\vec{t}, \xi_1, \ldots, \xi_p)$;
- $o(\Psi_1, \ldots, \Psi_n)^* = \big(M^o \mathsf{v}(\Psi_1^*) \cdots \mathsf{v}(\Psi_{n-1}^*)\big)^T \mathsf{v}(\Psi_n^*)$,

where M^o is the cubical hypermatrix of dimension p^n defined by:

$$M^o_{i_1 i_2 \ldots i_n} = \xi_k \iff o^{\mathbf{V}}(v_{i_n}, \ldots, v_{i_2}, v_{i_1}) = v_k.$$

In particular, the translation of $P \in \mathrm{Pvar}$ is simply $P^* = f_P(\xi_1, \ldots, \xi_p)$.

Note that, in the definition above, M^o has dimension p^n and each $\mathsf{v}(\psi_i^*)$ has dimension $p \times 1$. So $M^o \mathsf{v}(\Psi_1^*) \cdots \mathsf{v}(\Psi_{n-1}^*)$ is a $p \times 1$-matrix and its transposed a $1 \times p$-matrix $[u_1, \ldots, u_p]$. By multiplying it by $\mathsf{v}(\Psi_n^*)$ we get a 1×1-matrix, that is a term.

Moreover, we have the following substitution property:

$$o(\Psi_1, \ldots, \Psi_n)^* = \Psi_n^*\{u_1/\xi_1, \ldots, u_p/\xi_p\}. \tag{4.2}$$

It is easy to check by a straightforward induction on the open formula Φ that its translation Φ^* is actually a logical term.

Note that, in the propositional case, such a translation induces a congruence $\sim^*$ on the set of formulas: two formulas ϕ and ψ are $\sim^*$-equivalent whenever they have the same translation $\phi^* = \psi^*$. Interestingly enough, this defines a non-commutative intermediate logic $\mathcal{L}'$ which is strictly weaker than the logic $\mathcal{L}$ we started from. For instance, in the case of classical logic $\mathcal{C}$, we have $\neg\neg\phi \sim^* \phi$ and $(\phi_1 \vee \phi_2) \vee \phi_3 \sim^* \phi_1 \vee (\phi_2 \vee \phi_3)$, but $\phi_1 \vee \phi_2 \not\sim^* \phi_2 \vee \phi_1$. This intermediate logic $\mathcal{C}_{\mathrm{int}}$ is strictly weaker than classical logic because, for example, we have $(\neg P \vee P)^* = P(P(\xi_{\mathsf{t}}, \xi_{\mathsf{f}}), P(\xi_{\mathsf{t}}, \xi_{\mathsf{t}}))$ and $(P \vee \neg P)^* = P(P(\xi_{\mathsf{t}}, \xi_{\mathsf{t}}), P(\xi_{\mathsf{f}}, \xi_{\mathsf{t}}))$, hence $\neg P \vee P \not\sim^* P \vee \neg P$.

Problem 15. *What is the precise relationship between a logic $\mathcal{L}$ and the non-commutative intermediate logic $\mathcal{L}'$ induced by the translation in [M29]?*

Theorem 4.9.1. *Let $\mathcal{S}$ be a proper structure and $\rho : \mathrm{Var} \to S$ be a valuation. Then $[\![\Phi]\!]^{\mathcal{S}}_{\rho} = v_k$ if and only if $\mathrm{Fa}(\mathcal{S}) \models_{\rho} \forall \xi_1 \ldots \xi_p . \Phi^* = \xi_k$.*

Recall that Φ^{p} denotes the propositional translation of Φ defined at the end of Section 4.7. From Theorem 4.9.1 and Lemma 4.7.5, we obtain this corollary.

Corollary 4.9.2. *A universal ν-sentence Φ is a logical truth if and only if $\mathsf{V}_{\hat{\nu}} \models \forall \xi_1 \ldots \xi_p . \Phi^* = \xi_{\mathsf{t}}$ and Φ^{p} is a tautology.*

When the logic under consideration is without equality, a sentence Φ fails in a singleton structure if and only if it fails in some proper structure. Therefore, in this case it is possible to omit "and Φ^{p} is a tautology" in the statement of Corollary 4.9.2.

The algebraization of propositional logics. Propositional logic is a particular instance of quantified logic. Indeed, the set Pvar of propositional variables can be considered as a relational type, where every $P \in \mathrm{Pvar}$ is a relation symbol of arity 0.

According to Definition 4.7.3, a structure $\mathcal{S}$ of type Pvar, hereafter called a *propositional structure*, is a pair $(S, P^{\mathcal{S}})_{P \in \mathrm{Pvar}}$ such that $P^{\mathcal{S}} \in V$ for every $P \in \mathrm{Pvar}$. The propositional

structure $\mathcal{S}$ determines the truth assignment $\mathcal{I}_\mathcal{S} : \mathrm{Pvar} \to V$ defined by $\mathcal{I}_\mathcal{S}(P) = P^\mathcal{S}$. Conversely, every truth assignment $\mathcal{I} : \mathrm{Pvar} \to V$ determines, for each set S, a propositional structure $\mathcal{S}_\mathcal{I} = (S, P^{\mathcal{S}_\mathcal{I}})_{P \in \mathrm{Pvar}}$ where $P^{\mathcal{S}_\mathcal{I}} = \mathcal{I}(P)$.

The interpretation of a propositional formula ϕ in a propositional structure $\mathcal{S}$ coincides with its propositional interpretation with respect to the truth assignment $\mathcal{I}_\mathcal{S}$: in other words, $[\![\phi]\!]^\mathcal{S}_\rho = [\![\phi]\!]^{\mathcal{I}_\mathcal{S}}$ for every valuation $\rho : \mathrm{Var} \to S$.

We call p-factor algebra every factor algebra associated with a propositional structure according to Definition 4.8.2. The congruence lattice $\mathrm{Con}(\mathbf{A})$ of a p-factor algebra $\mathbf{A}$ coincides with the lattice of equivalence relations on A. As a consequence, a p-factor algebra $\mathbf{A}$ is directly indecomposable exactly when $\mathbf{A}$ is finite of prime cardinality.

We denote by $\mathsf{V}_{\mathrm{prop}}$ the factor variety generated by all p-factor algebras.

Corollary 4.9.3 (Salibra et Al. [M29])**.**
Let Pvar *be the type of propositional variables. A propositional formula ϕ is a tautology if and only if $\mathsf{V}_{\mathrm{prop}} \models \forall \xi_1 \dots \xi_p . \phi^* = \xi_\mathsf{t}$.*

We now apply our translation to propositional formulas of the logics presented in Section 4.7. To simplify the notations we confuse P with f_P, and i with ξ_i. We also perform some on-the-flight application of (F1) and directly write u rather than $s\{u/\xi_1, \dots, u/\xi_p\}$.

Example 4.9.4. (3-valued Logics with $0 < \frac{1}{2} < 1$) *The translation of some basic formulas:*

- $\mathcal{L}_3\mathcal{G}_3\mathcal{P}_3$: $(P \wedge Q)^* = Q(0, P(0, \frac{1}{2}, \frac{1}{2}), P(0, \frac{1}{2}, 1))$
- $\mathcal{L}_3\mathcal{G}_3\mathcal{P}_3$: $(P \vee Q)^* = Q(P(0, \frac{1}{2}, 1), P(\frac{1}{2}, \frac{1}{2}, 1), 1)$
- $\mathcal{L}_3$: $(\neg P)^* = P(1, \frac{1}{2}, 0)$
- $\mathcal{G}_3$: $(\neg P)^* = P(1, 0, 0)$
- $\mathcal{P}_3$: $(\neg P)^* = P(1, 0, \frac{1}{2})$
- $\mathcal{L}_3$: $(P \to Q)^* = Q(P(1, \frac{1}{2}, 0), P(1, 1, \frac{1}{2}), 1)$
- $\mathcal{G}_3$: $(P \to Q)^* = Q(P(1, 0, 0), P(1, 1, \frac{1}{2}), 1)$.

Example 4.9.5. *The translation of $P \vee \neg P$ in three-valued logics:*

- $\mathcal{L}_3$: $P(1, P(\frac{1}{2}, \frac{1}{2}, 1), P(0, \frac{1}{2}, 1))$
- $\mathcal{G}_3$: $P(1, P(0, \frac{1}{2}, 1), P(0, \frac{1}{2}, 1))$
- $\mathcal{P}_3$: $P(1, P(0, \frac{1}{2}, 1), P(\frac{1}{2}, \frac{1}{2}, 1))$.

Example 4.9.6. *The Peirce law $((P \to Q) \to P) \to P$ translated in classical logic and in some three-valued logics:*

- $\mathcal{C}$: $P(P(Q(P(\mathsf{t}, \mathsf{f}), \mathsf{t}), \mathsf{f}), \mathsf{t})$
- $\mathcal{L}_3$: $P(P(\alpha_1, \alpha_2, 0), P(\beta_1, \beta_2, \frac{1}{2}), 1)$ *where*
  - $\alpha_1 = Q(P(1, \frac{1}{2}, 0), P(1, 1, \frac{1}{2}), 1)$
  - $\alpha_2 = Q(P(\frac{1}{2}, 0, 0), P(\frac{1}{2}, \frac{1}{2}, 0), \frac{1}{2})$
  - $\beta_1 = Q(P(1, 1, \frac{1}{2}), 1, 1)$
  - $\beta_2 = Q(P(1, \frac{1}{2}, \frac{1}{2}), P(1, 1, \frac{1}{2}), 1)$
- $\mathcal{G}_3$: $P(P(\gamma_1, 0, 0), P(\delta_1, \delta_2, \frac{1}{2}), 1)$ *where*
  - $\gamma_1 = Q(P(1, 0, 0), 1, 1)$
  - $\delta_1 = Q(P(1, \frac{1}{2}, \frac{1}{2}), 1, 1)$
  - $\delta_2 = Q(P(1, \frac{1}{2}, \frac{1}{2}), P(1, 1, \frac{1}{2}), 1)$.

4.10 TERM REWRITING SYSTEM FOR FACTOR AXIOMS

We now show how to turn the equations (F1)-(F3) axiomatizing the factor variety $\mathsf{V}_{\hat{\nu}}$ into rewriting rules. The term rewriting system that we obtain enjoys confluence and strong normalization. Therefore, in order to check whether $\mathsf{V}_{\hat{\nu}} \models \Phi^* = \xi_k$ holds it is enough to see whether the normal form of Φ^* is ξ_k.

For the sake of simplicity, we consider a propositional matrix logic $\mathcal{L}$ with two truth values t, f (therefore $\Xi = \{\xi_\mathsf{t}, \xi_\mathsf{f}\}$). All definitions and results extend easily to all p-valued propositional matrix logics. This method is generalizable to arbitrary quantified matrix logics, but the actual generalization is left for future works.

We then consider a relational type ν only containing (countably many) propositional variables. Let us fix an enumeration $(P_i)_{i\in\mathbb{N}}$ of all the propositional variables in ν. Intuitively, this associates a priority $i \in \mathbb{N}$ with each propositional variable.

To simplify the notation, we will still denote by P_i the binary operator $f_{P_i} \in \hat{\nu}$.

Definition 4.10.1. *The rewriting rules $\mathcal{R}$ on $\mathsf{LT}_{\hat{\nu}}$ are (for $i \in \mathbb{N}$):*

(F_1) $P_i(x,x) \rightarrowtail x$;

(F_2^ℓ) $P_i(P_i(x,y),z) \rightarrowtail P_i(x,z)$;

(F_2^r) $P_i(x,P_i(y,z)) \rightarrowtail P_i(x,z)$;

(F_3) $P_i(P_j(x,y),P_j(w,z)) \rightarrowtail P_j(P_i(x,w),P_i(y,z))$;

(F_3^ℓ) $P_i(P_j(x,y),z) \rightarrowtail P_j(P_i(x,z),P_i(y,z))$;

(F_3^r) $P_i(x,P_j(y,z)) \rightarrowtail P_j(P_i(x,y),P_i(x,z))$;

where the rules (F_3), (F_3^ℓ) and (F_3^r) only apply when $i > j$.

The term rewriting system $\mathcal{R}$ is rather standard, except for the fact that it has infinitely many function symbols, a property that we need to handle carefully when proving termination. Note that equation $(F2)$ of Proposition 4.8.3 is recovered in two steps:

$$P_i(P_i(x,y),P_i(w,z)) \rightarrowtail_{F_2^\ell} P_i(x,P_i(w,z)) \rightarrowtail_{F_2^r} P_i(x,z).$$

Analogously, $(F3)$ corresponds to (F_3^ℓ) and (F_3^r), but we keep the redundant rule (F_3) to avoid an unnecessary growth of the size of the terms during the reduction.

We show that $\mathcal{R}$ is locally confluent and terminating , so we conclude that it is confluent by Newman's lemma [New42]. To prove the local confluence it is enough to check that all critical pairs are convergent.

Proposition 4.10.2. *The term rewriting system $\mathcal{R}$ is locally confluent.*

The fact that $\mathcal{R}$ is terminating is non-trivial because the duplication in the rules (F_3^*) may increase substantially the size of the term. Thanks to the condition "$i > j$" these rules push the symbols with small indices towards the root and those with big indices toward the leaves. Thus, two terms should be compared by first comparing their root symbols, and then recursively comparing their immediate subterms.

In other words, we need a lexicographic path order (lpo). We recall that lexicographic path orders were first described by Kamin and Lévy in [KL80] (see also [BKdV03, §6.4]).

Definition 4.10.3. *The* lexicographic path order $>_{\mathrm{lpo}}$ *on terms is defined by:* $s >_{\mathrm{lpo}} u$ *if*
(LPO_1**)** $u \in \mathrm{Var} \cup \Xi$, u *occurs in* s *and* $s \neq u$, *or*
(LPO_2**)** $s = P_i(s_1, s_2)$, $u = P_j(u_1, u_2)$ *and one of the following conditions holds:*
(a) $\exists k \in [1,2],\ s_k \geq_{\mathrm{lpo}} u$,
(b) $i > j$, *and* $\forall k \in [1,2], s >_{\mathrm{lpo}} u_k$,
(c) $i = j$, $(s_1, s_2) >^{\mathrm{lex}}_{\mathrm{lpo}} (u_1, u_2)$ *and* $\forall k \in [1,2], s >_{\mathrm{lpo}} u_k$, *where* $>^{\mathrm{lex}}_{\mathrm{lpo}}$ *stands for the lexicographic* lpo*-order on pairs.*

By [BKdV03, Prop. 6.4.25], the relation $>_{\mathrm{lpo}}$ is a *simplification order*, which means that it is an order closed under contexts, under substitutions, and possesses the subterm property.

The term rewriting system $\mathcal{R}$ satisfies the following property:

Lemma 4.10.4. *For all rewriting rules* $s \rightarrowtail u$ *of* $\mathcal{R}$ *we have* $s >_{\mathrm{lpo}} u$.

This property amounts to saying that the term rewriting system is *simplifying*, that is compatible with a simplification order. In the case of finite term rewriting system, this is enough to conclude termination. As shown in [Ohl92], for infinite term rewriting system one also need to check that the rules only introduce finitely many function symbols.

Theorem 4.10.5. *The term rewriting system* $\mathcal{R}$ *is confluent and terminating.*

We denote by $\mathrm{nf}(u)$ the (unique) normal form of u with respect to $\mathcal{R}$.

Corollary 4.10.6. *A propositional formula* φ *is a tautology if and only if* $\mathrm{nf}(\varphi^*) = \xi_{\mathsf{t}}$.

As an example, we apply the term rewriting system to show that the law of Peirce $((P \to Q) \to P) \to P$ holds in classical logic $\mathcal{C}$, but not in Gödel's logic $\mathcal{G}_3$. We recall that both translations are given in Example 4.9.6. Without loss of generality, we assume that the priority of P is smaller than the priority of Q. As in Example 4.9.6, we will just write i for ξ_i. In Classical Logic $\mathcal{C}$ we have the following reduction:

$$P(P(Q(P(\mathsf{t},\mathsf{f}),\mathsf{t}),\mathsf{f}),\mathsf{t}) \rightarrowtail_{F_2^\ell} P(Q(P(\mathsf{t},\mathsf{f}),\mathsf{t}),\mathsf{t}) \rightarrowtail_{F_3^\ell} P(P(Q(\mathsf{t},\mathsf{t}),Q(\mathsf{f},\mathsf{t})),\mathsf{t}) \rightarrowtail_{F_2^\ell} P(Q(\mathsf{t},\mathsf{t}),\mathsf{t}) \rightarrowtail_{F_1} P(\mathsf{t},\mathsf{t}) \rightarrowtail_{F_1} \mathsf{t}.$$

Since t is designated, the formula is a classical tautology. To compute the reduction in $\mathcal{G}_3$, we will use the notations γ_1, δ_1, δ_2 introduced in Example 4.9.6, and the following facts:

1. $\gamma_1 \overset{F_3}{\to} \gamma_1'$, for $\gamma_1' = P(Q(1,1,1), Q(0,1,1), Q(0,1,1))$;
2. $\delta_2 \overset{F_3}{\to} \delta_2'$, for $\delta_2' = P(Q(1,1,1), Q(\frac{1}{2},1,1), Q(\frac{1}{2},\frac{1}{2},1))$.

Therefore, in $\mathcal{G}_3$ we have the following reduction:

$$P(P(\gamma_1,0,0),P(\delta_1,\delta_2,\tfrac{1}{2}),1) \rightarrowtail_{F_2} P(\gamma_1,\delta_2,1) \rightarrowtail_{F_3} P(\gamma_1',\delta_2,1) \rightarrowtail_{F_3} P(\gamma_1',\delta_2',1) \rightarrowtail_{F_2} P(Q(1,1,1),\delta_2',1) \rightarrowtail_{F_2} P(Q(1,1,1),Q(\tfrac{1}{2},1,1),1) \rightarrowtail_{F_1} P(1,Q(\tfrac{1}{2},1,1),1)$$

Since $P(1, Q(\frac{1}{2},1,1), 1)$ is in normal form, we conclude that Peirce law is neither a tautology nor a contradiction in $\mathcal{G}_3$.

We end this section by remarking that the logical terms that appear during the reduction are not necessarily the translation of a logical formula. Henceforth, this process of calculus cannot be simulated within the logic under consideration.

Problem 16. *Consider the term rewriting system* $\mathcal{R}$ *defined in [M29]. What is the complexity of the system? Is the optimal reduction strategy effective? What is its complexity?*

4.11 FACTOR CIRCUITS AND APPLICATIONS TO HARDWARE DESIGN

Classical propositional logic is used as a technical tool for the analysis and the synthesis of electrical circuits built up from *switches* with two stable states: the voltage levels. Analogously, p-valued logics correspond to circuits built from similar switches with p stable states, each representing a different truth value. This whole field of application of logic is called many-valued (or fuzzy) switching. We refer the reader to [Eps93] for a good introduction on this subject.

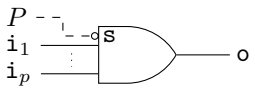
Our algebraic approach to multi-valued logics suggests a new notion of circuit, based on components that we call "decomposition gates" and behave as decomposition operators of an algebra $\mathbf{A}$ belonging to a factor variety $\mathsf{V}_{\dot{p}}$. In this section we consider $\mathbf{A}$ fixed.

We start by presenting the p-valued propositional case, then we instantiate it to propositional classical logic and compare it with the usual Boolean circuits, finally we discuss the most general case.

A propositional *decomposition gate* (*D-gate*, for short) has:

- p input ports $\mathtt{i}_1, \ldots, \mathtt{i}_p$ (one for each truth value);
- a switch $\mathtt{s}$, called the *selector switch*;
- an output port $\mathtt{o}$.

The graphical representation of a D-gate is the following:



The selector switch has a particular status since it specifies which decomposition operator f_P is implemented by the gate. For instance, when $\mathbf{A}$ is a factor algebra then f_P is a projection π^p_k (that is a trivial decomposition operator) and the selector switch transforms the D-gate into a multiplexer selecting its k-th input (thus $\mathtt{o} := \mathtt{i}_k$).

D-gates can be composed using *wires* by connecting the output port $\mathtt{o}$ of a D-gate with one (or more) input port(s) $\mathtt{i}_k$ of other D-gates. Therefore the wires transport the values of the algebraic variables $\xi_1, \ldots, \xi_p$, in other words elements of A.

The circuit obtained by composing several D-gates is called *factor circuit*. Since each D-gate implements a decomposition operator of the algebra $\mathbf{A}$ and by (F3) decomposition operators of $\mathbf{A}$ commute, by [MMT87, Ex. 4.38.15, p. 167] a factor circuit represents itself a decomposition operator on $\mathbf{A}$.

Every logical term u can be easily represented as a factor circuit by following its syntactic tree and drawing a D-gate with selector switch P_i for each function symbol f_{P_i}. A formula ϕ is then transformed into the factor circuit corresponding to the term ϕ^*.

D-gates for propositional classical logic $\mathcal{C}$ are shown in Figure 4.4(a): to simplify the picture, we omit the selector switch and directly label the gate with the propositional variable P_i, where i represents the priority of P (as in Section 4.10). A quick comparison between the usual Boolean circuits and factor circuits shows the novelty of this approach (cf. Figure 4.3). In the Boolean circuits, each logical gate implements a logical connective $o \in \tau$ of arity n, so it has n input ports $\mathtt{i}_1, \ldots, \mathtt{i}_n$, and its output is obtained by applying such a connective to the inputs: $o(\mathtt{i}_1, \ldots, \mathtt{i}_n)$. The logical gates are connected with each others through wires that transport truth values. The remaining input wires are connected

	logic gate AND	**D-gate**
operation	connective $\wedge$	decomposition operator f_P
meaning	static (AND)	dynamic (depends on P)
no. inputs	arity of $\wedge$	$\lvert V\rvert$
input values	propositional variables P, Q	algebraic variables $\xi_{\mathtt{f}}, \xi_{\mathtt{t}}$
signals carried by the wires	truth values	elements of A
output	$P \wedge Q$	$f_P(\xi_{\mathtt{f}}, \xi_{\mathtt{t}})$

Figure 4.3: Comparison between a logic gate AND and a D-gate.

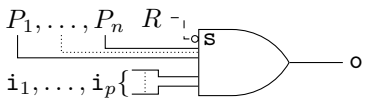
with propositional variables that can be seen as switches allowing to choose their truth values. The circuit as a whole corresponds to a Boolean expression and can be simplified accordingly. Popular techniques are based, for instance, on Karnaugh maps and the result is a circuit in sum-of-products form.

On the contrary, in factor circuits there is a unique kind of gate, the D-gate, whose behavior depends on its selector switch. Every D-gate implements a decomposition operator f_{P_i}, possesses two input ports because there are two truth values, and its output is $f_{P_i}(\mathtt{i}_1, \mathtt{i}_2)$. D-gates are connected through wires transporting elements of the $\hat{\nu}$-algebra $\mathbf{A}$. The remaining input wires are connected with switches representing algebraic variables ξ_i. Globally, a factor circuit represents a decomposition operator built up from basic decomposition operators (namely, those in $\hat{\nu}$). Factor circuits can be simplified by calculating their normal form using the term rewriting system defined in Section 4.10 (see Figure 4.4(b)). Note that a factor circuit in normal form has a particular shape (see Figure 4.4(c)): it is a binary tree such that all the D-gates $P_{i_1}, \ldots, P_{i_k}$ encountered in a root-to-leaf path have strictly increasing priority.

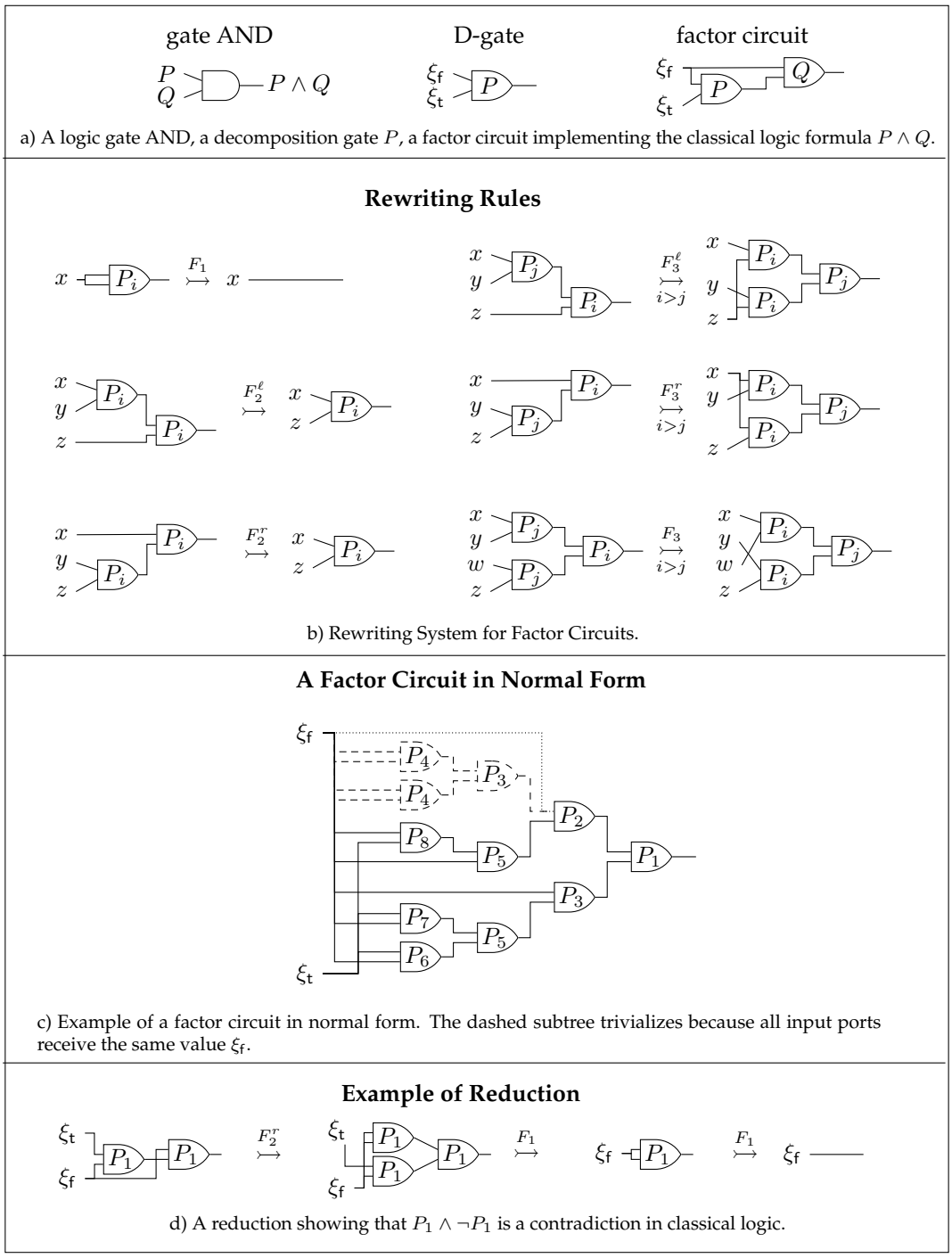
An interesting feature of factor circuits is that it is possible to exclude a sub-circuit by exploiting the algebraic properties of its components. Consider, for instance, the circuit in Figure 4.4(c) and suppose that we want to give $\xi_{\mathtt{f}}$ as first input of P_2 (rather than the result of $P_3(P_4(x, y), P_4(w, z))$). Then it is enough to connect the variable $\xi_{\mathtt{f}}$ to all input ports of the gates labelled with P_4 and the dashed subgraph trivializes thanks to axiom $(D1)$.

Problem 17. *What is the algebraic meaning of the "cycles" one could obtain by connecting in a factor circuit the output of a D-gate with one of its inputs?*

The D-gates for quantified matrix logics are a straightforward generalization of the propositional ones. Since an arbitrary D-gate represents a decomposition operator of shape $f_R \in \hat{\nu}_{n+p}$, it has n additional input parameters corresponding to the arguments of the relation $R(P_1, \ldots, P_n)$, that is it can be drawn as follows:



When composing arbitrary D-gates with each other, the new arguments do not play any role. In other words, it is forbidden to connect the output o with an input P_k. In a factor circuit the wires corresponding to $P_1, \ldots, P_n$ will remain as pending input lines.



a) A logic gate AND, a decomposition gate P, a factor circuit implementing the classical logic formula $P \wedge Q$.

b) Rewriting System for Factor Circuits.

c) Example of a factor circuit in normal form. The dashed subtree trivializes because all input ports receive the same value ξ_{f}.

d) A reduction showing that $P_1 \wedge \neg P_1$ is a contradiction in classical logic.

Figure 4.4: Factor circuits and decomposition gates.

4.12 SYMBOLIC COMPUTATION

Much work in computer science has been focused on reducing first-order logic to equational logic and term rewriting systems. In Tarski-Givant [TG87] one has a reduction of first-order Zermelo-Fraenkel set theory to traditional equational logic by using a sophisticated encoding into the equational logic of relation algebras. Burris and McKenzie's reduction of first-order logic with equality to equational logic through discriminator varieties uses a technique which is described in [Bur92]. The new technique of reduction introduced in [M29] and presented here is based on factor varieties and can be applied to first-order logic with or without equality.

Let ν be a relational type and let $T \cup \{\Phi\}$ be a finite set of first-order ν-sentences. One of the fundamental achievements of Gödel was to show that the semantic notion $T \models \Phi$ can be captured by a syntactic notion $T \vdash \Phi$. The usual procedure to avoid the manipulation of quantifiers consists in observing that $T \models \Phi$ holds if and only if $T \cup \{\neg\Phi\}$ is not satisfiable if and only if the set of sentences in $T \cup \{\neg\Phi\}$ Skolemized is not satisfiable. This reduces the syntactic level to universally quantified sentences. Such sentences are easily expressed as conjunctions of clauses (i.e., universally quantified disjunctions of atomic and/or negated atomic formulas), so we have $T \models \Phi$ if and only if a suitable set of clauses is not satisfiable. Robinson's resolution rule [Rob65] is complete for unsatisfiable sets of clauses, provided that the equality is not present in the language. In presence of equality, other rules must be introduced like paramodulation [NR99].

Burris and McKenzie replaces all atomic subformulas of the form $R(t_1, \ldots, t_n)$ in the universally quantified sentences obtained after Skolemization, by $f_R(t_1, \ldots, t_n) = t_1$, where f_R is a new function symbol corresponding to R (This approach to encoding relations as functions can be found in [Ack54, p. 98]). The switching function of a suitable discriminator variety is used to remove the logical connectives and to derive a set of equations axiomatizing a new discriminator variety, which can be used to analyze $T \models \Phi$ when we are working with a first-order language with equality.

Reduction to equations through factor varieties.

Let $T = \{\Psi_1, \ldots, \Psi_n\}$ be a set of first-order sentences in classical logic and Φ be a sentence. Our goal is to reduce the semantical problem of checking whether $T \models \Phi$ holds to an equational problem in factor varieties. This will be achieved by executing the following reduction procedure, and then applying Theorem 4.12.1 below.

Reduction procedure. Consider the set $\Sigma = \{\Psi_1, \ldots, \Psi_n, \neg\Phi\}$.

1. Convert all sentences in Σ into prenex normal form.
2. Compute the set $\Sigma^\sigma = \{\Psi_1^\sigma, \ldots, \Psi_n^\sigma, (\neg\Phi)^\sigma\}$ by Skolemizing the sentences obtained in Step 1. As it is customary, we omit the universal quantifiers in the Skolemized sentences.
3. Add to the relational type ν the new function symbols introduced by the Skolemization to obtain the new relational type μ.
4. Consider the $\hat{\mu}$-factor variety V_Σ axiomatized by:
   (i) the axioms (F1)-(F3);
   (ii) $(\Psi_1^\sigma)^* = \xi_\mathsf{t}, \ldots, (\Psi_n^\sigma)^* = \xi_\mathsf{t}$ and $((\neg\Phi)^\sigma)^* = \xi_\mathsf{t}$;
   (iii) $f_\mathsf{E}(x, x, \xi_\mathsf{f}, \xi_\mathsf{t}) = \xi_\mathsf{t}$ and $f_\mathsf{E}(x, y, x, y) = x$ *only if* the equality symbol E is present in the language.

Let us denote by $\mathrm{Ax}(\mathsf{V}_\Sigma)$ the set of these axioms.
We denote by $\vdash_{\mathrm{eq}}$ the deducibility in the equational calculus. We have the following completeness theorem.

Theorem 4.12.1 (Completeness Theorem, Salibra et Al. [M29])**.**
Let $\Phi, \Psi_1, \ldots, \Psi_n$ be first-order sentences in classical logic. Then we have $\Psi_1, \ldots, \Psi_n \models \Phi$ if and only if $\mathrm{Ax}(\mathsf{V}_\Sigma) \vdash_{\mathrm{eq}} \forall xy(x = y)$ and the propositional formula $(\Psi_1 \wedge \cdots \wedge \Psi_n \to \Phi)^{\mathrm{p}}$ is a tautology.

The following examples are described in [Bur92, pp. 198-199]. The reader can compare the simplicity of our method with respect to Burris's and McKenzie's reduction procedure.

Example 4.12.2. *Let T be empty and $\Phi = \forall x(R(x) \vee \neg R(x))$.*

- *Then $\neg\Phi$ is logically equivalent to $\exists x(\neg R(x) \wedge R(x))$.*
- *After Skolemization we obtain the formula $\neg R(c) \wedge R(c)$.*
- *We consider the factor variety axiomatized by the identity*

$$f_R(c, \xi_{\mathsf{f}}, f_R(c, \xi_{\mathsf{t}}, \xi_{\mathsf{f}})) = \xi_{\mathsf{t}},$$

  that implies $\xi_{\mathsf{f}} = \xi_{\mathsf{t}}$.
- *By Theorem 4.12.1 it follows that $\emptyset \models \Phi$.*

Example 4.12.3. *Let T be the theory axiomatized by:*

$$\begin{array}{ll} a \neq b, & \\ \forall x(x = a \vee x = b), & \forall xyz(R(x,y) \wedge R(x,z) \to y = z), \\ \forall x \exists y R(x,y), & \forall xyz(R(x,z) \wedge R(y,z) \to x = y). \end{array}$$

Let $\Phi = \forall y \exists x R(x,y)$ and $\Sigma = T \cup \{\neg\Phi\}$.
After Skolemization of Σ we get the following Σ^σ:

$$\begin{array}{llll} a \neq b, & x = a \vee x = b, & R(x,y) \wedge R(x,z) \to y = z, \\ R(x, g(x)), & \neg R(x,c), & R(x,z) \wedge R(y,z) \to x = y. \end{array}$$

The factor variety V_Σ is axiomatized by:

$$\begin{array}{l} f_{\mathsf{E}}(x,x,\xi_{\mathsf{f}},\xi_{\mathsf{t}}) = \xi_{\mathsf{t}}, \\ f_{\mathsf{E}}(x,y,x,y) = x, \\ f_{\mathsf{E}}(a,b,\xi_{\mathsf{t}},\xi_{\mathsf{f}}) = \xi_{\mathsf{t}}, \\ f_{\mathsf{E}}(x,b,f_{\mathsf{E}}(x,a,\xi_{\mathsf{f}},\xi_{\mathsf{t}}),\xi_{\mathsf{t}}) = \xi_{\mathsf{t}}, \\ f_R(x,z,\xi_{\mathsf{f}},f_R(x,y,\xi_{\mathsf{f}},y)) = f_R(x,z,\xi_{\mathsf{f}},f_R(x,y,\xi_{\mathsf{f}},z)), \\ f_R(x,z,\xi_{\mathsf{f}},f_R(y,z,\xi_{\mathsf{f}},x)) = f_R(x,z,\xi_{\mathsf{f}},f_R(y,z,\xi_{\mathsf{f}},y)), \\ f_R(x,g(x),\xi_{\mathsf{f}},\xi_{\mathsf{t}}) = \xi_{\mathsf{t}}, \\ f_R(x,c,\xi_{\mathsf{t}},\xi_{\mathsf{f}}) = \xi_{\mathsf{t}}. \end{array}$$

Since T has no singleton models, by Theorem 4.12.1 we have that $T \models \Phi$ if and only if we can equationally prove $\mathrm{Ax}(\mathsf{V}_\Sigma) \vdash_{\mathrm{eq}} a = b$.

Problem 18. *Verify whether factor varieties have unitary unification, which is at the basis of resolution theorem provers and of the Knuth-Bendix method for finding rewriting systems.*

Conclusions

We reviewed the main results obtained, jointly with other colleagues, in the last eight years. While describing these results, we individuated several interesting open problems that are listed on Page 95 in order to provide a global view. In this section we discuss more thoroughly the research lines that we think are the most promising and worth developing in the near future.

Generalizing the weighted relational semantics further. As shown in [M13, M14, M15], and discussed in Chapter 3, several categorical constructions are available for building quantitative models of the resource calculus. Some constructions give rise to categories of games that reveal precise intensional information about programs by interpreting them as sets of strategies, thus exposing their dynamic interaction with the environment (opponent). Other ones allow to add intensional information to the semantics of a program by interpreting it as a matrix taking values in a (continuous) commutative semiring $\mathcal{R}$ and, by varying the semiring, we can study different computational properties.

On the one hand, it would be interesting to combine these two approaches and define categories of games where strategies are instrumented with scalars from semirings, and check whether new quantitative properties of programs can be characterized. On the other hand, Laird recently showed in [Lai16] that the continuity condition on semirings, which is useful to define the fixed points in the resulting category, can be dropped by considering a different (but still canonical) construction for the fixed points. It would be interesting to verify whether the commutativity can be also dropped. In the non-commutative case, our construction give rise to pre-monoidal categories that could still have a structure rich enough to model PCF-like or imperative calculi. Interesting examples of non-commutative semirings would include semirings of formal languages, since the juxtaposition of strings is non-commutative. This would open the way for linking denotational semantics of programming languages with the theory of formal languages and automata theory.

A different line of generalization comes from the recent work of Hyland [Hyl10, Hyl15]. In these papers the author focuses on profunctors as a categorical generalization of the notion of relations. In the profunctorial semantics objects are categories and a map from a category $\mathbf{C}$ to a category $\mathbf{D}$ is a *profunctor*, that is a functor

$$F : \mathbf{D}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{Set}.$$

In [Win13], Winskel proposes the notion of profunctors as a generalization of strategies in games semantics. We wish to investigate whether this framework would allow to recover all the constructions described in Chapter 3 as instances.

Algebraic approach to Multi-Valued Logics. The preliminary investigations in [M29], summarized in Chapter 4, show that our approach for algebraizing logics through factor varieties is quite promising and deserves to be investigated further. To begin with, we wish to develop a theoretical and practical complexity analysis of our method. We have seen that the process of checking whether, for a propositional formula ϕ, $\phi^* = \mathsf{t}$ holds, can be automatized by defining a suitable term rewriting system. We know that such a term rewriting system is confluent and that, by labelling each operator f_P with a priority level, it is possible to ensure normalization. We plan to analyze the computational complexity of such a term rewriting system. On the one side we wish to generalize some theoretical results [BCMT01, Hof92] that cannot be directly applied since they rely on the presence

of constructors in the system. On the other side we plan to define optimal reduction and labelling strategies, to minimize the number of reduction steps. The hope is to prove that the optimal reduction strategy, combined with the optimal labelling, normalizes a term in polynomial time. Notice that this does not imply that we would be able to check whether a formula is a tautology in polynomial time, which would in its turn imply P = NP. Indeed, the size of a formula may increase exponentially in the translation, and the problem of finding an optimal labelling is NP-complete [BW96] (but good heuristic algorithms are known). We also plan to verify whether this approach extends to infinite propositional logics, that are logics having a infinitely many truth values. This will require to develop new techniques for handling term rewriting systems over infinitary terms. We expect that the resulting theory will be general enough to encompass fuzzy logic [Hàj98] and probabilistic logic [Nil86].

We have seen that our framework also suggests a new notion of logic circuits, that we call factor circuits and are based on components called D-gates that represent decomposition operators. The theory of factor circuits is still at the beginning. We plan to compare them with the usual multi-valued circuits [Eps93] and analyze advantages and disadvantages of the two approaches. We will also try to answer some natural questions, like the algebraic meaning (if any) of the "cycles" obtained by connecting the output of a gate with one of its inputs and the relationship between the classical simplification method based on Karnaugh map and the one based on our rewriting system.

We have also seen that the problem of algebraizing predicate logics is complicated because of the variable binding properties of the quantifiers. Much work in computer science has been focused on reducing first-order logic to equational logic. Starting from McKenzie's article [McK75], Burris made a substantial advance in [Bur92] by using discriminator varieties [Wer78]. Their works appears to have been largely overlooked by the proof assistants community, probably because it is not readily apparent how to manipulate the axioms of a discriminator variety. Our approach suggests a new procedure for reducing first-order classical logic to equational logic based on factor varieties and depending on much simpler axioms that are easier to manipulate. We plan to investigate whether factor varieties have unitary unification, which is at the basis of resolution theorem provers and of the Knuth-Bendix method for finding rewriting systems.

List of Problems

Problem 1. *The λ-theories representable by relational graph models belong to the interval $[\mathcal{B}, \mathcal{H}^*]$. How many distinct λ-theories can be represented by* rgms*? Is it possible to give a complete characterization of all representable λ-theories?*

Problem 2. *Is it possible to adapt the techniques developed by Breuvart in [Bre14] to characterize all relational graph models that are fully abstract for $\mathcal{H}^*$?*

Problem 3. *As reported in [Bar84, Proof of Thm. 17.4.16], Sallé conjectured that the inclusion $\mathcal{B}\omega \subseteq \mathcal{H}^+$ is actually strict. Prove Sallé's conjecture, or show $\mathcal{B}\omega = \mathcal{H}^+$.*

Problem 4. *Construct an "easy" assignment $\#(-)$ of (possibly transfinite) ordinals to simply typed λ-terms in such a way that $M \to_\beta N$ entails that $\#M > \#N$. By the fact that the ordinals are well-ordered, this immediately shows that the system is strongly normalizing.*

Problem 5. *While the reduction IHP $\leq_T$ DP presented in [M30] is a proper Turing reduction, DP $\leq_T$ IHP is an "ordinary" (many-one, actually one-to-one) reduction, which is logically simpler. Are IHP and DP equivalent with respect to the finer structure of many-one degrees?*

Problem 6. *Is it possible to characterize the must-solvability of the resource calculus in terms of outer-reduction and in terms of typability in a suitable type system?*

Problem 7. *Is it possible to define a convincing notion of Böhm tree for the full resource calculus?*

Problem 8. *Is it possible to prove an equational completeness theorem for categorical models of resource calculus without assuming an idempotent sum?*

Problem 9. *Show that the relational semantics does not contain any model fully abstract for the resource calculus.*

Problem 10. *Is it possible to find a fully abstract model of the untyped resource calculus in categories of games?*

Problem 11. *Show that the model $\mathcal{V}$ introduced in [M12] (equivalently, Ehrhard's models of [Ehr12]) is not equationally fully abstract for the call-by-value λ-calculus.*

Problem 12. *Provide a direct proof of the fact that every finite element of the relational semantics is the denotation of some term of Resource* PCF*, and conclude that it is fully abstract.*

Problem 13. *In [Lai16], Laird has shown that it is enough to start with a complete commutative semiring $\mathcal{R}$, and still obtain fixed points in $\mathcal{R}_!^\oplus$ without requiring any order-theoretic structure. These fixed points corresponding to infinite sums of finitary approximants indexed over the nested finite multisets, each representing a unique call-pattern for computation of the fixed point. It would be interesting to see whether the commutativity can be also released, working on the premonoidal setting, and what kind of languages it is possible to model in the coKleisli.*

Problem 14. *Show that $\mathcal{R}_!^\oplus$ is a fully abstract model for resource* PCF *extended with scalars from $\mathcal{R}$. Is the model fully abstract for an Erratic Idealized Algol with scalars?*

Problem 15. *What is the precise relationship between a logic $\mathcal{L}$ and the non-commutative intermediate logic $\mathcal{L}'$ induced by the translation in [M29]?*

Problem 16. *Consider the term rewriting system $\mathcal{R}$ defined in [M29]. What is the complexity of the system? Is the optimal reduction strategy effective? What is its complexity?*

Problem 17. *What is the algebraic meaning of the "cycles" one could obtain by connecting in a factor circuit the output of a D-gate with one of its inputs?*

Problem 18. *Verify whether factor varieties have unitary unification, which is at the basis of resolution theorem provers and of the Knuth-Bendix method for finding rewriting systems.*

Addendum. This manuscript intends to photograph the state of the art of our research in the day we started writing it (September 1, 2016). In the meanwhile, in collaboration with Intrigila and Polonsky, we solved negatively Sallé's conjecture appearing in the above list as Problem 3. In other words, we proved that the λ-theories $\mathcal{B}\omega$ and $\mathcal{H}^+$ coincide.

Notations

Set Theory

$\mathbb{N}$	set of natural numbers	5
$\mathbb{R}^+$	positive real numbers	5
$\lvert A\rvert$	cardinality of A	5
$\mathcal{P}(A)$	powerset of A	5
$A \subseteq_{\mathrm{f}} B$	A is a finite subset of B	5
$[]$	empty multiset	5
$a = [\alpha_1, \alpha_2, \ldots]$	multiset whose elements are $\alpha_1, \alpha_2, \ldots$	5
$a_1 \uplus a_2$	multiset union of a_1, a_2	5
$\mathcal{M}_{\mathrm{f}}(A)$	set of all finite multisets over A	5
$\mathcal{M}_{\mathrm{f}}(A)^{(\omega)}$	set of quasi-finite $\mathbb{N}$-indexed sequences of $\mathcal{M}_{\mathrm{f}}(A)$	5
$\mathcal{R}$	continuous semiring	58
$\lvert\mathcal{R}\rvert$	underlying set of $\mathcal{R}$	58
$\mathbf{0}$	neutral element of the sum in $\mathcal{R}$	58
$\mathbf{1}$	neutral element of the product in $\mathcal{R}$	58
$(\lvert\mathcal{R}\rvert, \preceq)$	cpo associated with $\mathcal{R}$	58
$\overline{X}$	$:= X \cup \{\infty\}$	58
$X_\perp$	$:= X \cup \{-\infty\}$	58
$\sum_{p\in I} p$	$:= \bigvee_{F\subseteq_{\mathrm{f}} I}(\sum_{p\in F} p)$	58
$\boldsymbol{\infty}$	$:= \sum_{p\in\mathcal{R}} p$	58
$\mathcal{B}$	Boolean semiring $(\{\mathsf{t},\mathsf{f}\}, \vee, \wedge, \mathsf{f}, \mathsf{t})$	58
$\mathcal{N}$	$\mathbb{N}$ completed $(\overline{\mathbb{N}}, +, \cdot, 0, 1, \leq)$	58
$\mathcal{T}$	Tropical semiring $(\overline{\mathbb{N}}, \min, +, \infty, 0, \geq)$	58
$\mathcal{A}$	Arctic semiring $(\overline{\mathbb{N}}_\perp, \max, +, -\infty, 0, \leq)$	58
$\mathcal{P}$	$\mathbb{R}^+$ completed $(\overline{\mathbb{R}^+}, +, \cdot, 0, 1, \leq)$	58
$\delta_{\alpha,\alpha'}$	Kronecker symbol	58

Category Theory

$\mathbf{C}$	arbitrary category	5
$\mathbf{1}$	category with one object and one morphism	53
$\mathbf{Rel}$	category of sets and relations	13
$\mathbf{MRel}$	co-Kleisli of $\mathcal{M}_{\mathrm{f}}(-)$ on $\mathbf{Rel}$	13
$\mathbf{PCoh}_!$	probabilistic coherence spaces	68
$\mathbf{G}$	category of arenas whose roots are all O-moves	54
$\mathbf{EG}$	category of exhausting games	55
$\mathbf{EG}_\sim$	subcategory of $\sim$-closed strategies of $\mathbf{EG}$	55
$\mathbf{G}_\sim$	subcategory of $\sim$-closed strategies of $\mathbf{G}$	55
$\mathbf{C}_!$	co-Kleisli category of ! on $\mathbf{C}$	6
$f \mathbin{;_!} g$	diagrammatic composition in $\mathbf{C}_!$	6
$\mathcal{K}_!(\mathbf{C})$	co-Kleisli category of $\mathcal{K}(\mathbf{C})$	52
$\mathcal{K}(\mathbf{C})$	Karoubi envelope of $\mathbf{C}$	52
$\mathbf{C}^+$	sup-lattice completion of $\mathbf{C}$	53
$\mathbf{C}^\oplus$	biproduct completion of $\mathbf{C}$	53

$\mathbf{C}^{\otimes}$	subcategory of comonoid homomorphisms of $\mathbf{C}$	6
$\mathbf{FamRel}(\mathbf{C})$	$(\mathbf{C}^{+})^{\oplus}$	53
A, Id_A	identity on A	5
$A \otimes B$	tensor product of A and B	5
$A^{\otimes n}$	n-fold tensor power of A	52
A^n	*symmetric* tensor power	52
$\Theta_{A,n} : A^{\otimes n} \to A^{\otimes n}$	the sum of the $n!$ permutation maps	52
$A \multimap B$	monoidal exponential object	5
ev	linear evaluation morphism	5
$\lambda(f)$	linear Currying	5
$A^{\perp}$	$:= A \multimap \perp$ (the dual of A)	5
$A \times B$	categorical product of A and B	5
T	terminal object	5
T^A	unique morphism in $\mathbf{C}(A, \mathsf{T})$	5
$A \,\&\, B$	categorical product in $\mathbf{MRel}$ (disjoint union)	5
π_1, π_2	projections	5
$\langle f, g \rangle$	pairing of f and g	5
$A \oplus B$	biproduct of A and B	5
ι_1, ι_2	injections	5
$[f, g]$	copairing of f and g	5
$A \to B$	exponential object	5
Eval	evaluation morphism	5
$\Lambda(f)$	Currying	5
contr	contraction	6
weak	weakening	6
der	dereliction	6
dig	digging	6
$f^{\dagger} : B \to !A$	unique comonoid morphism satisfying $f^{\dagger}; \mathrm{der}^A = f$	6
$\mathrm{m}^{\mathsf{T}}, \mathrm{m}^{A,B}$	Seely's isomorphisms	6
$\mathsf{D}(f)$	derivative of f in a Cartesian category	37
$f \star g$	linear application of of f to g	38
$\mathcal{U} = (U, \mathrm{app}, \mathrm{abs})$	linear reflexive object $U \to U \triangleleft U$	39

Universal Algebra

$\mathbf{A}$	algebra	70
τ	algebraic type	70
ν	relational type	80
τ_n	set of n-ary function symbols in τ	70
ν_n	relational type containing n-ary relation symbols	80
Σ	equational theory	75
$L(\Sigma)$	lattice of equational theories	75
Ξ	set of logical variables	82
ξ_i	logical variable	82
$\hat{\nu}$	algebraic type associated with ν	81
f_R	function in $\hat{\nu}_{n+p}$ associated with $R \in \nu_n$	81
Δ	$:= \{(a, a) \mid a \in A\}$	70

∇	$:= A \times A$	70
$\mathrm{Con}(\mathbf{A})$	set of congruences of $\mathbf{A}$	70
$[a]_\varphi$	$:= \{a' \in A \mid a \,\varphi\, a'\}$	70
$\vartheta(a, b)$	principal congruence generated by a and b	70
$\varphi \circ \vartheta$	$:= \{(a, c) \mid \exists b \in A,\ a \,\vartheta\, b \,\varphi\, c\}$	70
$\mathbf{A} \cong \mathbf{B}$	$\mathbf{A}$ and $\mathbf{B}$ are isomorphic algebras	70
$\mathbf{A} \leq \Pi_{i\in I}\mathbf{B}_i$	$\mathbf{A}$ is a subdirect product of $(\mathbf{B}_i)_{i\in I}$	70
ite	if-then-else term of a Church algebra	72
s	switch term of a discriminator variety	76
ϑ_e	$:= \vartheta(1, e)$	72
$\overline{\vartheta}_e$	$:= \vartheta(e, 0)$.	72
$\mathrm{Ce}(\mathbf{A})$	the set of central elements of $\mathbf{A}$	72
$f_e(x, y)$	$\mathsf{ite}(e, x, y)$	72
$e \leq d$	$\iff \overline{\vartheta}_e \subseteq \overline{\vartheta}_d$	73
$[\varphi)$	interval notation for $\{\varphi' \in \mathrm{Con}(\mathbf{A}) \mid \varphi \subseteq \varphi'\}$	73
v^T	transpose of the vector v	82
$\mathsf{v}(s)$	constant vector $[s, \ldots, s]^T$	82
$\phi \sim^* \psi$	equivalence induced by $(\cdot)^*$ on propositional formulas ϕ, ψ	83
$\mathrm{Str}(\mathbf{A})$	structure associated with a factor algebra $\mathbf{A}$	81
$\mathrm{Fa}(\mathcal{S})$	factor algebra associated with a structure $\mathcal{S}$	81
V_{fa}	class of $\hat{\nu}$-factor algebras belonging to a factor variety V	81
$\mathsf{V}_{\mathrm{prop}}$	factor variety generated by all p-factor algebras	84
$\mathrm{Ax}(\mathsf{V}_\Sigma)$	set of these axioms associated with V_Σ	91

Logic

$\mathcal{L}$	propositional logic $\mathcal{L}$	79
$\mathcal{QL}$	quantified logic	80
$\mathcal{C}$	Classical logic	79
$\mathcal{Ł}_n$	Łukasiewicz's n-ary logic	79
$\mathcal{G}_n$	Gödel's n-ary logic	79
$\mathcal{P}_n$	Post's n-ary logic	79
t, f	classical Boolean values	79
$\wedge$	logical and	79
$\vee$	logical or	79
$\neg$	logical not	79
$\rightarrow$	logical implication	79
τ	algebraic type of logical connectives	79
P	propositional variable	79
Pvar	set of propositional variables	79
ϕ, ψ	propositional formulas	79
$o(-_1, \ldots, -_n)$	n-ary connective $o \in \tau_n$	79
Φ, Ψ	well formed formulas	80
$R(-_1, \ldots, -_m)$	m-ary relation $R \in \nu_m$	80
T_ν	set of ν-terms	80
Φ^p	propositional translation of Φ	80
$\mathsf{LT}_{\hat{\nu}}$	logical terms of type $\hat{\nu}$	82

Games Semantics

$A \uplus B$	disjoint union of the arenas A and B	54
$A^{\perp}$	the arena A with O and P-moves interchanged	54
$A \multimap B$	the arena B with $A^{\perp}$ attached below each initial move	54
$\sigma : A \to B$	strategy in $A^{\perp} \uplus B$	54
M_A^P	Player move in the arena A	54
M_A^O	Opponent move in the arena A	54
$\vdash$	edge relation	54
(Q)	question	54
(A)	answer	54
$\ulcorner s \urcorner$	Player's view of s	54
$\mathsf{comp}(A)$	set of complete justified sequences of A	54
contr : $A \to A \uplus A$	copycat strategy which interleaves play in the two copies of A on the right to produce the play on the left (contraction)	54
$\mathsf{D}(\sigma)$	derivative of a strategy	54
$A \multimap R$	R-exponentials	55
$\sim$	$\sim$-equivalence on strategies	55

λ-calculus

Var	set of variables of λ-calculus	8
Λ	set of λ-terms	8
Λ^o	set of closed λ-terms	8
M, N, P, Q	λ-terms	8
$M\{N/x\}$	capture-free substitution	8
$C(\!\|-\|\!)$	λ-calculus context with a hole $(\!\|-\|\!)$	11
$\mathrm{FV}(M)$	set of free variables of M	8
I	$:= \lambda x.x$	8
K	$:= \lambda xy.x$	8
K'	$:= \lambda xy.y$	8
Ω	$:= (\lambda x.xx)(\lambda x.xx)$	8
Y	$:= \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$	8
J	$:= \mathrm{Y}(\lambda jxy.x(jy))$	8
J_T	η-expansion of I following the infinite tree T	15
$\to_\beta$	β-reduction	8
$\to_\eta$	η-reduction	8
$\to_{\beta\eta}$	$:= \to_\beta \cup \to_\eta$	8
$\twoheadrightarrow_{\mathtt{R}}$	multistep R-reduction	8
$=_{\mathtt{R}}$	R-conversion	8
$\mathrm{BT}(M)$	Böhm tree of M	10
$\perp$	empty Böhm tree	10
$\mathsf{App}(M)$	the set of all finite approximants of $\mathrm{BT}(M)$	14
$\lambda\mathcal{T}$	the lattice of λ-theories ordered by $\subseteq$	10
$\mathcal{T} \vdash M = N$	the λ-theory $\mathcal{T}$ equates M and N	10
$M =_{\mathcal{T}} N$	the λ-theory $\mathcal{T}$ equates M and N	10
$\boldsymbol{\lambda}$	smallest λ-theory	10
$\boldsymbol{\lambda\eta}$	smallest extensional λ-theory	10

$\mathcal{H}$	λ-theory equating all unsolvable λ-terms	10
$\mathcal{B}$	λ-theory equating all terms with the same Böhm tree	10
$\mathcal{H}^+$	Morris's extensional observational λ-theory	12
$\mathcal{H}^*$	maximal consistent λ-theory	11
$\mathcal{T}\eta$	smallest extensional λ-theory containing $\mathcal{T}$	17
$\mathcal{T}\omega$	closure of $\mathcal{T}$ under the (ω)-rule	17
$\mathcal{T} \vdash \omega$	$\mathcal{T}$ satisfies the ω-rule	17
$\mathcal{M}$	λ-calculus model	10
$[\![M]\!]^{\mathcal{M}}$	interpretation of the λ-term M in $\mathcal{M}$	10
$\mathcal{M} \models M = N$	M, N have the same interpretation in $\mathcal{M}$	10
$\mathrm{Th}(\mathcal{M})$	$:= \{M = N \mid \mathcal{M} \models M = N\}$	10
$\Lambda_{\mathcal{T}}$	term model of $\mathcal{T}$	40
$\equiv^{\mathcal{O}}$	observational equivalence where $\mathcal{O}$ is the set of observable	11
$\mathrm{BT}(M) =_{\eta\infty} \mathrm{BT}(N)$	$\mathrm{BT}(M)$ and $\mathrm{BT}(N)$ are equal up to infinite η-expansions	11
$\mathrm{BT}(M) =_{\eta\mathrm{fin}} \mathrm{BT}(N)$	$\mathrm{BT}(M)$ and $\mathrm{BT}(N)$ are equal up to finite η-expansions	12
$\mathcal{E}$	relational analogue of Engeler's model	14
$\mathcal{D}_\infty$	Scott's original model	11
$\mathcal{D}_\omega$	relational analogue of Scott's model	14
$\mathcal{D}_{\mathrm{CDZ}}$	Coppo, Dezani and Zacchi's filter model	12
$\mathcal{D}_\varepsilon$	relational analogue of Coppo, Dezani and Zacchi's model	14
$\mathsf{W}_{\mathcal{D}}(T)$	the set of all witnesses in $\mathcal{D}$ for T following some f	16

Simply typed λ-calculus

$\mathbb{T}^0$	set of simple types having a single ground type 0	18
Δ	type environment over simple types	18
$\Delta \vdash M : A$	judgement in simply typed λ-calculus	18
$\mathcal{M} = ((\mathcal{M}_A)_{A\in\mathbb{T}^0}, \cdot)$	typed applicative structure	19
$[\![\Delta \vdash M : A]\!]^{\mathcal{M}}_\nu, [\![M]\!]^{\mathcal{M}}_\nu$	interpretation of $\Delta \vdash M : A$ in $\mathcal{M}$ with respect to ν	19
$\nu\{d/x\}$	valuation such that $\nu\{d/x\}(x) = d$ and $\nu\{d/x\}(y) = \nu(y)$	19
$\mathcal{F} = (\mathcal{F}_A)_{A\in\mathbb{T}^0}$	full model of simply typed λ-calculus	21
$\mathcal{F}_n$	full model over a ground set of cardinality n	21
$\mathcal{S} = ((\mathcal{S}_A, \sqsubseteq_A)_{A\in\mathbb{T}^0}, \cdot)$	monotone model of simply typed λ-calculus	24
$f\uparrow$	upward closure $\{f' \in \mathcal{S}_A \mid f \sqsubseteq f'\}$ of $f \in \mathcal{S}_A$	24
$f \mapsto g$	step function	24
ι_A	order reversing isomorphism from $\mathcal{U}^\omega_X(A)$ to $\mathcal{S}_A$.	24
$\mathcal{R} = \{\mathcal{R}_A\}_{A\in\mathbb{T}^0}$	logical relation between typed applicative structures	19
$\mathcal{R}_A(Y)$	$:= \cup_{f\in Y}\mathcal{R}_A(f)$	19
$\mathcal{R}^-$	inverse of $\mathcal{R}$	19
$\mathcal{I}_0$	$:= \{(Y, Y) \mid Y \in \mathcal{P}(X)\}$	25
$\mathcal{I}$	logical retraction generated by the identity	25
$\mathcal{J}_0$	$:= \{(f, F) \mid f \in F \subseteq \mathcal{F}_0\}$	26
$\mathcal{J}$	logical relation generated by $\mathcal{J}_0$	26
$\mathcal{K}_0$	$:= \{(f, \{f\}) \mid f \in X\}$	26
$\mathcal{K}$	logical relation generated by $\mathcal{K}_0$	26

Intersection type systems

CDV	Coppo, Dezani and Venneri's intersection type system	20
CDV^{ω}	uniform intersection type system with stratified top ω	23
$\mathbb{A}$	set of atomic types	20
$\mathbb{T}_{\wedge}^{\mathbb{A}}$	set of intersection types over $\mathbb{A}$	20
$Y \to Z$	$:= \{\tau \to \sigma \mid \tau \in Y, \sigma \in Z\}$	22
$Y^{\wedge}$	$:= \{\sigma_1 \wedge \cdots \wedge \sigma_n \mid \sigma_i \in Y \text{ for all } 1 \leq i \leq n\}$	22
$\mathcal{U}_X(A)$	set of intersection types over X uniform with $A \in \mathbb{T}^0$	22
$\mathcal{G}$	Urzyczyn's game types	22
$\mathbb{T}_{\wedge}^{\mathbb{A}\cup\{\omega\}}$	intersection types over $\mathbb{A}$ and ω	23
$\mathcal{U}_X^{\omega}(A)$	set of intersection types $\mathcal{U}_{X\cup\{\omega\}}(A)$ with a stratified top ω_A	23
ω_A	stratified top element of uniform intersection types $\mathcal{U}_{\mathbb{A}}^{\omega}(A)$	23
$\leq$	subtyping	20
Γ	type environment over intersection types	20
$\Gamma \vdash_{\wedge} M : \sigma$	judgement in CDV	20
$\Gamma \vdash_{\wedge}^{\omega} M : \sigma$	judgement in CDV^{ω}	23

$\lambda_{+\|}$-calculus

$\Lambda_{+\|}$	set of nondeterminitic λ-calculus	47
$\mathrm{V}_{+\|}$	values of call-by-value $\lambda_{+\|}$-calculus	49
M, N, P, Q	set of λ-terms with parallel and nondeterministic choice	47
$M + N$	nondeterministic choice between M and N	47
$M \parallel N$	parallel composition of M and N	47
V	value	49
O	$:= (\lambda xy.xx)(\lambda xy.xx)$	51
$\to_{\mathtt{h}}$	one step head reduction	47
$\twoheadrightarrow_{\mathtt{h}}$	multi-step head reduction	47
$\odot$	mix-based merging operator on $\mathcal{D}_{\omega}$	47
$\mathbb{T}^{\|}$	set of parallel-types	49
$\mathbb{C}$	set of computational-types	49
Γ	environment of computational types	50
$\Gamma_1 \otimes \Gamma_2$	tensor product of Γ_1, Γ_2	50
$\Gamma \vdash_{\mathcal{V}} M : \sigma$	judgement for $\lambda_{+\|}$-calculus with intersection types	50
π	derivation tree	50
$\|\pi\|$	measure of a derivation tree π	50
$\equiv^{\mathsf{cbv}}$	operational equivalence in call-by-value λ-calculus	51
$\sqsubseteq^{\mathsf{cbv}}$	operational preorder in call-by-value λ-calculus	51

Resource Calculus

Λ^r	set of resource terms	32
M, N, L	resource terms	32
Λ^b	set of bags	32
P	bag of resources	32
1	empty bag	32
$P \cdot P'$	union of bags	32

$\mathbb{N}\langle\Lambda^r\rangle$	set of sums of resource terms	32
$\mathbf{M}, \mathbf{N}$	sum of resource terms	32
0	empty sum of resource terms	32
$\mathbf{P}$	sum of bags	32
$M\{N/x\}$	capture-free substitution of N for x in M	33
$M\langle N/x\rangle$	capture-free linear substitution of N for x in M	33
$\to_{\beta_\mathsf{r}}$	β_r-reduction	33
$\to_{\eta_\mathsf{r}}$	η_r-reduction	34
$\to_\mathsf{o}$	outer-reduction	34
$\mathscr{T}(M)$	Taylor expansion of M	35
$\mathrm{NF}_{\beta_\mathsf{r}}(\mathscr{T}(M))$	$= \{\mathrm{NF}_{\beta_\mathsf{r}}(t) \mid t \in \mathscr{T}(M)\}$	35
Λ^b_f	set of !-free resource terms	35
t	!-free resource term	35
b	bag of !-free resource term	35
$\equiv^{\mathscr{T}}$	congruence generated by setting $[x^!] = 1 + [x, x^!]$	36
$\equiv^{\mathscr{T}}_{\eta_\mathsf{r}}$	congruence generated by $\equiv^{\mathscr{T}} \cup \equiv_{\eta_\mathsf{r}}$	36
$\equiv^{\mathrm{monf}}$	may-observational equivalence	42
$\sqsubseteq^{\mathrm{monf}}$	may-observational preorder	42
$\Lambda^{\bar\tau}$	set of resource terms with tests	43
Q, R	resource tests	43
$\mathbf{Q}$	sum of resource tests	43
$Q\downarrow$	the test Q converges	43
$C(\!\|-\|\!)$	test-contexts of the resource calculus with tests	43
α^-	resource term associated with α	43
$\alpha^+(\!\|-\|\!)\downarrow$	test-context associated with α	43
$\equiv^\tau$	observational equivalence of resource calculus with tests	43

PCF **and its variations**

$\mathsf{PCF}^{\mathtt{or}}$	PCF extended with nondeterministic choice	57
Resource PCF	PCF extended with nondeterminism and bags of linear and reusable resources	56
$\mathsf{PCF}^{\mathcal{R}}$	$\mathsf{PCF}^{\mathtt{or}}$ extended with scalars from $\mathcal{R}$	61
$\mathscr{C}^{\Delta,A}_B$	set of $\mathsf{PCF}^{\mathcal{R}}$ contexts mapping terms of type A in Δ, into closed terms of type B	65 65
$Q(\!\|-\|\!)$	$\mathsf{PCF}^{\mathcal{R}}$ contexts	65
$\underline{n}$	numeral $\mathtt{succ}^n(\underline{0})$	61
Ω^{int}	$:= \mathtt{fix}(\lambda x^{\mathsf{int}}.x)$	64
Ψ	$:= \mathtt{fix}(\lambda x^{\mathsf{int}}.(x\ \mathtt{or}\ \underline{0}))$	64
Φ	$:= \lambda x^{\mathsf{int}}.\mathtt{ifz}(x, \mathtt{succ}\, x, \underline{0})$	64
Ξ	$:= \lambda y^{\mathsf{int}}.\infty\underline{0}$	65
Υ	$:= \lambda y^{\mathsf{int}}.(\infty\underline{0}\ \mathtt{or}\ \mathtt{ifz}(y, \underline{0}, \Omega^{\mathsf{int}}))$	65
$M \xrightarrow{\mathtt{p}}_\ell P$	elementary reduction step	62
$M \xrightarrow{\mathtt{p}} P$	$M \xrightarrow{\mathtt{p}}_\ell P$ for some label ℓ	62
π	reduction sequence	62
$M \Rightarrow^{\le k} P$	set of reduction sequences from M to P of length $\le k$	62

$M \Rightarrow P$	set of reduction sequences from M to P	62
$\mathrm{weight}(\pi)$	the weight of a reduction sequence π	62
$\lhd^A$	logical relation between vectors in $\mathcal{R}^A$ and closed $\mathsf{PCF}^{\mathcal{R}}$terms of type A	63
$\sqsubseteq^{\Delta}$	observational preorder of $\mathsf{PCF}^{\mathcal{R}}$	65
$\equiv^{\Delta}$	observational equivalence of $\mathsf{PCF}^{\mathcal{R}}$	65
$\mathrm{Red}_{M,P}$	Markov matrix describing the reduction from M to P	67

List of Coauthors

Hendrik Pieter Barendregt ..... Radboud University

Chantal Berline ..... Université Paris-Diderot

Flavien Breuvart ..... Université Paris-Nord

Antonio Bucciarelli ..... Université Paris-Diderot

Alberto Carraro ..... Università Ca'Foscari

Alejandro Díaz-Caro ..... Universidad Nacional de Quilmes

Thomas Ehrhard ..... Université Paris-Diderot

Giordano Favro ..... Università Ca'Foscari

Mai Gehrke ..... Université Paris-Diderot

James Laird ..... University of Bath

Guy McCusker ..... University of Bath

Michele Pagani ..... Université Paris-Diderot

Rinus Plasmeijer ..... Radboud University

Andrew Polonksy ..... Université Paris-Diderot

Domenico Ruoppolo ..... Université Paris-Nord

Sylvain Salvati ..... Université de Lille

Antonino Salibra ..... Università Ca'Foscari

Paolo Tranquilli ..... (ex) Università di Bologna

Bibliography

[Abr87] S. Abramsky. Domain theory in logical form. In *Proceedings, Symposium on Logic in Computer Science, 22-25 June 1987, Ithaca, New York, USA*, pages 47–53. IEEE Computer Society, 1987.

[AC98] R. Amadio and P.-L. Curien. *Domains and lambda-calculi*. Number 46 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1998.

[Ack54] W. Ackermann. *Solvable Cases of the Decision Problem*. Studies in Logic and the Foundations of Mathematics. North-Holland, 1954.

[AD08] P. Arrighi and G. Dowek. Linear-algebraic lambda-calculus: higher-order, encodings, and confluence. In A. Voronkov, editor, *Rewriting Techniques and Applications, 19th International Conference, RTA 2008, Hagenberg, Austria, July 15-17, 2008, Proceedings*, volume 5117 of *Lecture Notes in Computer Science*, pages 17–31. Springer, 2008.

[Ake78] S. B. Akers. Binary Decision Diagrams. *IEEE Transactions on Computers*, C-27(6):509–516, 1978.

[Bar71] H. P. Barendregt. *Some extensional term models for combinatory logics and λ-calculi*. Ph.D. thesis, Utrecht Universiteit, the Netherlands, 1971.

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

[Bas96] O. Bastonero. *Modèles fortement stables du λ-calcul et résultats d'incomplétude*. PhD thesis, Université de Paris 7, 1996.

[BBKV78] H. P. Barendregt, J. A. Bergstra, J. W. Klop, and H. Volken. Degrees of sensible lambda theories. *Journal of Symbolic Logic*, 43(1):45–55, 1978.

[BCD83] H. P. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *Journal of Symbolic Logic*, 48:931–940, 1983.

[BCMT01] G. Bonfante, A. Cichon, J. Marion, and H. Touzet. Algorithms with polynomial interpretation termination proof. *J. Funct. Program.*, 11(1):33–53, 2001.

[BCS06] R. Blute, J. Cockett, and R. Seely. Differential categories. *Mathematical Structures in Computer Science*, 16(6):1049–1083, 2006.

[BCS09] R. Blute, J. Cockett, and R. Seely. Cartesian differential categories. *Theory and Applications of Categories*, 22(23):622–672, 2009.

[BDER98] P. Baillot, V. Danos, T. Ehrhard, and L. Regnier. Timeless games. In M. Nielsen and W. Thomas, editors, *Computer Science Logic: 11th International Workshop, CSL '97, Annual Conference of the EACSL*, volume 1414 of *Lecture Notes in Computer Science*, pages 56–77. Springer-Verlag, 1998.

[BDPR79] C. Böhm, M. Dezani-Ciancaglini, P. Peretti, and S. Ronchi Della Rocca. A discrimination algorithm inside $\lambda\beta$-calculus. *Theoretical Computer Science*, 8(3):271– 291, 1979.

[BDS13] H. P. Barendregt, W. Dekkers, and R. Statman. *Lambda Calculus with Types*. Perspectives in logic. Cambridge University Press, 2013.

[Ber00] C. Berline. From computation to foundations via functions and application: The λ-calculus and its webbed models. *Theoretical Computer Science*, 249(1):81–161, 2000.

[BET12] R. Blute, T. Ehrhard, and C. Tasson. A convenient differential category. *Cahiers de topologie*, LIII:211–232, 2012.

[BG99] O. Bastonero and X. Gouy. Strong stability and the incompleteness of stable models of λ-calculus. *Annals of Pure and Applied Logic*, 100:247–277, 1999.

[Bir35] G. Birkhoff. On the structure of abstract algebras. In *Proc. Cambridge Philos. Soc.*, volume 31, pages 433–454, 1935.

[BKdV03] M. Bezem, J. W. Klop, and R. de Vrijer. *Term Rewriting Systems – TeReSe*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

[BL96] G. Boudol and C. Laneve. The discriminating power of multiplicities in the lambda-calculus. *Information and Computation*, 126(1):83–102, 1996.

[BL00] G. Boudol and C. Laneve. Lambda-calculus, multiplicities, and the π-calculus. In G. D. Plotkin, C. Stirling, and M. Tofte, editors, *Proof, Language, and Interaction, Essays in Honour of Robin Milner*, pages 659–690. The MIT Press, 2000.

[BL11] A. Bernadet and S. Lengrand. Complexity of strongly normalising λ-terms via non-idempotent intersection types. In M. Hofmann, editor, *Foundations of Software Science and Computational Structures - 14th International Conference, FOSSACS 2011*, volume 6604 of *Lecture Notes in Computer Science*, pages 88–107. Springer, 2011.

[Bou93] G. Boudol. The lambda-calculus with multiplicities (abstract). In E. Best, editor, *CONCUR '93, 4th International Conference on Concurrency Theory, Hildesheim, Germany, August 23-26, 1993, Proceedings*, volume 715 of *Lecture Notes in Computer Science*, pages 1–6. Springer, 1993.

[Bou94] G. Boudol. Lambda-calculi for (strict) parallel functions. *Information and Computation*, 108(1):51–127, 1994.

[Bre13] F. Breuvart. The resource lambda calculus is short-sighted in its relational model. In M. Hasegawa, editor, *Typed Lambda Calculi and Applications, 11th International Conference, TLCA 2013*, volume 7941 of *Lecture Notes in Computer Science*, pages 93–108. Springer, 2013.

[Bre14] F. Breuvart. On the characterization of models of $\mathcal{H}^*$. In T. Henzinger and D. Miller, editors, *Joint Meeting CSL-LICS '14*, pages 24:1–24:10. Association for Computing Machinery, 2014.

[Bre15] F. Breuvart. *Disséquer les Sémantiques Dénotationnelles*. Thèse de doctorat, Université Paris-Diderot, October 2015.

[Bry86] R. E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, 1986.

[Bry92] R. E. Bryant. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Comput. Surv.*, 24(3):293–318, 1992.

[BS81] S. Burris and H. P. Sankappanavar. *A course in universal algebra*. Springer-Verlag, Berlin, 1981.

[Bur92] S. Burris. Discriminator varieties and symbolic computation. *Journal of Symbolic Computation*, 13(2):175 – 207, 1992.

[BW96] B. Bollig and I. Wegener. Improving the variable ordering of obdds is np-complete. *IEEE Transactions on Computers*, 45(9):993–1002, 1996.

[Böh68] C. Böhm. Alcune proprietà delle forme β-η-normali nel λ-K-calcolo. *Pubblicazioni INAC*, 696:1–19, 1968.

[Böh75] C. Böhm, editor. *Lambda Calculus and Computer Science Theory, Proceedings of the Symposium Held in Rome March 25–27, 1975*. Number 37 in Lecture Notes in Computer Science. Springer-Verlag, 1975.

[CD80] M. Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre Dame Journal of Formal Logic*, 21(4):685–693, 1980.

[CDV80] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Principal Type Schemes and Lambda-Calculus Semantics. In *To H. B. Curry. Essays on Combinatory Logic, Lambda-calculus and Formalism*, pages 480–490. Academic Press, 1980.

[CDZ87] M. Coppo, M. Dezani-Ciancaglini, and M. Zacchi. Type theories, normal forms and $\mathcal{D}_\infty$-lambda-models. *Information and Computation*, 72(2):85–116, 1987.

[CG16] J. Cockett and J. Gallagher. Categorical models of the differential λ-calculus revisited. *Electronic Notes in Theoretical Computer Science*, 325:63 – 83, 2016. The Thirty-second Conference on the Mathematical Foundations of Programming Semantics (MFPS XXXII).

[Chu40] A. Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5(2):56–68, 1940.

[Com71] S. Comer. Representations by algebras of sections over Boolean spaces. *Pacific Journal of Mathematics*, 38:29–38, 1971.

[CR36] A. Church and J. B. Rosser. Some properties of conversion. *Transactions of the American Mathematical Society*, 39(3):472–482, 1936.

[dC07] D. de Carvalho. *Sémantiques de la logique linéaire et temps de calcul*. Thèse de doctorat, Université Aix-Marseille II, 2007.

[dC09] D. de Carvalho. Execution time of lambda-terms via denotational semantics and intersection types. *Technical Report*, abs/0905.4251, 2009. Available at `http://arxiv.org/abs/0905.4251`.

[DdP96] M. Dezani-Ciancaglini, U. de'Liguoro, and A. Piperno. Filter models for conjunctive-disjunctive lambda-calculi. *Theoretical Computer Science*, 170(1-2):83–128, 1996.

[DdP98] M. Dezani-Ciancaglini, U. de'Liguoro, and A. Piperno. A filter model for concurrent lambda-calculus. *SIAM Journal of Computing*, 27(5):1376–1419, 1998.

[DE11] V. Danos and T. Ehrhard. Probabilistic coherence spaces as a model of higher-order probabilistic computation. *Information and Computation*, 209(6):966–991, 2011.

[DFH99] P. Di Gianantonio, G. Franco, and F. Honsell. Game semantics for untyped $\lambda\beta\eta$-calculus. In *Typed Lambda Calculi and Applications, 4th International Conference, TLCA'99*, volume 1581 of *Lecture Notes in Computer Science*, pages 114–128. Springer, 1999.

[DK00] V. Danos and J. Krivine. Disjunctive tautologies as synchronisation schemes. In P. Clote and H. Schwichtenberg, editors, *Computer Science Logic, 14th Annual Conference of the EACSL*, volume 1862 of *Lecture Notes in Computer Science*, pages 292–301. Springer, 2000.

[DK09] M. Droste and W. Kuich. Semirings and formal power series. In M. Droste, W. Kuich, and H. Vogler, editors, *Handbook of Weighted Automata*, chapter 1. Springer-Verlag, 2009.

[dV87] R. de Vrijer. Exactly estimating functionals and strong normalization. *Indagationes Mathematicae (Proceedings)*, 90(4):479 – 493, 1987.

[EHR92] L. Egidi, F. Honsell, and S. Ronchi Della Rocca. Operational, denotational and logical descriptions: a case study. *Fundamenta Informaticae*, 16(1):149–169, 1992.

[Ehr02] T. Ehrhard. On Köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science*, 12:579–623, 2002.

[Ehr05] T. Ehrhard. Finiteness spaces. *Mathematical Structures in Computer Science*, 15(4):615–646, 2005.

[Ehr12] T. Ehrhard. Collapsing non-idempotent intersection types. In P. Cégielski and A. Durand, editors, *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the EACSL*, volume 16 of *LIPIcs*, pages 259–273. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[EL10] T. Ehrhard and O. Laurent. Interpreting a finitary pi-calculus in differential interaction nets. *Information and Computation*, 208(6):606–633, 2010.

[Eng81] E. Engeler. Algebras and combinators. *Algebra Universalis*, 13(3):389–392, 1981.

[Eps93] G. Epstein. *Multi-valued Logic Design: an Introduction*. IOP Publishing, London, 1993.

[ER03] T. Ehrhard and L. Regnier. The differential λ-calculus. *Theoretical Computer Science*, 309(1):1–41, 2003.

[ER05] T. Ehrhard and L. Regnier. Differential interaction nets. *Electronic Notes in Theoretical Computer Science*, 123:35–74, 2005.

[ER08] T. Ehrhard and L. Regnier. Uniformity and the Taylor expansion of ordinary λ-terms. *Theoretical Computer Science*, 403(2-3):347–372, 2008.

[FM09] G. Faure and A. Miquel. A categorical semantics for the parallel lambda-calculus. Research Report RR-7063, INRIA, 2009.

[Fri73] H. Friedman. Equality between functionals. In R. Parikh, editor, *Logic Colloquium: Symposium on Logic Held at Boston, 1972–73*, pages 22–37, Berlin, Heidelberg, 1973. Springer Berlin Heidelberg.

[Gan80a] R. O. Gandy. An early proof of normalization by A.M. Turing. In J. Seldin and J. Hindley, editors, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 453–455. Academic Press Limited, 1980.

[Gan80b] R. O. Gandy. Proofs of strong normalization. In J. Seldin and J. Hindley, editors, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 457–477. Academic Press Limited, 1980.

[Gir72] J.-Y. Girard. *Interprétation Fonctionnelle et Élimination des Coupures de l'Arithmétique d'Ordre Supérieur*. Thèse de doctorat, Université Paris 7, 1972.

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

[Gir88] J.-Y. Girard. Normal functors, power series and λ-calculus. *Annals of Pure and Applied Logic*, 37(2):129–177, 1988.

[Gir03] J. Girard. From foundations to ludics. *Bulletin of Symbolic Logic*, 9(2):131–168, 2003.

[GLT89] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Number 7 in Cambridge tracts in Theoretical Computer Science. Cambridge University Press, 1989.

[Got01] S. Gottwald. *A Treatise on Many-Valued Logics*, volume 9 of *Studies in Logic and Computation*. Research Studies Press Ltd., 2001.

[Gou94] J. Goubault. *Proving with BDDs and control of information*, pages 499–513. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.

[Gou95a] J. Goubault. A bdd-based simplification and skolemization procedure. *Logic Journal of the IGPL*, 3(6):827–855, 1995.

[Gou95b] X. Gouy. *Etude des théories équationnelles et des propriétés algébriques des modèles stables du λ-calcul*. Thèse de doctorat, Université de Paris 7, 1995.

[GP94] J. Goubault and J. Posegga. Bdds and automated deduction. In Z. W. Ras and M. Zemankova, editors, *Methodologies for Intelligent Systems, 8th International Symposium, ISMIS '94, Proceedings*, volume 869 of *Lecture Notes in Computer Science*, pages 541–550. Springer, 1994.

[Gue81] I. Guessarian. *Algebraic Semantics*, volume 99 of *Lecture Notes in Computer Science*. Springer, 1981.

[Göd32] K. Gödel. Zum intuitionistischen aussagenkalkül. In *Anzeiger der Akademie der Wissenschaften in Wien*, volume 69, pages 65–66, 1932.

[Hal54] P. R. Halmos. Polyadic Boolean algebras. *Proceedings of the National Academy of Sciences USA*, 40(5):296–301, 1954.

[Har99] R. Harmer. *Games and full abstraction for nondeterministic languages*. PhD thesis, University of London, 1999.

[HM99] R. Harmer and G. McCusker. A fully abstract game semantics for finite nondeterminism. In *14th Annual IEEE Symposium on Logic in Computer Science*, pages 422–430. IEEE Computer Society, 1999.

[HMT85] L. Henkin, J. D. Monk, and A. Tarski. *Cylindric algebras, Part I and II*. North-Holland, 1971, 1985.

[HNPR06] J. M. E. Hyland, M. Nagayama, J. Power, and G. Rosolini. A category theoretic formulation for Engeler-style models of the untyped λ-calculus. *Electronic Notes in Theoretical Computer Science*, 161:43–57, 2006.

[HO00] J. M. E. Hyland and C. L. Ong. On full abstraction for PCF: I, II, and III. *Information and Computation*, 163(2):285–408, 2000.

[Hof92] D. Hofbauer. Termination proofs by multiset path orderings imply primitive recursive derivation lengths. *Theoretical Computer Science*, 105(1):129–140, 1992.

[HR90] F. Honsell and S. Ronchi Della Rocca. Reasoning about interpretation in qualitative lambda-models. In M. Broy and C. Jones, editors, *Proceedings of Working Conference on Programming Concepts and Methods – IFIP 2.2*, pages 505–521. North-Holland, 1990.

[HR92] F. Honsell and S. Ronchi Della Rocca. An approximation theorem for topological lambda models and the topological incompleteness of lambda calculus. *Journal of Computer and System Sciences*, 45:49–75, 1992.

[Hyl75] J. M. E. Hyland. A survey of some useful partial order relations on terms of the λ-calculus. In *Lambda-Calculus and Computer Science Theory*, volume 37 of *Lecture Notes in Computer Science*, pages 83–95. Springer, 1975.

[Hyl10] M. Hyland. Some reasons for generalising domain theory. *Mathematical Structures in Computer Science*, 20(2):239–265, 2010.

[Hyl15] M. Hyland. Classical lambda calculus in modern dress. *Mathematical Structures in Computer Science*, pages 1–20, 2015.

[Hyl76] J. M. E. Hyland. A syntactic characterization of the equality in some models for the λ-calculus. *Journal London Mathematical Society (2)*, 12(3):361–370, 1975/76.

[Hàj98] P. Hàjek. *Metamathematics of Fuzzy Logic*. Kluwer, Dordrect, 1998.

[IS04] B. Intrigila and R. Statman. The omega rule is Π^0_2-hard in the $\lambda\beta$-calculus. In *Symposium on Logic in Computer Science (LICS 2004)*, pages 202–210. IEEE Computer Society, 2004.

[IS09] B. Intrigila and R. Statman. The omega rule is Π^1_1-complete in the $\lambda\beta$-calculus. *Logical Methods in Computer Science*, 5(2), 2009.

[Jol03] T. Joly. Encoding of the halting problem into the monster type and applications. In M. Hofmann, editor, *Typed Lambda Calculi and Applications: 6th International Conference, TLCA 2003*, pages 153–166, Berlin, Heidelberg, 2003.

[JT93] A. Jung and J. Tiuryn. A new characterization of lambda definability. In *Proceedings of the International Conference on Typed Lambda Calculi and Applications*, TLCA '93, pages 245–257, London, UK, UK, 1993. Springer-Verlag.

[Kar78] M. Karoubi. *K-theory*. Springer-Verlag, Berlin, New York, 1978.

[KL80] S. Kamin and J.-J. Levy. Two generalizations of the recursive path ordering. Technical report, University of Illinois, Urbana, Illinois, 1980.

[Koy82] C. Koymans. Models of the lambda calculus. *Information and Control*, 52(3):306–332, 1982.

[KT17] M. Kerjean and C. Tasson. Mackey-complete spaces and power series - a topological model of differential linear logic. *Mathematical Structures in Computer Science*, 2017. To appear.

[KZSS15] J. Kostolny, E. Zaitseva, S. Stojković, and R. Stanković. *Application of Multivalued Decision Diagrams in Computing the Direct Partial Logic Derivatives*, pages 41–48. Springer International Publishing, Cham, 2015.

[Laf88] Y. Lafont. *Logiques, catégories et machines*. PhD thesis, Université Paris 7, 1988.

[Lai06] J. Laird. Bidomains and full abstraction for countable nondeterminism. In L. Aceto and A. Ingólfsdóttir, editors, *Foundations of Software Science and Computation Structures, 9th International Conference, FOSSACS 2006*, volume 3921 of *Lecture Notes in Computer Science*, pages 352–366. Springer, 2006.

[Lai16] J. Laird. Fixed points in quantitative semantics. In M. Grohe, E. Koskinen, and N. Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16*, pages 347–356. ACM, 2016.

[Lam86] W. Lampe. A property of the lattice of equational theories. *Algebra Universalis*, 23:61–69, 1986.

[Lau02] O. Laurent. *Étude de la polarisation en logique*. PhD thesis, Université de Aix-Marseille II, France, 2002.

[LBR03] D. Le Botlan and D. Rémy. ML^{F}: raising ML to the power of system F. *SIGPLAN Notices*, 38(9):27–38, 2003.

[Lee59] C. Lee. Representation of switching circuits by binary-decision programs. *Bell System Technical Journal*, 38(4):985–999, 1959.

[Loa01] R. Loader. The undecidability of λ-definability. In C. A. Anderson and M. Zelëny, editors, *Logic, Meaning and Computation: Essays in Memory of Alonzo Church*, pages 331–342, Dordrecht, 2001. Springer Netherlands.

[LS04] S. Lusin and A. Salibra. The lattice of lambda theories. *J. Log. Comput.*, 14(3):373–394, 2004.

[Lu20] J. Łukasiewicz. O logice tròjwartościowej. *Ruch Filozoficzny*, 5:169–171, 1920. Translated in [McC67].

[Lév76] J.-J. Lévy. An algebraic interpretation of the $\lambda\beta k$-calculus; and an application of a labelled λ-calculus. *Theoretical Computer Science*, 2(1):97–114, 1976.

[Lév05] J.-J. Lévy. Le lambda calcul - notes du cours, 2005. In French. `http://pauillac.inria.fr/~levy/courses/X/M1/lambda/dea-spp/jjl.pdf`.

[Mal68] A. Malcev. Some borderline problems of algebra and logic. In *International Congress of Mathematicians (Moscow, 1966)*, pages 217–231. Mir Publishers, 1968.

[McC67] S. McCall. *Polish Logic 1920–1939*. Oxford University Press, 1967.

[McC98] G. McCusker. *Games and Full Abstraction for a Functional Metalanguage with Recursive Types*. Distinguished Dissertations in Computer Science. Springer-Verlag, 1998.

[McK75] R. McKenzie. On the spectra, and negative solution of the decision problem for identities having finite nontrivial model. *Journal of Symbolic Logic*, 40:186–196, 1975.

[McK83] R. McKenzie. Finite forbidden lattices. In *Universal algebra and lattice theory (Puebla, 1982)*, volume 1004 of *Lecture Notes in Mathematics*, pages 176–205, Berlin, 1983. Springer.

[MD02] D. M. Miller and R. Drechsler. On the construction of multiple-valued decision diagrams. In *Proceedings 32nd IEEE International Symposium on Multiple-Valued Logic*, pages 245–253, 2002.

[Mel06] P.-A. Melliès. Asynchronous games 2: the true concurrency of innocence. *Theoretical Computer Science*, 358:200–228, 2006.

[Mel09] P.-A. Melliès. Categorical semantics of linear logic. *Panoramas et Synthèses*, 27, 2009.

[Mil78] R. Milner. A theory of type polymorphism in programming. *Journal of Computer and System*, 17(3):348–375, 1978.

[Mil84] R. Milner. A proposal for standard ML. In *LISP and Functional Programming*, pages 184–197, 1984.

[MMT87] R. McKenzie, G. McNulty, and W. Taylor. *Algebras, lattices, varieties, Volume I*. Wadsworth Brooks, Monterey, California, 1987.

[Mor68] J. Morris. *Lambda calculus models of programming languages*. PhD thesis, MIT, 1968.

[MOTW99] J. Maraist, M. Odersky, D. N. Turner, and P. Wadler. Call-by-name, call-by-value, call-by-need and the linear λ-calculus. *Theoretical Computer Science*, 228(1-2):175–210, 1999.

[MTT09] P.-A. Melliès, N. Tabareau, and C. Tasson. An explicit formula for the free exponential modality of linear logic. In S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. Nikoletseas, and W. Thomas, editors, *Automata, Languages and Programming*, volume 5556 of *Lecture Notes in Computer Science*, pages 247–260. Springer Berlin / Heidelberg, 2009.

[New42] M. Newman. On theories with a combinatorial definition of equivalence. *Annals of Mathematics*, 43:223–243, 1942.

[New93] N. Newrly. Lattices of equational theories are congruence lattices of monoids with one additional unary operation. *Algebra Universalis*, 30:217–220, 1993.

[Nil86] N. J. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28(1):71–88, 1986.

[NR99] R. Nieuwenhuis and A. Rubio. Paramodulation-based theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 1, pages 371–443. Elsevier, 1999.

[NS03] S. Nagayama and T. Sasao. Compact representations of logic functions using heterogeneous mdds. In *33rd International Symposium on Multiple-Valued Logic, 2003. Proceedings.*, pages 247–252, May 2003.

[Ohl92] E. Ohlebusch. A note on simple termination of infinite term rewriting systems. Technical report, 1992.

[Ong93] C. L. Ong. Non-determinism in a functional setting. In *Proceedings of the Eighth Annual Symposium on Logic in Computer Science (LICS'93)*, pages 275–286. IEEE Computer Society, 1993.

[Pao06] L. Paolini. A stable programming language. *Information and Computation*, 204(3):339–375, 2006.

[Par92] M. 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.

[Pie67] R. Pierce. *Modules over commutative regular rings*. Memoirs Amer. Math. Soc., 1967.

[PL92] J. Posegga and B. Ludäscher. Towards first-order deduction based on shannon graphs. In H. J. Ohlbach, editor, *GWAI-92: Advances in Artificial Intelligence, 16th German Conference on Artificial Intelligence, Proceedings*, volume 671 of *Lecture Notes in Computer Science*, pages 67–75. Springer, 1992.

[Plo73] G. D. Plotkin. Lambda-definability and logical relations. Memorandum SAI–RM–4, University of Edinburgh, Edinburgh, Scotland, October 1973.

[Plo74] G. D. Plotkin. The lambda-calculus is ω-incomplete. *Journal of Symbolic Logic*, 39(2):313–317, 1974.

[Plo76] G. D. Plotkin. A powerdomain construction. *SIAM Journal of Computing*, 5(3):452–487, 1976.

[Plo77] G. D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5(3):225–255, 1977.

[Pos21] E. L. Post. Introduction to a general theory of propositions. *American Journal of Mathematics*, 43:163–185, 1921. Reprinted in [vH67].

[Pos92] J. Posegga. First-order shannon graphs (extended abstract). In B. Fronhöfer, R. Hähnle, and T. Käufl, editors, *Workshop Theorem Proving with Analytic Tableaux and Related Methods, Lautenbach. Universität Karlsruhe, Fakultät für Informatik, Institut für Logik, Komplexität und Deduktionssysteme, Interner Bericht 8/92, March 18-20, 1992*, pages 67–69, 1992.

[PPR15] L. Paolini, M. Piccolo, and S. Ronchi Della Rocca. Essential and relational models. *Mathematical Structures in Computer Science*, FirstView:1–25, 9 2015. DOI:10.1017/S0960129515000316.

[PR10a] M. Pagani and S. Ronchi Della Rocca. Linearity, non-determinism and solvability. *Fundamenta Informaticae*, 103(1-4):173–202, 2010.

[PR10b] M. Pagani and S. Ronchi Della Rocca. Solvability in resource lambda-calculus. In *Foundations of Software Science and Computation Structures*, volume 6014 of *Lecture Notes in Computer Science*, pages 358–373. Springer, 2010.

[PT09] M. Pagani and P. Tranquilli. Parallel reduction in resource lambda-calculus. In Z. Hu, editor, *Programming Languages and Systems, 7th Asian Symposium, APLAS 2009*, volume 5904 of *Lecture Notes in Computer Science*, pages 226–242. Springer, 2009.

[Rob65] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

[RP04] S. Ronchi Della Rocca and L. Paolini. *The Parametric λ-Calculus: a Metamodel for Computation*. EATCS Series. Springer, Berlin, 2004.

[Ruo16] D. Ruoppolo. *Relational Graph Models and Morris's observability*. Thèse de doctorat, Université Paris-Nord, 2016.

[RV84] S. Ronchi Della Rocca and B. Venneri. Principal type schemes for an extended type theory. *Theoretical Computer Science*, 28:151–169, 1984.

[RY10] D. Rémy and B. Yakobowski. A Church-style intermediate language for ML^F. In M. Blume, N. Kobayashi, and G. Vidal, editors, *Functional and Logic Programming, 10th International Symposium, FLOPS 2010*, volume 6009 of *Lecture Notes in Computer Science*, pages 24–39. Springer, 2010.

[RY12] D. Rémy and B. Yakobowski. A Church-style intermediate language for ML^F. *Theoretical Computer Science*, 435:77–105, 2012.

[Sal09] S. Salvati. Recognizability in the simply typed lambda-calculus. In H. Ono, M. Kanazawa, and R. J. G. B. de Queiroz, editors, *Logic, Language, Information and Computation, 16th International Workshop, WoLLIC 2009*, volume 5514 of *Lecture Notes in Computer Science*, pages 48–60. Springer, 2009.

[Sas97] T. Sasao. Ternary decision diagrams. survey. In *Proceedings 1997 27th International Symposium on Multiple- Valued Logic*, pages 241–250, May 1997.

[SB96] T. Sasao and J. T. Butler. A method to represent multiple-output switching functions by using multi-valued decision diagrams. In *Multiple-Valued Logic, 1996. Proceedings., 26th International Symposium on*, pages 248–254, May 1996.

[Sch91] H. Schwichtenberg. An upper bound for reduction sequences in the typed λ-calculus. *Archive for Mathematical Logic*, 30:405–408, 1991.

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

[Sco80] D. Scott. Relating theories of the lambda-calculus. In Hindley and Seldin, editors, *Essays on Combinatory Logic, Lambda-Calculus, and Formalism*, pages 589–606. Academic Press, 1980.

[See89] R. Seely. Linear logic, $\star$-autonomous categories and cofree coalgebras. *Contemporary mathematics*, 92, 1989.

[Sel02] P. Selinger. The lambda calculus is algebraic. *Journal of Functional Programming*, 12(6):549–566, 2002.

[Sie92] K. Sieber. Reasoning about sequential via logical relations. In *Proceedings of London Mathematical Society Symposium on Applications of Categories in Computer Science*, volume 177 of *London Mathematical Society Lecture Notes Series*. Cambridge University Press, 1992.

[SKMB90] A. Srinivasan, T. Kam, S. Malik, and R. K. Brayton. Algorithms for discrete function manipulation. In *IEEE/ACM International Conference on Computer-Aided Design, ICCAD 1990. Digest of Technical Papers*, pages 92–95. IEEE Computer Society, 1990.

[SLP15] A. Salibra, A. Ledda, and F. Paoli. Factor varieties. *Soft Computing*, pages 1–12, 2015.

[Sta82] R. Statman. Completeness, invariance and λ-definability. *Journal of Symbolic Logic*, 47(1):17–26, 1982.

[SU06] M. H. Sørensen and P. Urzyczyn. *Lectures on the Curry-Howard Isomorphism, Volume 149 (Studies in Logic and the Foundations of Mathematics)*. Elsevier Science Inc., New York, NY, USA, 2006.

[Tar35] A. Tarski. Grundzüge des systemenkalküls I. *Fundamenta Mathematicae*, 25:503–526, 1935.

[TG87] A. Tarski and S. Givant. *A formalization of set theory without variables*, volume 41. AMS Colloquium Publications, 1987.

[Tra09] P. Tranquilli. *Nets Between Determinism and Nondeterminism*. PhD thesis, Univ. Paris 7 and Univ. Roma 3, 2009.

[Tra11] P. Tranquilli. Intuitionistic differential nets and λ-calculus. *Theoretical Computer Science*, 412(20):1979–1997, 2011.

[Urz99] P. Urzyczyn. The emptiness problem for intersection types. *The Journal of Symbolic Logic*, 64(3):1195–1215, 1999.

[Vag96] D. Vaggione. Varieties in which the Pierce stalks are directly indecomposable. *Journal of Algebra*, 184:424–434, 1996.

[vH67] J. van Heijenoort, editor. *From Frege to Gödel; a source book in mathematical logic, 1879-1931*. Harvard University Press, Cambridge, Massachusset, 1967.

[Wad76] C. Wadsworth. The relation between computational and denotational properties for Scott's $\mathcal{D}_\infty$-models of the lambda-calculus. *SIAM Journal of Computing*, 5(3):488–521, 1976.

[Weg94] I. Wegener. Efficient data structures for boolean functions. *Discrete Mathematics*, 136(1-3):347–372, 1994.

[Wel94] J. B. Wells. Typability and type-checking in the second-order lambda-calculus are equivalent and undecidable. In *Proceedings of the Ninth Annual Symposium on Logic in Computer Science (LICS '94)*, pages 176–185. IEEE Computer Society, 1994.

[Wer78] H. Werner. *Discriminator Algebras*. Studien zur Algebra und ihre Anwendungen, Band 6, Akademie-Verlag, Berlin, 1978.

[Win13] G. Winskel. Strategies as profunctors. In F. Pfenning, editor, *Foundations of Software Science and Computation Structures - 16th International Conference, FOSSACS 2013, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2013*, volume 7794 of *Lecture Notes in Computer Science*, pages 418–433. Springer, 2013.

Personal Bibliography

[M1] A. Bucciarelli, A. Carraro, T. Ehrhard, and G. Manzonetto. Full abstraction for resource calculus with tests. In M. Bezem, editor, *Computer Science Logic, 25th International Workshop / 20th Annual Conference of the EACSL, CSL 2011*, volume 12 of *LIPIcs*, pages 97–111. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.

[M2] A. Bucciarelli, A. Carraro, T. Ehrhard, and G. Manzonetto. Full abstraction for the resource lambda calculus with tests, through Taylor expansion. *Logical Methods in Computer Science*, 8(4), 2012.

[M3] A. Bucciarelli, T. Ehrhard, and G. Manzonetto. Not enough points is enough. In J. Duparc and T. A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL*, volume 4646 of *Lecture Notes in Computer Science*, pages 298–312. Springer, 2007.

[M4] A. Bucciarelli, T. Ehrhard, and G. Manzonetto. A relational model of a parallel and non-deterministic lambda-calculus. In *Logical Foundations of Computer Science 2009*, volume 5407 of *Lecture Notes in Computer Science*, pages 107–121, 2009.

[M5] A. Bucciarelli, T. Ehrhard, and G. Manzonetto. Categorical models for simply typed resource calculi. *Electronic Notes in Theoretical Computer Science*, 265:213–230, 2010.

[M6] A. Bucciarelli, T. Ehrhard, and G. Manzonetto. A relational semantics for parallelism and non-determinism in a functional setting. *Annals of Pure and Applied Logic*, 163(7):918–934, 2012.

[M7] H. P. Barendregt and G. Manzonetto. Turing's contributions to lambda calculus. In B. Cooper and J. van Leeuwen, editors, *Alan Turing - His Work and Impact*, pages 139–143. Elsevier, 2013.

[M8] H. P. Barendregt, G. Manzonetto, and M. J. Plasmeijer. The imperative and functional programming paradigm. In B. Cooper and J. van Leeuwen, editors, *Alan Turing - His Work and Impact*, pages 121–126. Elsevier, 2013.

[M9] F. Breuvart, G. Manzonetto, A. Polonsky, and D. Ruoppolo. New results on Morris's observational theory: The benefits of separating the inseparable. In D. Kesner and B. Pientka, editors, *1st International Conference on Formal Structures for Computation and Deduction, FSCD 2016*, volume 52 of *LIPIcs*, pages 15:1–15:18. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[M10] C. Berline, G. Manzonetto, and A. Salibra. Lambda theories of effective lambda models. In J. Duparc and T. A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL*, volume 4646 of *Lecture Notes in Computer Science*, pages 268–282. Springer, 2007.

[M11] C. Berline, G. Manzonetto, and A. Salibra. Effective lambda-models versus recursively enumerable lambda-theories. *Mathematical Structures in Computer Science*, 19(5):897–942, 2009.

[M12] A. Díaz-Caro, G. Manzonetto, and M. Pagani. Call-by-value non-determinism in a linear logic type discipline. In S. N. Artëmov and A. Nerode, editors, *Logical*

Foundations of Computer Science, International Symposium, LFCS 2013, volume 7734 of *Lecture Notes in Computer Science*, pages 164–178. Springer, 2013.

[M13] J. Laird, G. Manzonetto, and G. McCusker. Constructing differential categories and deconstructing categories of games. In L. Aceto, M. Henzinger, and J. Sgall, editors, *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Proceedings, Part II*, volume 6756 of *Lecture Notes in Computer Science*, pages 186–197. Springer, 2011.

[M14] J. Laird, G. Manzonetto, and G. McCusker. Constructing differential categories and deconstructing categories of games. *Information and Computation*, 222:247–264, 2013.

[M15] J. Laird, G. Manzonetto, G. McCusker, and M. Pagani. Weighted relational models of typed lambda-calculi. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS*, pages 301–310. IEEE Computer Society, 2013.

[M16] G. Manzonetto. *Models and theories of lambda calculus*. PhD thesis, Università Ca'Foscari and Université Paris Diderot, 2008.

[M17] G. Manzonetto. A general class of models of $\mathcal{H}^*$. In R. Královic and D. Niwinski, editors, *Mathematical Foundations of Computer Science 2009, 34th International Symposium, MFCS 2009*, volume 5734 of *Lecture Notes in Computer Science*, pages 574–586. Springer, 2009.

[M18] G. Manzonetto. What is a categorical model of the differential and the resource λ-calculi? *Mathematical Structures in Computer Science*, 22(3):451–520, 2012.

[M19] G. Manzonetto and M. Pagani. Böhm's theorem for resource lambda calculus through Taylor expansion. In C. L. Ong, editor, *Typed Lambda Calculi and Applications - 10th International Conference, TLCA 2011*, volume 6690 of *Lecture Notes in Computer Science*, pages 153–168. Springer, 2011.

[M20] G. Manzonetto and A. Polonsky. On unification of lambda terms. In *22textsuperscriptnd International Conference on Types for Proofs and Programs, TYPES*, 2016.

[M21] G. Manzonetto and D. Ruoppolo. Relational graph models, Taylor expansion and extensionality. *Electronic Notes in Theoretical Computer Science*, 308:245–272, 2014.

[M22] G. Manzonetto and A. Salibra. Boolean algebras for lambda calculus. In *21th IEEE Symposium on Logic in Computer Science (LICS 2006)*, pages 317–326. IEEE Computer Society, 2006.

[M23] G. Manzonetto and A. Salibra. From lambda-calculus to universal algebra and back. In E. Ochmanski and J. Tyszkiewicz, editors, *Mathematical Foundations of Computer Science 2008, 33rd International Symposium, MFCS 2008*, volume 5162 of *Lecture Notes in Computer Science*, pages 479–490. Springer, 2008.

[M24] G. Manzonetto and A. Salibra. Lattices of equational theories as Church algebras. In C. Drossos, P. Peppas, and C. Tsinakis, editors, *Proc. 7th Panhellenic Logic Symposium*, pages 117–121. Patras University Press, 2009.

[M25] G. Manzonetto and A. Salibra. Applying universal algebra to lambda calculus. *Journal of Logic and Computation*, 20(4):877–915, 2010.

[M26] G. Manzonetto and P. Tranquilli. A calculus of coercions proving the strong normalization of ML^F. In *Proc. of 5th International Workshop on Higher-Order Rewriting*, pages 17–21, 2010.

[M27] G. Manzonetto and P. Tranquilli. Harnessing ml^f with the power of system F. In P. Hlinený and A. Kucera, editors, *Mathematical Foundations of Computer Science 2010, 35th International Symposium, MFCS 2010*, volume 6281 of *Lecture Notes in Computer Science*, pages 525–536. Springer, 2010.

[M28] G. Manzonetto and P. Tranquilli. Strong normalization of ML^F via a calculus of coercions. *Theoretical Computer Science*, 417:74–94, 2012.

[M29] A. Salibra, G. Manzonetto, and G. Favro. Factor varieties and symbolic computation. In M. Grohe, E. Koskinen, and N. Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16*, pages 739–748. ACM, 2016.

[M30] S. Salvati, G. Manzonetto, M. Gehrke, and H. P. Barendregt. Loader and Urzyczyn are logically related. In A. Czumaj, K. Mehlhorn, A. M. Pitts, and R. Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Part II*, volume 7392 of *Lecture Notes in Computer Science*, pages 364–376. Springer, 2012.

Index