UNIVERSITÉ PARIS XIII - SORBONNE PARIS NORD
École Doctorale Sciences, Technologies, Santé Galilée

Une Histoire de λ-calcul et d'Approximation

A Story of λ-calculus and Approximation

Thèse de doctorat en informatique
présentée par

Axel Kerinec

Laboratoire d'Informatique Paris Nord

Dirigée par Giulio Manzonetto

Soutenue le 6 juin 2023 devant le jury composé de:

Tom Hirschowitz	Rapporteur
Silvia Ghilezan	Rapporteur
Delia Kesner	Jury
Stefano Guerrini	Jury
Marie Kerjean	Jury
Pierre Clairambault	Jury

Abstract

Böhm trees are historically the first notion of λ-calculus approximation, and were introduced in [Barendregt, 1977] for the Call-by-Name λ-calculus. They enjoy an interesting link with the other notion of approximation that is the Taylor expansion from [Ehrhard and Regnier, 2003]: the normal form of the Taylor expansion of a λ-term corresponds to the Taylor expansion of its Böhm tree.

The theory of program approximation in the Call-by-Value λ-calculus is far less developed. We present the first notion of Böhm tree in this context, using the Call-by-Value setting enhanced with permutation rules from [Carraro and Guerrieri, 2014]. We observe that the λ-terms with the same Böhm tree are observationally equivalent and we provide a characterisation on a set of approximants for it to be the Böhm tree of some λ-term. A refinement of those approximants allows to characterise solvability.

The connection between our Böhm trees and the Taylor expansion is slightly different than for the Call-by-Name case: here the normal form of the Taylor expansion of a λ-term corresponds with the normalised Taylor expansion of the Böhm tree of this term.

In the second part of this work, we focus on the Call-by-Name case. We introduce a class of categorified graph models in a distributor-induced bicategorical semantics. Those models can be seen as a categorification of traditional relational models and they can similarly be presented as intersection type systems. We prove an approximation theorem for them: the interpretation of a λ-term corresponds to the filtered colimit of the interpretations of its approximants i.e. the interpretation of its Böhm tree.

Our models are actually proof-relevant: the interpretation does not contain only typings but whole type derivations. This additional information, compared to traditional models, allows to prove a commutation theorem: the normal form of the denotation of a λ-term coincides with the denotation of its Böhm tree.

From a derivation we can reconstruct a minimal approximant with the desired type in the same environment and we demonstrate that any derivation in the interpretation of a λ-term M but not in the interpretation of another λ-term N induces an approximant of M but not of N. From this, we deduce the characterisation of the theory of categorified graph models: this theory is $\mathcal{B}$, two λ-terms have isomorphic interpretations if and only if their Böhm trees are the same.

Résumé

Les arbres de Böhm sont historiquement la première notion d'approximation pour le λ-calcul, ils ont été introduit dans [Barendregt, 1977] pour le λ-calcul en Appel-par-Nom. Ils jouissent d'un lien intéressant avec l'autre notion d'approximation qu'est le développement de Taylor ([Ehrhard and Regnier, 2003]): la forme normale du développement de Taylor d'un λ-terme correspond au développement de Taylor de son arbre de Böhm.

La théorie de l'approximation des programmes est beaucoup moins developpée pour le λ-calcul en Appel-par-Valeur. Nous présentons la première notion d'arbre de Böhm dans ce contexte, en utilisant le λ-calcul Appel-par-Valeur étendu par les règles de permutation de [Carraro and Guerrieri, 2014]. Nous constatons que les λ-termes avec les mêmes arbres de Böhm sont équivalents observationnellement, et nous caractérisons les ensembles d'approximants qui correspondent aux arbres de Böhm de λ-termes. Ensuite nous caractérisons la solvabilité en utilisant des approximants plus précis.

La connexion entre nos arbres de Böhm et le développement de Taylor est légèrement différente du cas Appel-par-Nom: ici la forme normale du développement de Taylor d'un λ-terme correspond à une version normalisée du développement de Taylor de l'arbre de Böhm de ce terme.

Dans la seconde partie de ce travail, nous nous intéressons au λ-calcul en Appel-par-Nom. Nous introduisons les modèles de graphe catégorifiés dans une sémantique bicatégorique basée sur les distributeurs. Ils peuvent être vus comme une catégorification des modèles relationnels traditionnels et pareillement ils peuvent être présentés comme des systèmes de type intersection.

Dans ce cadre, nous prouvons un théorème d'approximation: l'interprétation d'un λ-terme correspond à la colimite filtrée de l'interprétation de ses approximants i.e. à l'interprétation de son arbre de Böhm.

Nos modèles sont sensibles aux preuves: l'interprétation ne contient pas seulement les typages mais les dérivations de type. De cette information supplémentaire, comparé aux modèles traditionnels, nous déduisons un théorème de commutation: la forme normale de la dénotation d'un λ-terme coïncide avec la dénotation de son arbre de Böhm.

D'une dérivation nous trouvons un approximant minimal avec les bons types et environnement et nous montrons qu'une dérivation dans l'interprétation d'un terme M mais pas dans celle d'un autre terme N induit un approximant de M qui n'approxime pas N. Nous en déduisons la caractérisation de la théorie de nos modèles: cette théorie est $\mathcal{B}$, les interprétations de deux λ-termes sont isomorphes si et seulement si ces termes ont le même arbre de Böhm.

Contents

1. Introduction

Logic was at first, a long time ago in Greece, the study of reasoning and language. In the end of the XIXth century it became about mathematical languages and no longer about human ones. From that time we can praise the work of Frege, Russell, Peano, Hilbert,... In the 20's and 30's the mathematical understanding of algorithms and computation had become the focus point. Even before the creation of computers, this was a golden age for computer science.

Many different equivalent systems arose, trying to capture the essence of computation: Turing machines, partial recursive functions by Gödel, rewriting systems by Post, combinatory logic by Schönfinkel and Curry... Among them was one that will be widely used in theoretical computer science and that is central in the present work: the λ-calculus.

The λ-calculus. The λ-calculus was created by Church in the 30's with the idea of being a "formal calculus" based on functions rather than on traditional sets ([Church, 1932, Church, 1941]).

The λ-calculus and the famous Turing machine are computationally equivalent ([Kleene, 1936, Turing, 1937]). However, even though Turing machines are visually great and easily understandable tools to represent algorithms, they are not really satisfying from a mathematical perspective: a Turing machine basically follows a sequence of instructions using a given space to write, akin to assembly code. Other equivalent structures such as partial recursive functions are, conversely, too abstract and lose the principle of computation. The λ-calculus reduction is a key to symbolise it:

$$(\lambda x.M)N \rightarrow_\beta M[N/x]$$

the rule (β) corresponds to the rewriting of the λ-term $(\lambda x.M)N$ by replacing the λ-term N for all free occurrences of the variable x inside the λ-term M. We can also use it to rewrite subterms inside a λ-term.

The underlying idea is to see the λ-term $\lambda x.M$ as a function $f(x)$ depending on the variable x and N as the argument of this function. As an example, consider a function representing a polynomial: $f(x) = x^2 - 3x + 42$, then $f(4) = 4^2 - 3 \times 4 + 42$. Basically, $f(4)$ corresponds to the replacement of the occurrences of x by 4. In λ-notation, this could informally be denoted $(\lambda x.(x^2 - 3x + 42))4 \rightarrow_\beta 4^2 - 3 \times 4 + 42$.

This simple but powerful reduction rule is the core of λ-calculus.

Applying it can also be seen as executing a step in a program. In fact, the λ-calculus is the kernel of many functional programming languages and has been throughout the years a major subject of study for computer scientists.

Regarding functions or programs it is natural to be interested in the "meaning" of individual λ-terms. If a λ-term cannot be reduced then it is called a normal form, and λ-terms reaching normal forms by reduction are called normalisable. Since λ-calculus is a consistent rewriting system and the reduction rule is confluent (due to [Church and Rosser, 1936]), each λ-term has at most one normal form, reached after a sequence of reductions. We can take as example:

$$\lambda w.(((\lambda x.xx)(\lambda z.yz))u) \to_\beta \lambda w.(((\lambda z.yz)(\lambda z.yz))u) \to_\beta \lambda w.((y(\lambda z.yz))u).$$

One could then consider those normal forms as the meaning of λ-terms, akin the meaning of results of functions. However not all λ-terms have normal forms: some of them lead to infinite chains of reductions. We could at first consider those terms as meaningless. But λ-terms are not mathematical functions, where only the extensional definition matters. In the same way that a program computing the decimals of π has a purpose but never ends, non-normalisable λ-terms are not meaningless in general. Still, some of them are, for example: $\Omega = (\lambda x.xx)(\lambda x.xx) \to_\beta \Omega$ endlessly reduces to itself.

Solvability. Meaningful terms are usually considered to be the solvable ones. A term is solvable if it can be reduced to a fully defined result (traditionally, the identity $\lambda x.x$) when used as subterm of some λ-term with a specific shape. More precisely, M is solvable if there are variables $x_1, \ldots x_n$ and λ-terms $N_0, \ldots N_l$ such that

$$(\ldots((\lambda x_1(\lambda x_2(\ldots(\lambda x_n.M)\ldots))N_0)\ldots)N_l) \to_\beta \cdots \to_\beta \lambda x.x.$$

The notion of solvability was first studied in [Wadsworth, 1971, Wadsworth, 1976, Barendregt, 1971, Barendregt, 1977].

A λ-term may be solvable even with unsolvable subterms, if those can be erased in the right environment (without erasing the whole λ-term). A classical example of an unsolvable λ-term is Ω. However $(\lambda yx.x)\Omega$ is solvable, since $(\lambda yx.x)\Omega \to_\beta \lambda x.x$. In this case, the subterm Ω has no influence on the whole λ-term and could be replaced by any other subterm without altering the reduction properties.

Solvable terms may be seen as the terms producing some information, even if they sometimes have no normal form.

They have been characterised in various ways. Among them, in [Wadsworth, 1976] this characterisation is performed operationally, by the mean of a specific reduction called head reduction: the solvable terms are the ones with a head normal form. Solvability also have a strong link with the famous notion of Böhm tree.

Böhm trees. Böhm introduced a separability theorem stating that two normalisable λ-terms whose normal forms are distinguished modulo η-equivalence (and renaming of bound variables) whenever they can be separated by an applicative context ([Böhm, 1968]). Equivalently, looking at the contrapositive, if two terms behave in the same way when applied to any λ-terms, then their normal forms must be similar. In other words, the extensional equivalence on λ-calculus normal forms is definable in terms of syntactic equivalences.

This result is particularly appealing because the proof is constructive and inspired a tree representation of normal forms. In [Barendregt, 1977], this representation was extended to terms with no normal forms, calling it Böhm tree. The Böhm tree of a λ-term is constructed by successively reducing the term and collecting parts of it that will never change again during following reductions.

Different equivalent methods for defining Böhm trees exist, famous ones comes from [Lassen, 1999] and [Amadio and Curien, 1998]. They respectively use coinduction and a language of approximants.

Those approximants are λ-terms in normal forms but possibly containing a special subterm used to represent the indefiniteness: $\perp$. A pre-order is defined on them, considering $\perp$ smaller than any subterm. The approximants of a given λ-term may be infinitely many but they are all of finite size and comparable. They have a similar external shape and $\perp$ is used in deeper subterms for more precise approximants. The Böhm tree of a λ-term is given by the supremum of its approximants.

The labels on the tree representation consist of the external subterms that are settled during reductions and the study of the behaviour of the λ-term corresponds to the study of finite parts of the tree. Unsolvable λ-terms, such as Ω, have just $\perp$ as approximant, since they do not produce any information and no subterm become permanent while reducing.

Models. A whole world is devoted to the meaning associated with the terms of a theory: the semantics. The aim is to introduce the interpretation of the syntax of a language, such as the λ-calculus, in a suitable model.

In particular, denotational semantics are based on the thought that programs and terms are symbolic representations of abstract mathematical concepts. Thus the principle is to associate the appropriate mathematical object (number, function, tuple...) with the objects of the syntax. Moreover the interpretation should be stable under reduction: indeed a term and its reducts have the same "meaning".

The notion of denotational semantics is mainly issued from the work of Strachey and Scott in [Scott, 1970, Scott and Strachey, 1971].

The first model of λ-calculus, $\mathcal{D}_\infty$, was introduced in [Scott, 1972] in the category of complete lattices and Scott-continuous functions. What is formally a model of λ-calculus? A categorical definition states that models of λ-calculus are reflexive objects living in some Cartesian Closed Category. They can also equivalently be seen as combinatory algebras satisfying some axioms.

After $\mathcal{D}_\infty$, a lot of other models of λ-calculus were created (see [Barendregt, 1984, Plotkin, 1993]). In [Coppo and Dezani-Ciancaglini, 1980], a type assignment system was introduced: intersection types. This framework assigns types to λ-terms, with the specificity that such a system takes into account that a term can be typed in several ways by means of the intersection constructor $\cap$. Typing the λ-term M with $a \cap b$ means that M can be typed both with a and with b. The intuition derives from realisability semantics, where programs that realise $a \cap b$ are programs that realise a and b, individually. In such a model the interpretation of a λ-term is given by the set of its typings:

$$[\![M]\!] = \{(\Gamma, a) \mid \Gamma \vdash M : a\}$$

where $\Gamma \vdash M : a$ means that M can be typed with a in an environment Γ (setting the types of the free variables of M).

In the same way that, in programming, types are used to ensure that a string is not given as an entry to a program needing a number, they are here assigned to a λ-term through the use of diverse rules depending of the types of its subterms. For example, in a given environment, a λ-term $\lambda x.M$, representing a function, has a type of the form $a \to b$, and can only be applied to λ-terms N of type a. The "result" of the function $M[N/x]$ is (then) of type b. And since the "meaning" does not change while rewriting, $(\lambda x.M)N$ is also of type b.

Major properties of normalisation in λ-calculus can be characterised using intersection types, like head normalisation, β-normalisation and strong normalisation ([Krivine, 1993], [Bernadet and Lengrand, 2013, Bucciarelli et al., 2017]). Solvability can also be characterise using them: a λ-term is solvable if and only if it is typable with intersection types ([Coppo and Dezani-Ciancaglini, 1980]).

If the operator $\cap$ is non-idempotent ($a \cap a \neq a$), then the type system becomes resource sensitive ([Gardner, 1994, de Carvalho, 2007]). Now a type $a_1 \cap \cdots \cap a_n \Rightarrow b$ captures the exact number of resources needed during computations, no argument can be duplicated nor deleted. This is a really capital property when one is interested in programs. Indeed when performing $(\lambda x.M)N \to_\beta M[N/x]$ we duplicate N (or delete it) the number of times x appears in M and make the replacement in one unique step, which is highly unnatural from a computational point of view. Taking into account the resources with types is a real improvement.

Resources allow to prove operational properties such as β-normalisation by combinatorial means [Bucciarelli et al., 2017], and give quantitative operational information about solvable terms: the number of head reduction steps in [Accattoli and Dal Lago, 2012] and the size of the head normal form in [de Carvalho, 2007, de Carvalho, 2018].

λ-theories. Models are strongly linked to λ-theories. The λ-theories are essential if the focus is on the equivalence between terms rather than on their computational process. Formally, they are defined as congruences on λ-terms containing β-conversion (two terms are β-convertible if we can reach one from the other by following a sequence of β-rules where we forget the direction of the arrows). Each denotational model generates a λ-theory, by equating the terms having the same interpretation in the model:

$$\mathrm{Th}(\mathcal{D}) = \{(M, N) \mid [\![M]\!]_{\mathcal{D}} = [\![N]\!]_{\mathcal{D}}\}.$$

They can also be designed to capture a given operational behaviour. Famous examples are $\mathcal{H}$ axiomatised by equating unsolvable λ-terms, $\mathcal{B}$ equating λ-terms with the same Böhm tree, and $\mathcal{H}^*$ equating all observationally indistinguishable λ-terms.

Scott's model $\mathcal{D}_\infty$ equates all unsolvable terms, in fact its theory is exactly $\mathcal{H}^*$.

We aim at defining fully abstract models. By fully abstract models we mean models whose induced theories capture some observational equivalence between λ-terms: two terms are observationally equivalent if and only if they have the same interpretation in the model. A mandatory but not sufficient step toward full abstaction is equating the λ-terms that have the same Böhm tree.

Linear Logic. Another groundbreaking invention concerning resources is Linear Logic (in [Girard, 1987]). Independently from intersection type systems, and inspired by the coherence spaces category, Girard decomposed the intuitionistic arrow $a \Rightarrow b$ into $(!a) \multimap b$ using two new constructors:

! means "as much as we want" and allows to duplicate or erase.

$\multimap$ is a linear implication sensitive to the amount of resources.

A program typed with $a \multimap b$ uses its input of type a exactly once during computation in order to produce an output of type b. But a program typed with $(!a) \multimap b$ can use the input as much as needed.

In terms of denotational models it is a refinement of the intuitionistic interpretation, Cartesian Closed Categories are replaced by Symmetric Monoidal Closed Categories.

The most emblematic semantics of Linear Logic is the relational semantics in the category Rel of Set and Relations. Its coKelisli MRel originated from [Girard, 1988] but was first studied in [de Carvalho, 2007, Hyland et al., 2006, Bucciarelli et al., 2007]. It constitutes the simplest quantitative model of linear logic, where $a \multimap b$ is given by the Cartesian product $a \times b$ and $!a$ by the set of all finite multisets over a.

The semantics induced by the category Rel can be presented as a non-idempotent intersection type system (System R) with a non-idempotent operator [de Carvalho, 2007].

Categorification. Relational Semantics has been generalised in many ways. A promising one is categorification: set-theoretic concepts are replaced by categorical ones and the traditional models in Cartesian closed categories are transposed in 2-categories or bicategories. The first work on this topic is [Seely, 1987], where the author sketched a connection between 2-categories equipped with a (lax) Cartesian closed structure and the rewriting rules of the simply-typed λ-calculus. Following this idea, it is showed in [Hilken, 1996] that 2-categories can be used to model rewriting via 2-cells between terms denotations. In [Hirschowitz, 2013], the author constructed 2-dimensional type theories to describe 2-categorical structures in rewriting theory.

In [Melliès and Zeilberger, 2015] the authors gave a categorification of type systems, representing them as functors between a category of type derivations and a category of terms. Following those works and [Hyland, 2017], a higher categorical approach to intersection types and linear approximation, based on the setting of multicategories and discrete distributors, was introduced in [Mazza et al., 2017].

The passage from 2-categories to bicategories consists in a weakening of the structure: some equalities become equivalences "up to isomorphism". This allows to adopt a term-rewriting perspective when studying denotational semantics. Now, the interpretations of λ-terms M and N such that $M \to_\beta N$ are only equal up to isomorphism:

$$\beta : [\![M]\!] \cong [\![N]\!].$$

In [Tsukada et al., 2017] the authors showed that this isomorphism can be interpreted with a reduction relation on some approximants of the λ-calculus. This inspired the author of [Olimpieri, 2021, Olimpieri, 2020], where models of λ-calculus living in distributors-based bicategories are introduced, and syntactically presented as intersection type systems. In addition, this semantic is proof-relevant in the sense that the interpretation of a term is the whole set of its typing derivations, and not only its typings as in classical models:

$$[\![M]\!](\Gamma, a) \cong \left\{ \begin{array}{c} \pi \\ \vdots \\ \Gamma \vdash M : a \end{array} \right\}$$

where π is a derivation of $\Gamma \vdash M : a$.

Among the bicategorical constructions, it is natural to work with distributors, which can be seen as a categorification of relations between sets. They had been used multiple times: in [Cattani and Winskel, 2005] for a bicategorical model of linear logic generalising Scott's domains, in [Fiore et al., 2008] for introducing the bicategory of generalised species of structures, and then in [Olimpieri, 2021, Olimpieri, 2020].

Taylor expansion. It is often convenient to see λ-terms as functions, they even have an equivalent of the major tool that is differentiation! Differential λ-calculus was introduced in [Ehrhard and Regnier, 2003]. This followed previous works ([Ehrhard, 2005, Ehrhard, 2002]), where linear logic and λ-calculus were extended with differential constructions.

The differential extension of λ-calculus is called resource calculus ([Tranquilli, 2009]). It is similar to λ-calculus except that the reduction is sensitive to the amount of needed resources: they are explicitly represented and consumed linearly. Thus the size of a resource term strictly decreases during reduction and the calculus is strongly normalising.

An essential ingredient of classical analysis is the Taylor expansion: a way of approximating a function near a point as an infinite sum of terms depending of the function's derivatives at this point. Near 0, The expansion of $f(x)$ is given by:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)x^n}{n!}$$

where $f^{(n)}$ corresponds to the n-th derivative of f. We can notice that a term of rank n "uses" exactly n times x.

This notion has been captured for λ-calculus in [Ehrhard and Regnier, 2003]. The Taylor expansion translates λ-terms into a (possibly) infinite linear combination of resource terms. Since resource calculus is strongly normalising, the Taylor expansion of a λ-term can always be normalised. The infinitary behaviour of a λ-term with no normal form is captured by the infinite number of resource terms in its Taylor expansion even if each one of them has a normal form.

The non-idempotent intersection type assignment system R is also strongly linked to the Taylor expansion ([Ehrhard and Regnier, 2008, de Carvalho, 2007]): the interpretation of a λ-term in the relational model can be recovered straightforwardly from the normal form of its Taylor expansion.

There is a really strong result, relating Taylor expansion (denoted here by $\mathscr{T}(-)$) and Böhm tree (denoted $\mathrm{BT}_\beta(-)$):

$$\textit{for any } \lambda\textit{-term } M\textit{:}\ \ \mathrm{NF}(\mathscr{T}(M)) = \mathscr{T}(\mathrm{BT}_\beta(M)).$$

The normal form of the Taylor expansion of a λ-term is the Taylor expansion of the Böhm tree of this term ([Ehrhard and Regnier, 2008, Ehrhard and Regnier, 2006a]). Thus the Taylor expansion can be seen as a quantitative version of the Böhm tree.

Call-by-Value. On a whole different level, when approaching λ-calculus with a more practical point of view, the β-reduction leads to some problems. Indeed, we can perform β-reduction with arguments being λ-terms not in normal forms, while in real programming languages the steps have a specific order: subprograms are computed before being used as arguments.

The choice of the order in which subterms are reduced in a λ-term is named reduction strategy. The classical β-reduction introduced by Church corresponds to the Call-by-Name reduction strategy.

Here we will focus on Call-by-Value: some λ-terms are called values and only values can be used as arguments. This strategy is actually used in many modern functional programming languages, like OCaml.

The terminology "Call-by-Value" seems to have first appeared in the report defining ALGOL 60 (see [Backus et al., 1960]). ALGOL 60 is a computer programming language of the ALGOL family, born in 1960. It is the first language implementing nested function definitions with lexical scope.

The first appearance of "Call-by-Value" is in Section 4.7.3.1:

> *4.7.3.1. Value assignment (call by value)*
>
> *All formal parameters quoted in the value part of the procedure declaration heading are assigned the values (cf. section 2.8. VALUES AND TYPES) of the corresponding actual parameters, these assignments being considered as being performed explicitly before entering the procedure body. These formal parameters will subsequently be treated as local to the procedure body.*

Landin noticed strong connection between ALGOL 60 and λ-calculus and he formalised the principle of mapping values to variables in [Landin, 1964]. The first formalisation of Call-by-Value λ-calculus appeared in [Plotkin, 1975], together with a method for simulating Call-by-Value with Call-by-Name, and *vice versa*.

Sadly, there are way less results for Call-by-Value than for Call-by-Name. In particular, the Call-by-Value theory of program approximation is not as developed. For instance, it was unclear what should be the Böhm tree of a λ-term because some redexes may get stuck while waiting for a value (see [Ronchi Della Rocca and Paolini, 2004], and the discussion in [Accattoli and Guerrieri, 2016]). For example, the term $(\lambda y.(\lambda x.xx))(xx)(\lambda x.xx)$ is in normal form in Call-by-Value, since xx is not a value and cannot be used as argument, while $(\lambda y.(\lambda x.xx))(xx)(\lambda x.xx) \to_\beta (\lambda x.xx)(\lambda x.xx) = \Omega$ in Call-by-Name.

In [Ronchi Della Rocca and Paolini, 2004] the authors show that the continuous model built in [Egidi et al., 1992] does satisfy an Approximation Theorem stating that the interpretation of a λ-term in the model is given by the supremum of the interpretations of its finite approximants. But the considered approximants turn out to be too weak for capturing any interesting operational property.

And while a plethora of adequate models — models which equate all λ-terms that are observationally equivalent — has been constructed, e.g., in the Scott continuous and in the stable semantics [Egidi et al., 1992, Pravato et al., 1999, Honsell and Lenisa, 1993], none is fully abstract. Since one of the main interests of denotational models is to supply tools for proving equivalence between terms, full abstraction is often researched and adequacy needed.

But there is hope: a possible solution has been proposed in [Carraro and Guerrieri, 2014]. They introduced permutation reductions inspired by linear logic, allowing to unblock stuck redexes without altering fundamental operational properties of the calculus, such as the capability of a program to reduce to a value or such as the notion of (Call-by-Value) solvability (as shown in [Guerrieri et al., 2017]). Equivalently, in [Accattoli and Paolini, 2012], the authors introduced a refinement of Call-by-Value where the substitution is explicit and can be delayed. This setting has allowed to internally characterise Call-by-Value solvability for the first time.

In recent years there has been a renewal of interest in the Call-by-Value λ-calculus: the interested reader may consult [Accattoli and Guerrieri, 2016, Guerrieri et al., 2017, Accattoli and Guerrieri, 2018, Manzonetto et al., 2019a, Bucciarelli et al., 2020] as well as [Kesner and Peyrot, 2022, Arrial et al., 2023].

Relational models for Call-by-Value λ-calculus also exist, and were introduced in [Ehrhard, 2012]. The author was inspired by the relational semantics of Linear Logic ([Girard, 1988]) and he exploited Girard's "boring" translation of intuitionistic arrow in linear logic, sending $A \to B$ into $!(A \multimap B)$. In the same paper and exploiting the same translation, Ehrhard also introduced a resource calculus and a Taylor expansion for the Call-by-Value λ-calculus. This opens the door to the development of a theory of approximation similar to the one in Call-by-Name.

1.1. Content

My work is divided in two parts, the first one is about syntax and comes from a conjoint work with Giulio Manzonetto and Michele Pagani in [Kerinec et al., 2020] and with Giulio Manzonetto and Simona Ronchi Della Rocca in [Kerinec et al., 2021]. The second part is about semantics and presents results obtained with Giulio Manzonetto and Federico Olimpieri and published in [Kerinec et al., 2023].

PART I: Syntax

Chapter 2 This chapter recalls some definitions used for the rest of this thesis: first of the Call-by-Name λ-calulus and then of the Call-by-Value one.

Chapter 3 The first part of this chapter is devoted to an introduction to the classical Böhm trees for Call-by-Name λ-calculus, with both the coinductive and the inductive definitions. Then an original definition of Böhm trees for Call-by-Value is given. We define the Böhm tree of a term as the supremum of appropriate approximants. We then provide necessary and sufficient conditions on sets of approximants for them to correspond to Böhm trees of some λ-terms.

Chapter 4 This chapter is purely in a Call-by-Value framework, it contains the definitions of the resource calculus and the Taylor expansion. We also characterise those sets of resource terms arising as the Taylor expansion of some λ-term.

Chapter 5 This chapter is still in a Call-by-Value framework. We present investigations of our Böhm tree definition, in particular of the relation between Böhm trees and Taylor approximation. We also prove that the normal form of the Taylor expansion of a λ-term coincides with the Taylor expansion of its Böhm Tree.

From this theorem we deduce a characterisation of the potential valuability. Finally we present a study of Call-by-Value solvability.

Chapter 6 This chapter is a conclusion of Part I.

PART II: Semantics

Chapter 7 This chapter sums-up basic definitions and theorems for categories and bicategories.

Chapter 8 This chapter and the following ones consider the Call-by-Name setting. We present bicategorical models, and show how to explicitly model the computation with the help of the second dimension. Then, we introduce the bicategory of distributors which will be the framework of our investigations.

Chapter 9 This chapter presents a generalisation of relational graph models in categorified graph models. They allow to extend the notion of bicategorical model and can be presented as intersection type assignment systems, where the intersection is non-idempotent. In fact the interpretation of a λ-term can be seen as an intersection type distributor. We also give a definition for the interpretation of Böhm trees.

Chapter 10 This chapter introduces a notion of reduction on intersection type distributors, and the normal form of the interpretation of a λ-term. We prove a commutation theorem: for any λ-term the normal form of its interpretation equates the interpretation of its Böhm tree. Such a theorem recalls us the crucial one between Böhm trees and Taylor expansion.

We construct for each derivation an associated minimal approximant. Using those minimal approximants we prove inductively an approximation theorem.

Chapter 11 This chapter is devoted to the study of the λ-theory associated to the bicategorical model, and this is in fact $\mathcal{B}$.

Chapter 12 In this chapter we perform a decategorification and adapt the previous results in the 1-dimensional setting, in particular the approximation theorem.

Chapter 13 This chapter is a conclusion of Part 2.

Part I.

Syntax

2. λ-calculus

In this chapter we will present basic definitions, examples and theorems about λ-calculus, first Call-by-Name and then Call-by-Value. Even if both of them are well known, we believe it is interesting to recall those notions since they constitute the starting point of our work.

2.1. Rewriting Systems

A *rewriting system* is a set of objects $\mathcal{X} = \{X_1, X_2, \dots\}$ together with a binary relation R on $\mathcal{X}^2$. The underlying meaning behind $(X_1, X_2) \in R$ is "X_1 is rewritten in X_2 using the rule R". In respect to the rewriting systems tradition we will write $X_1 \to_{\mathrm{R}} X_2$ when $(X_1, X_2) \in R$.

Definition 2.1.1. *Given a binary relation $R \subseteq \mathcal{X}^2$ we define:*

- *the reflexive closure of $\to_{\mathrm{R}}$:*

$$\to_{\mathrm{R}} \cup \{(X_1, X_1) \mid X_1 \in \mathcal{X}\};$$

- *the transitive closure $\to_{\mathrm{R}}$:*

$$\begin{gathered}\{(X_1, X_1') \mid X_1, X_1' \in \mathcal{X}, \exists X_2, \dots, X_n \in \mathcal{X}, n \geq 1, \\ X_1 \to_{\mathrm{R}} X_2 \to_{\mathrm{R}} \dots \to_{\mathrm{R}} X_n \to_{\mathrm{R}} X_1'\};\end{gathered}$$

- *the* multistep R-reduction *as the reflexive-transitive closure of $\to_{\mathrm{R}}$ i.e. the smallest transitive and reflexive binary relation containing $\to_{\mathrm{R}}$, denoted $\twoheadrightarrow_{\mathrm{R}}$;*
- *the symmetric closure of $\to_{\mathrm{R}}$:*

$$\to_{\mathrm{R}} \cup \{(X_1, X_1') | X_1, X_1' \in \mathcal{X}, X_1' \to_{\mathrm{R}} X_1\};$$

- *the* R-conversion *as the reflexive, transitive and symmetric closure of $\to_{\mathrm{R}}$ i.e. the symmetric closure of $\twoheadrightarrow_{\mathrm{R}}$, denoted $=_R$.*

Definition 2.1.2. *An object $X_1 \in \mathcal{X}$ is:*

- R-normal *if there is no $X_2 \in \mathcal{X}$ such that $X_1 \to_R X_2$;*
- R-normalisable *if there is a R-normal $X_2 \in \mathcal{X}$ such that $X_1 \twoheadrightarrow_{\mathrm{R}} X_2$;*
- strongly R-normalising *if there is no sequence $(X_i)_{i \in \mathbb{N}^*} \in \mathcal{X}$ such that $X_i \to_R X_{i+1}$ for every $i \in \mathbb{N}^*$.*

Definition 2.1.3. *A rewriting system $(\mathcal{X}, \rightarrow_R)$ is:*

- strongly normalising *if all objects of $\mathcal{X}$ are strongly R-normalising;*
- locally confluent *if for all $X_1, X_2, X_2' \in \mathcal{X}$ such that $X_2 \leftarrow_R X_1 \rightarrow_R X_2'$, there exists $X_3 \in \mathcal{X}$ such that $X_2 \twoheadrightarrow_{\mathrm{R}} X_3 \twoheadleftarrow_{\mathrm{R}} X_2'$;*
- confluent *if for all $X_1, X_2, X_2' \in \mathcal{X}$ such that $X_2 \twoheadleftarrow_{\mathrm{R}} X_1 \twoheadrightarrow_{\mathrm{R}} X_2'$, there exists $X_3 \in \mathcal{X}$ such that $X_2 \twoheadrightarrow_{\mathrm{R}} X_3 \twoheadleftarrow_{\mathrm{R}} X_2'$.*

Two rules $\rightarrow_1$ and $\rightarrow_2$ commute if for all $X, Y_1, Y_2 \in \mathcal{X}$ having $X \twoheadrightarrow_1 Y_1$ and $X \twoheadrightarrow_2 Y_2$ imply there exists $Y \in \mathcal{X}$ such that $Y_2 \twoheadrightarrow_1 Y$ and $Y_1 \twoheadrightarrow_2 Y$.

If a rule R is confluent then $X_1 \in \mathcal{X}$ reduces to at most one R-normal X_2, if so we call X_2 the *R-normal form* (R-nf) of X_1 and write $X_2 = \mathrm{NF}_R(X_1)$.

Lemma 2.1.4 (Newman's Lemma)**.** *If a rewriting system is strongly normalising and locally confluent then it is confluent.*

Lemma 2.1.5 (Hindley-Rosen)**.** *If $\rightarrow_1$ and $\rightarrow_2$ are confluent and commute then $\rightarrow_1 \cup \rightarrow_2$ is confluent.*

2.2. Call-by-Name

Concerning the syntax of λ-calculus, we mainly use the notations of [Barendregt, 1984] and explicitly specify when it is not the case. We consider fixed a countably infinite set $\mathbb{V}$ of *variables* denoted $x, y, z, \ldots$ (possibly with indices).

Definition 2.2.1. *The set Λ of λ-terms is defined by the following grammar:*

$$(\Lambda) \qquad M, N, L ::= \; x \;\mid\; MN \mid\; \lambda x.M \qquad \textit{(for } x \in \mathbb{V}\textit{)}$$

We call $\lambda x.M$ an *abstraction*, it represents a traditional function depending of the variable x. On the other side MN is called an *application* and symbolises a function M applied to an argument N.

Application associates to the left, and has higher precedence than abstraction.

Example 2.2.2.

- $\lambda xyz.xyz = \lambda x.(\lambda y.(\lambda z.((xy)z)))$;
- $\lambda xy.z(xy)\lambda z.zyy = \lambda x.(\lambda y.((z(xy))(\lambda z.((zy)y)))))$.

Notation 2.2.3. *Given $x_1, \ldots, x_n \in \mathbb{V}$, we let $\lambda\vec{x}.M$ stands for $\lambda x_1 \ldots \lambda x_n.M$, and given $M_1, \ldots, M_n \in \Lambda$, $\vec{M}$ stands for $M_1 \ldots M_n$. We write MN^n for $M\underbrace{N \cdots N}_{n \text{ times}}$.*

$\mathbf{I}$	$=$	$\lambda x.x$	*the identity;*
$\mathbf{1}$	$=$	$\lambda xy.xy$	*an η-expansion of* $\mathbf{I}$;
$\mathbf{B}$	$=$	$\lambda fgx.f(gx)$	*the composition;*
Δ	$=$	$\lambda x.xx$	*the self-applicator;*
Ω	$=$	$\Delta\Delta$	*the paradigmatic looping*;
$\mathbf{K}$	$=$	$\lambda xy.x$	*the first projection;*
$\mathbf{F}$	$=$	$\lambda xy.y$	*the second projection*;
$\mathbf{P_n}$	$=$	$\lambda x_0 \ldots x_n.x_n$	*erases n arguments*;
$\mathbf{Y}$	$=$	$\lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$	*Curry's fixed point combinator*;
$\mathbf{Z}$	$=$	$\lambda f.(\lambda y.f(\lambda z.yyz))(\lambda y.f(\lambda z.yyz))$	*Plotkin's recursion operator;*
$\mathbf{K}^*$	$=$	$\mathbf{ZK}$	*produces an increasing amount of external abstractions.*

Figure 2.1.: Some useful combinators.

The set $\mathrm{FV}(M)$ of *free variables of M* and the *α-conversion* are defined as usual (see [Barendregt, 1984, §2.1]):

Definition 2.2.4 (Free variables and α-conversion)**.** *Given $M \in \Lambda$:*

- *for $N = \lambda x.M$ we say that x is a* bound variable *of N;*
- *the set of* free *variables of a λ-term M is denoted by* $\mathrm{FV}(M)$*, namely:*
  $\mathrm{FV}(x) = \{x\}$, $\mathrm{FV}(\lambda x.M) = \mathrm{FV}(M)/\{x\}$ *and* $\mathrm{FV}(MN) = \mathrm{FV}(M) \cup \mathrm{FV}(N)$*;*
- *a λ-term M is called* closed *or* a combinator *if* $\mathrm{FV}(M) = \emptyset$*. The set of all combinators is denoted by Λ^o;*
- *the* α-conversion *identifies two λ-terms which only differ by the name of bound variables.*

Example 2.2.5.

- *In $\lambda x.yzx$ variables y and z are free, but x is bound;*
- *In $(\lambda x.yzx)(xy)$ variables y, z and x are free, since x is free in the subterm (xy);*
- $\lambda x.yzx =_\alpha \lambda k.yzk \neq_\alpha \lambda x.kzx$*;*
- $(\lambda x.yzx)(xy) =_\alpha (\lambda w.yzw)(xy)$ *only the bound x is renamed, the one in the subterm (xy) represents a different variable and is unchanged.*

Hereafter, λ-terms will be considered up to α-conversion, except when otherwise specified. Classical examples of combinators are given in Figure 2.1.

Definition 2.2.6 (Context)**.**

1. *A* (single-hole) context $C\llparenthesis - \rrparenthesis$ *is a λ-term containing an occurrence of an algebraic variable, called* hole *and denoted by $\llparenthesis - \rrparenthesis$. Formally, $C\llparenthesis - \rrparenthesis$ is generated by the grammar:*

$$C ::= \llparenthesis - \rrparenthesis \mid CM \mid MC \mid \lambda x.C \qquad \textit{(for } M \in \Lambda \textit{ and } x \in \mathbb{V}\textit{)}$$

2. *Given $M \in \Lambda$, we write $C(\!|M|\!)$ for the λ-term obtained by replacing M for the hole $(\!|-|\!)$ in $C(\!|-|\!)$, possibly with capture of free variables.*

3. *An* occurrence *of a subterm N in a λ-term M is identified by a context $C(\!|-|\!)$ such that $M = C(\!|N|\!)$.*

4. *Given* $\mathrm{R} \subseteq \Lambda^2$, *its* contextual closure R' *is the least relation containing* R *and satisfying:*
$$M \ \mathrm{R}' \ N \Rightarrow \forall C(\!|-|\!),\ C(\!|M|\!) \ \mathrm{R}' \ C(\!|N|\!).$$

By "possibly with capture of free variables" we mean that we did not use α-conversion before the replacement of the hole of $C(\!|-|\!)$ by a λ-term M. Some free variables of M may become bound in $C(\!|M|\!)$.

Remark 2.2.7. *In [Barendregt, 1984] a hole is represented by $[-]$, but here we use $(\!|-|\!)$ in order to avoid confusion with the bags $[v_1, \ldots, v_n]$ of resource calculus present in Chapters 4 and 5.*

Example 2.2.8.

- *$\lambda fz.f(gx) = C(\!|gx|\!)$ with $C(\!|-|\!) = \lambda fz.f(\!|-|\!)$ and gx is a subterm of $\lambda fz.f(gx)$ with an occurrence $\lambda fz.f(\!|-|\!)$;*
- *Let $C(\!|-|\!) = \lambda x.yz(\!|-|\!)$ and $M = xy$. In M, x is free, but not in $C(\!|M|\!) = \lambda x.yz(xy)$: it has been captured.*

The set Λ of λ-terms is endowed with notions of reduction turning the λ-calculus into a higher-order term rewriting system.

Definition 2.2.9. *The* β- and η-reductions *are defined as the contextual closures of the following relations:*

$$\begin{array}{lrcll} (\beta) & (\lambda x.M)N & \mapsto & M[N/x] & \\ (\eta) & \lambda x.Mx & \mapsto & M & (\textit{if } x \notin \mathrm{FV}(M)) \end{array}$$

where $M[N/x]$ denotes the capture-free substitution of N for all free occurrences of x in M. This is the λ-term obtained by substituting N for every free occurrence of x in M, subject to the renaming of bound variables in M to avoid capture of free variables in N.

More generally, given variables $x_1, \ldots, x_n$ and λ-terms $N_1, \ldots, N_n$, a substitution $\theta : \Lambda \to \Lambda$ is a finite map substituting N_i to x_i for $i = 1, \ldots, n$. Given a λ-term M, we write M^θ for $M[x_1/N_1] \cdots [x_n/N_n]$.

Example 2.2.10.

- *$(\lambda y.xyz(xz))[yz/x] = \lambda y.(xyz(xz))[wz/x] = \lambda y.wzyz(wzz)$;*
- *$(zzy(\lambda z.xz))[yx/z] = (yx)(yx)y(\lambda z.xz)$.*

The λ-term on the left-hand side of the arrow is called *redex*, the one on the right-hand side is its *reduct*.
For a relation R, a λ-term is in R-normal form if and only if it contains no R-redexes.

Example 2.2.11. *Using combinators defined in Figure 2.1.*

1. $\mathbf{I}x \to_\beta x$;

2. $\mathbf{I}(xy) \to_\beta xy$;

3. $\Omega \to_\beta \Omega$, *whence* Ω *is a looping combinator;*

4. $\mathbf{I}(\Delta(xx)) \to_\beta \Delta(xx) \to_\beta (xx)(xx)$;

5. *For all* λ*-terms* M*, we have* $\mathbf{Z}M =_\beta M(\lambda x.\mathbf{Z}Mx)$ *with* $x \notin \mathrm{FV}(M)$*;*

6. $\mathbf{K}^* =_\beta \mathbf{K}(\lambda y.\mathbf{K}^*y) =_\beta \lambda x_0x_1.\mathbf{K}^*x_1 =_\beta \lambda x_0x_1x_2.\mathbf{K}^*x_2 =_\beta \cdots =_\beta \lambda x_0 \ldots x_n.\mathbf{K}^*x_n$;

7. $\mathbf{ZB} =_\beta \mathbf{B}(\lambda z.\mathbf{ZB}z) =_\beta \lambda gx.(\lambda z.\mathbf{ZB}z)(gx) =_\beta \lambda gx.(\lambda fy.(\lambda z.\mathbf{ZB}z)(fy))(gx) \cdots$

Remark 2.2.12. *Looking at* λ*-terms as functions and taking their extensional definitions (only considering inputs and outputs), we may want to consider terms that reduce in the same way when applied to any argument as equal. This principle is called* extensionality *and is captured by the* η*-equivalence. Observe that, if* x *does not occur in* M*, then* $\lambda x.Mx$ *and* M *have the same behavior on any input, since* $(\lambda x.Mx)N \to_\beta MN$*.*

Theorem 2.2.13 ([Church and Rosser, 1936])**.** *The reductions* $\to_\beta$ *and* $\to_\eta$ *are confluent.*

In consequence the normal form of a λ-term, if any, is well defined. Moreover, $M =_\beta N$ (or $M =_\eta N$) holds, if and only if M and N have a common β-reduct (or η-reduct) Z: $M \twoheadrightarrow_\beta Z$ and $N \twoheadrightarrow_\beta Z$ (resp. $M \twoheadrightarrow_\eta Z$ and $N \twoheadrightarrow_\eta Z$) for some Z.

Lemma 2.2.14. *A* λ*-term* M *is in* β*-normal form if and only if* $M = \lambda\vec{x}.xM_1 \cdots M_m$ *with* $M_1, \ldots, M_m$ *being in* β*-normal form.*

Note that in the lemma above the number of abstractions as well as the number m can possibly be 0.

The λ-terms are classified into solvable/unsolvable, depending on their capability of interaction with the environment:

Definition 2.2.15. *A* λ*-term* M *is:*

- CbN-solvable *if there exist* $\vec{x} \in \mathbb{V}, \vec{N} \in \Lambda$ *such that* $(\lambda\vec{x}.M)\vec{N} \twoheadrightarrow_\beta \mathbf{I}$;

- CbN-unsolvable*, if it is not solvable.*

Example 2.2.16.

- *$x\Omega$ is CbN-solvable, with the use of a context $C(\!|-|\!) = (\lambda x.(\!|-|\!))(\lambda xy.y)$, indeed $(\lambda x.x\Omega)(\lambda xy.y) \to_\beta (\lambda xy.y)\Omega \to_\beta \mathbf{I}$;*
- *$\lambda x.\Omega$ is CbN-unsolvable since $\Omega \to_\beta \Omega$ and we cannot erase the subterm Ω without erasing the whole term.*

Historically solvable terms are considered to be the meaningful terms of λ-calculus, we can see them as the λ-terms that can produce information. Therefore characterising which specific λ-terms are solvable is a key question.

Head Normal Form. Here we introduce a method due to Wadsworth to characterise solvable λ-terms (see [Wadsworth, 1976]).

Definition 2.2.17 (Head normal form and head reduction)**.**

- *A* head normal form *(HNF) is a λ-term M, such that for some $M_1, \dots, M_m \in \Lambda$ and $\vec{x}, x \in \mathbb{V}$, $M = \lambda\vec{x}.xM_1 \cdots M_m$.*
- *A λ-term M* has a head normal form *if M reduces to N in a finite number of reductions and N is a head normal form.*
- *In a λ-term of the form $\lambda x_1 \dots x_n.(\lambda y.N)LM_1 \cdots M_m$ we call* head redex *the subterm $(\lambda y.N)L$.*
- *The* head reduction $\to_h$ *corresponds to the contraction of the head redex:*

$$\lambda x_1 \dots x_n.\underline{(\lambda y.N)M_0}M_1 \cdots M_m \to_h \lambda x_1 \dots x_n.\underline{N[M_0/y]}M_1 \cdots M_m.$$

Notice that any β-normal form is a head normal form.

Remark 2.2.18. *If M has a head normal form, then such a normal form can be reached by performing head reductions.*

Theorem 2.2.19 ([Wadsworth, 1976])**.** *A λ-term M is CbN-solvable if and only if M has a head normal form.*

2.3. Call-by-Value

Here we present the Call-by-Value paradigm, first originated in [Plotkin, 1975]. The fundamental point is to use only a subset of λ-terms, called *values*, as arguments of applications.

Definition 2.3.1. *We consider fixed an infinite countable set $\mathbb{V}$ of variables. The set Λ of* λ-terms *and the set Λ^V of* values *are defined inductively by:*

$$\begin{array}{llcl} (\Lambda) & M, N, L & ::= & V \mid MN \\ (\Lambda^V) & U, V & ::= & x \mid \lambda x.M \qquad (\textit{for } x \in \mathbb{V}) \end{array}$$

Variables and abstractions are special λ-terms, called values. They are not necessarily in normal form, since $\lambda x.M$ may contain redexes inside M.

As usual, we assume that application associates to the left and has higher precedence than abstraction (see Example 2.2.2). In the following, λ-terms are considered up to α-conversion, and we use the usual definition of free variables (see Definition 2.2.4).

We use the same contexts as in the Call-by-Name case (Definition 2.2.6(1)). And we call *head context* a context of the shape $(\lambda x_1 \ldots x_n.(\!|-|\!))V_1 \cdots V_m$, with $x_1, \ldots, x_n \in \mathbb{V}$ and $V_1, \ldots, V_m \in \Lambda^V$.

Classically the Call-by-Value λ-calculus λ_v is endowed with one standard notion of reduction: the β_{v}-reduction defined in [Plotkin, 1975].

Definition 2.3.2. *The* β_{v}-reduction *is the contextual closure (Definition 2.2.6*(4)*) of the following rule:*

$$(\beta_{\mathsf{v}}) \qquad (\lambda x.M)V \;\mapsto\; M[V/x] \qquad (\textit{for } V \in \Lambda^V)$$

Issue. Since the β_{v}-reduction only applies when arguments are values, some λ-terms can get stuck in premature normal forms. The λ-term $L = (\lambda y.\Delta)(xx)\Delta$ is a paradigmatic example of this kind of situation (see [Paolini and Ronchi Della Rocca, 1999] and [Accattoli and Guerrieri, 2017]). This term is a Call-by-Value normal form because the argument xx is not a value and blocks the evaluation, however one would expect M to behave as the divergent term $\Omega = \Delta\Delta \to_{\beta_{\mathsf{v}}} \Omega$, similarly as in Call-by-Name where $L \to_\beta \Omega$. The β_{v}-reduction has been shown, in [Ronchi Della Rocca and Paolini, 2004], too weak to characterise some important properties such as solvability and potential valuability (Definition 2.3.8).

Inspired by [Herbelin and Zimmermann, 2009] and [Accattoli and Kesner, 2012] - which both present alternative versions of Call-by-Value λ-calculus - Accattoli and Paolini introduced the *value-substitution calculus* λ_{sub} in [Accattoli and Paolini, 2012]. The major idea is to add a new term representing an explicit substitution $M[N/x]$ and delay the effective substitution by means of two reduction rules. This setting has allowed to internally characterise Call-by-Value solvability for the first time. Solvable terms are the λ-terms having normal forms for a reduction called stratified-weak-reduction.

Another equivalent way of fixing the Call-by-Value λ-calculus was introduced in [Carraro and Guerrieri, 2014] where the authors, inspired by Regnier's work in the Call-by-Name setting ([Regnier, 1994]), defined permutations rules, naturally arising from the translation of λ-terms into Linear Logic proof-nets. Those rules are called σ-rules. Using them, the aforementioned λ-term L can be rewrite in $(\lambda y.\Delta\Delta)(xx)$, which in turn rewrites to itself, thus giving rise to the desired infinite reduction sequence.
In [Guerrieri et al., 2017], Guerrieri, Paolini and Ronchi Della Rocca showed that this extended calculus λ_v^σ still enjoys nice properties like confluence, and that adding the σ-rules preserves the operational semantics of Plotkin's Call-by-Value λ-calculus as well as the

observational equivalence (Definition 2.3.12). As a consequence, the calculus λ_v^σ can be used as a tool for studying the original calculus λ_v.

In the following of this work, when reasoning about Call-by-Value λ-calculus, we will consider λ_v^σ.

Definition 2.3.3. *The σ-reductions $\to_{\sigma_1}$ and $\to_{\sigma_3}$ are the contextual closures (Definition 2.2.6(4)) of the following rules:*

$$\begin{array}{lrcll} (\sigma_1) & (\lambda x.M)NL & \mapsto & (\lambda x.ML)N & \textit{with } x \notin \mathrm{FV}(L) \\ (\sigma_3) & V((\lambda x.M)N) & \mapsto & (\lambda x.VM)N & \textit{with } x \notin \mathrm{FV}(V) \end{array}$$

We also set $\to_\sigma = \to_{\sigma_1} \cup \to_{\sigma_3}$ and $\to_v = \to_{\beta_\mathsf{v}} \cup \to_\sigma$.

Notice that the condition for contracting a σ_1- (resp. σ_3-) redex can always be satisfied by performing appropriate α-conversions. In the following we will always consider those conditions satisfied.

Lemma 2.3.4. *The set Λ^V is closed under substitution of values for free variables. Namely $U, V \in \Lambda^V$ and $x \in \mathbb{V}$ entail $V[U/x] \in \Lambda^V$.*

Proof. Simple by induction on V. In particular, if $V = \lambda y.M$ then $V[U/x] = \lambda y.(M[U/x]) = \lambda y.M' \in \Lambda^V$ for $M' = M[U/x]$. □

Example 2.3.5. *Using combinators defined in Figure 2.1. And referring to Definition 2.1.1 for $=_v$.*

1. $\mathbf{I}x \to_{\beta_\mathsf{v}} x$;

2. $\mathbf{I}(xy)$ *is a* V*-normal form;*

3. $\Omega \to_{\beta_\mathsf{v}} \Omega$, *whence* Ω *is a looping combinator;*

4. $\mathbf{I}(\Delta(xx))$ *is a* β_v*-nf, but* $\mathbf{I}(\Delta(xx)) \to_{\sigma_3} (\lambda z.\mathbf{I}(zz))(xx)$;

5. *For all values* V, *we have* $\mathbf{Z}V =_v V(\lambda x.\mathbf{Z}Vx)$ *with* $x \notin \mathrm{FV}(V)$;

6. $\begin{array}{ll} \mathbf{K}^* & = \mathbf{ZK} =_v \mathbf{K}(\lambda y.\mathbf{K}^*y) =_v \lambda x_0x_1.\mathbf{K}^*x_1 =_v \lambda x_0x_1x_2.\mathbf{K}^*x_2 =_v \cdots \\ & =_v \lambda x_0 \ldots x_n.\mathbf{K}^*x_n; \end{array}$

7. *Let* $\Xi = \mathbf{Z}N$ *for* $N = \lambda f.(\lambda y_1.f\mathbf{I})(zz)$, *then we have:*

$$\begin{array}{lll} \Xi =_v & N(\lambda w.\Xi w) & \textit{by } \mathbf{Z}V =_v V(\lambda w.\mathbf{Z}Vw) \\ =_v & (\lambda y_1.(\lambda w.\Xi w)\mathbf{I})(zz) & \textit{by } (\beta_\mathsf{v}) \\ =_v & (\lambda y_1.\Xi\mathbf{I})(zz) & \textit{by } (\beta_\mathsf{v}) \\ =_v & (\lambda y_1.((\lambda y_2.\Xi\mathbf{I})(zz))\mathbf{I})(zz) & \textit{by } (\beta_\mathsf{v}) \\ =_v & (\lambda y_1.((\lambda y_2.\Xi\mathbf{II})(zz)))(zz) & \textit{by } (\sigma_1) \\ =_v & (\lambda y_1.((\lambda y_2.((\lambda y_3.\Xi\mathbf{III})(zz)))(zz)))(zz) & = \cdots; \end{array}$$

8. $\mathbf{ZB} =_v \mathbf{B}(\lambda z.\mathbf{ZB}z) =_v \lambda gx.(\lambda z.\mathbf{ZB}z)(gx) =_v \lambda gx.(\lambda fy.(\lambda z.\mathbf{ZB}z)(fy))(gx) \cdots$.

One can check the differences with the β-reduction in Call-by-Name: Example 2.2.11.

The next lemma was already used implicitly in [Guerrieri et al., 2017].

Lemma 2.3.6. *A λ-term is in v-normal form if and only if it is a G-term defined by:*

$$\begin{array}{lcll} G & ::= & E \mid F & \\ F & ::= & x \mid \lambda x.G \mid xFG_1 \cdots G_k & \textit{(for } k \geq 0\textit{)} \\ E & ::= & (\lambda x.G)(yHG_1 \cdots G_k) & \textit{(for } k \geq 0\textit{)} \end{array}$$

Proof.

($\Rightarrow$) Assume that M is in v-nf and proceed by structural induction. Observe that every λ-term M can be uniquely written as $\lambda x_1 \ldots x_m.M'N_1 \cdots N_n$ where $m, n \geq 0$ and either $M' = x$ or $M' = (\lambda x.L_1)L_2$. Moreover, the λ-terms $M', N_1, \ldots, N_n$ must be in v-nf's since M is v-nf.

- If $m > 0$ then M is of the form $\lambda x.L_1$ with L_1 in v-nf and the result follows from the induction hypothesis.
- Otherwise we assume $m = 0$ and split into cases depending on M':
    - $M' = x$ for some $x \in \mathbb{V}$:
        - if $n = 0$ then we are done since x is an F-term.
        - if $n > 0$ then $M = xN_1 \cdots N_n$ where all the N_i's are G-terms by induction hypothesis. Moreover, N_1 cannot be an E-term for otherwise M would have a σ_3-redex. Whence, N_1 must be an F-term and M is of the form $xFG_1 \cdots G_k$ for $k = n - 1$.
    - $M' = (\lambda x.L_1)L_2$ for some variable x and λ-terms L_1, L_2 in v-nf. In this case we must have $n = 0$ because M cannot have a σ_1-redex. By induction hypothesis, L_1, L_2 are G-terms, but L_2 cannot be an E-term or a value for otherwise M would have a σ_3- or a β_{v}-redex, respectively. The only possibility for the shape of L_2 is $yFG_1 \cdots G_k$, whence M must be an R-term.

($\Leftarrow$) By induction on the grammar generating M:

- $M = x$ is obvious.
- $M = \lambda x.G$ by induction hypothesis on G.
- $M = xFG_1 \cdots G_k$ could have a σ_3-redex if $H = (\lambda y.L_1)L_2$, but this is impossible by definition of an H-term. As $H, G_1, \ldots, G_k$ are in v-nf by induction hypothesis, so must be M.
- $M = (\lambda x.G)(yFG_1 \cdots G_k)$ where $G, F, G_1, \ldots, G_k$ are in v-nf by induction hypothesis. In the previous item we established that $yFG_1 \cdots G_k$ is in v-nf. Thus, M could only have a β_{v}-redex if $yFG_1 \cdots G_k \in \Lambda^V$, but this is not the case by definition of Λ^V. □

Property 2.3.7 (Properties of reductions [Plotkin, 1975, Carraro and Guerrieri, 2014])**.**

1. *The reduction $\to_\sigma$ is strongly normalising.*
2. *The reduction $\to_{\beta_\mathsf{v}}$ and $\to_v$ are confluent.*

Proof.

1. The proof comes from [Carraro and Guerrieri, 2014]: define two sizes $s(M)$ and $\#M$ by induction on the λ-term M:

$$\begin{array}{rcl} \mathrm{s(x)} & = & 2 \\ \mathrm{s(\lambda x.M)} & = & \mathrm{s(M) + 1} \\ \mathrm{s(MN)} & = & \mathrm{s(M) + s(N)} \end{array} \qquad \begin{array}{rcl} \mathrm{\#x} & = & 1 \\ \mathrm{\#\lambda x.M} & = & \mathrm{\#\ M + s(M)} \\ \mathrm{\#\ MN} & = & \mathrm{\#\ M + \#\ N + 2s(M)s(N)\ \text{-}\ 1} \end{array}$$

   Observe that if $N \to_\sigma N'$ then $s(N) = s(N')$ and $\#N > \#N'$.

2. β_v is known to be confluent from [Plotkin, 1975].

   The proof for $\to_v$ comes from [Carraro and Guerrieri, 2014]: they prove the local confluence for σ by easy induction. Using Lemma 2.1.4 we obtain that σ is confluent. It is easy to see by induction that σ and β_v commute and conclude using Lemma 2.1.5. □

Since $\to_v$ is confluent, the v-normal form of a λ-term, if any, is well defined.

There are some properties of λ-terms that we aim at characterising:

Definition 2.3.8. *A λ-term M is called:*

- valuable *if it reduces to a value, namely $M \twoheadrightarrow_v V$ for some $V \in \Lambda^V$;*
- potentially valuable *if there exists a substitution $\vartheta : \mathbb{V} \to \Lambda^V$ such that M^ϑ is valuable, or equivalently there exists a head context $C(\!|-|\!) = (\lambda x_1 \ldots x_n.(\!|-|\!))V_1 \cdots V_m$, where* $\mathrm{FV}(M) = \{x_1, \ldots, x_n\}$*, such that $C(\!|M|\!)$ is valuable;*
- CbV-solvable *if there exist sequences $\vec{x} \in \mathbb{V}, \vec{V} \in \Lambda^V$ such that $(\lambda\vec{x}.M)\vec{V} \twoheadrightarrow_v \mathbf{I}$ (similar to Call-by-Name case: Definition 2.2.15);*
- CbV-unsolvable*, if it is not solvable.*

CbV-solvability is also stronger than potential valuability, but is orthogonal to valuability.

Example 2.3.9. *Some discriminating examples (using combinators defined in Figure 2.1):*

- $\mathbf{I}, \mathbf{\Delta}, \mathbf{P}_n, \mathbf{\Delta}(\mathbf{II}), \mathbf{P}_1(\lambda x.\mathbf{\Omega})$ *are (potentially) valuable and CbV-solvable;*

- $\mathbf{P}_1 x(\lambda x.\mathbf{\Omega}), xy\mathbf{I\Delta}$ *and* $\mathbf{\Delta}(xy)$ *are not valuable, but potentially valuable and CbV-solvable;*
- $\lambda x.\mathbf{\Omega}, \mathbf{ZB}$ *and* $\mathbf{K}^\star$ *are valuable, but CbV-unsolvable. The* λ*-term* $\mathbf{K}^\star$ *is called an ogre because of its capability of eating any* $\vec{V}$ *while remaining valuable:* $\mathbf{K}^\star\vec{V} \twoheadrightarrow_{\beta_\mathsf{v}} \ldots \twoheadrightarrow_{\beta_\mathsf{v}} \lambda x_1 \ldots x_n.\mathbf{K}^\star\vec{V}$;
- $\mathbf{\Omega}, \mathbf{\Omega}(xy), (\lambda y.\mathbf{\Delta})(x\mathbf{I})\mathbf{\Delta}, \mathbf{I\Omega}, \mathbf{ZI}$ *are not potentially valuable nor CbV-unsolvable. The same holds for* $\mathbf{Y}M$*, where* $\mathbf{Y}$ *is a Call-by-Name fixed point combinator.*

Remark 2.3.10. *The original definitions of valuability, potential valuability and CbV-solvability are given in terms of* β_v*-reduction (see respectively [Plotkin, 1975] and [Ronchi Della Rocca and Paolini, 2004]). In the articles [Carraro and Guerrieri, 2014] and [Guerrieri et al., 2017], it is shown that extending the original Call-by-Value* λ*-calculus [Plotkin, 1975] with the* σ*-rules does not alter those operational properties. In particular, for all* λ*-terms* M*, we have that* $M \twoheadrightarrow_{\beta_\mathsf{v}} \mathbf{I}$ *holds exactly when* $M \twoheadrightarrow_v \mathbf{I}$ *does.*

Property 2.3.11. *If* $M = (\lambda x_1 \ldots x_k.L)N_1 \cdots N_n \twoheadrightarrow_v \mathbf{I}$ *then each* N_i *is valuable, say* $N_i \twoheadrightarrow_v V_i$*. Moreover, we must have* $k \leq n+1$*.*

Proof. By the above remark, $M \twoheadrightarrow_v \mathbf{I}$ entails $M \twoheadrightarrow_{\beta_\mathsf{v}} \mathbf{I}$. The N_is will give arguments for the β_v-reductions, they obviously need to be valuable.

If $k > n+1$ then $M \twoheadrightarrow_{\beta_\mathsf{v}} (\lambda\vec{x}.L)\vec{V} \twoheadrightarrow_{\beta_\mathsf{v}} \lambda x_{n+1} \ldots x_k.L' \neq_{\beta_\mathsf{v}} \mathbf{I}$. □

In general M valuable entails M potentially valuable and for $M \in \Lambda^o$ the two notions coincide.

Observational Equivalence. In order to capture the equivalence of behaviour of λ-terms, [Plotkin, 1975] introduced an *observational equivalence* analogous to the following one:

Definition 2.3.12. *The CbV-observational equivalence* $\equiv$ *is defined by (for* $M, N \in \Lambda$*):*

$$M \equiv N \qquad \iff \qquad \begin{array}{c} \forall C(\!|-|\!),\ C(\!|M|\!), C(\!|N|\!) \in \Lambda^o \\ [\ \exists V \in \Lambda^V,\ C(\!|M|\!) \twoheadrightarrow_{\beta_\mathsf{v}} V \iff \exists U \in \Lambda^V,\ C(\!|N|\!) \twoheadrightarrow_{\beta_\mathsf{v}} U\ ] \end{array}$$

Example 2.3.13. $\mathbf{I} \equiv \lambda xy.xy$ *and* $\Xi \equiv \Omega$ *(see Example 2.3.5), but* $\Omega \not\equiv \lambda x.\Omega$*.*

Remark 2.3.14. *It is well known that, in order to check whether* $M \equiv N$ *holds, it is enough to consider head contexts (cf. [Ong, 1997, Paolini, 2008]). In other words,* $M \not\equiv N$ *if and only if there exists a head context* $C(\!|-|\!)$ *such that* $C(\!|M|\!)$ *is valuable, while* $C(\!|N|\!)$ *is not.*

3. Böhm Trees

In the first part of this chapter, 3.1, we recall the definition of classical Call-by-Name Böhm trees originated from [Barendregt, 1977]. There are several equivalent ways of defining Böhm trees, the most famous one is coinductive and from [Lassen, 1999], while the original one in Barendregt's book exploits the principle of "effective Böhm-like trees" which is not easy to handle in practice. The definition given in [Amadio and Curien, 1998, Def. 2.3.3] does not require coinductive techniques: the idea is to first define the set $\mathcal{A}_\beta(M)$ of approximants of a λ-term M, then show that it is directed with respect some preorder $\sqsubseteq_\beta$ and, finally, define the Böhm tree of M as the supremum of $\mathcal{A}_\beta(M)$. We present both the coinductive definition in Subsection 3.1.1 and the one using approximants in Subsection 3.1.2.

In the second part of this chapter, we consider Böhm trees in a Call-by-Value setting. In Call-by-Value the theory of program approximation is still unsatisfactory and constitutes an ongoing line of research, explored in [Ehrhard, 2012, Carraro and Guerrieri, 2014, Manzonetto et al., 2019b]. To our knowledge, no satisfying theory of Böhm tree has ever been developed for this reduction strategy before.

However, some work has been done on theories of program approximation arising from denotational models. In [Ronchi Della Rocca and Paolini, 2004], the authors study a filter model of Call-by-Value λ-calculus and, in order to prove an Approximation Theorem, define sets of *upper* and *lower* approximants of a λ-term. Sadly, the authors admit that this approach is not satisfying, because it leads to an "over" (resp. "under") approximation of the behaviour of the λ-term.

Our contribution, which is the focus of this chapter, consists in producing an interesting definition for Call-by-Value Böhm trees. This result has already been published in [Kerinec et al., 2020]. We follow the methodology of [Amadio and Curien, 1998]: we provide an appropriate notion of approximants, and obtain the Böhm trees as the "limit" of those approximants. Finding a Call-by-Value coinductive definition of Böhm trees remains an open problem.

In Section 3.2, we introduce a constant $\perp$ representing an *undefined value*, and approximants correspond to λ-terms possibly containing $\perp$ and in normal form w.r.t. the reduction rules of λ_v^σ (Definition 3.2.3). In this context we define a preorder $\sqsubseteq_v$ between approximants which is generated by $\perp \sqsubseteq_v V$, for all approximated values V.

We associate to every λ-term M the set $\mathcal{A}_v(M)$ of its approximants and verify that they enjoy the following properties:

- the "external shape" of an approximant of M is stable by reduction (Lemma 3.2.11);

- two interconvertible λ-terms share the same set of approximants (Lemma 3.2.12);
- the set of approximants of M is directed (Lemma 3.2.13).

Once this preliminary work is accomplished, it is possible to define the Böhm tree of M as the supremum of $\mathcal{A}_v(M)$ (Definition 3.2.14), the result being a possibly infinite labelled tree $\mathrm{BT}_v(M)$, as expected.

In Subsection 3.2.2 we define the Call-by-Value "Böhm-like" trees as those labelled trees that can be obtained by arbitrary superpositions of compatible approximants. The Böhm-like trees corresponding to Call-by-Value Böhm trees of λ-terms have specific properties, that are due to the fact that λ-calculus constitutes a model of computation. Indeed, since every λ-term M is finite, $\mathrm{BT}_v(M)$ can only contain a finite number of free variables and, since M represents a program, the tree $\mathrm{BT}_v(M)$ must be computable. In Theorem 3.2.23 we demonstrate that these conditions are actually sufficient for a Böhm-like tree to actually be the Böhm tree of some λ-term.

3.1. Call-by-Name

Böhm trees in the Call-by-Name framework, introduced in [Barendregt, 1977], have a very long and rich history. We recall here some basic definitions and theorems.

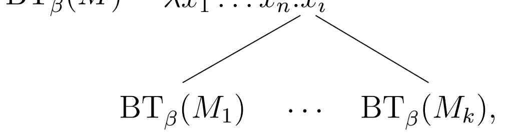
3.1.1. Coinductive Definition of Böhm Trees

In this section we will present the coinductive definition of Böhm trees, which was formalised in [Lassen, 1999].

We introduce a new symbol: $\perp$. It will be used as a subterm in order to represent indefiniteness.

Definition 3.1.1. *The* Böhm tree *of a λ-term M is defined coinductively, in term of head reduction (see Definition 2.2.17), as follows:*

- *if $M \twoheadrightarrow_h \lambda x_1 \dots x_n.x_i M_1 \cdots M_k$ (for $n, k \geq 0$), then*


$\mathrm{BT}_\beta(M) = \lambda x_1 \dots x_n.x_i$
$\mathrm{BT}_\beta(M_1)$
$\cdots$
$\mathrm{BT}_\beta(M_k)$,

- *otherwise*

$$\mathrm{BT}_\beta(M) = \perp.$$

Since having a head normal form corresponds to being CbN-solvable (Theorem 2.2.19), there is a very strong connection between Böhm trees and CbN-solvability. In particular the CbN-unsolvable λ-terms are exactly the λ-terms with Böhm trees equal to $\perp$.

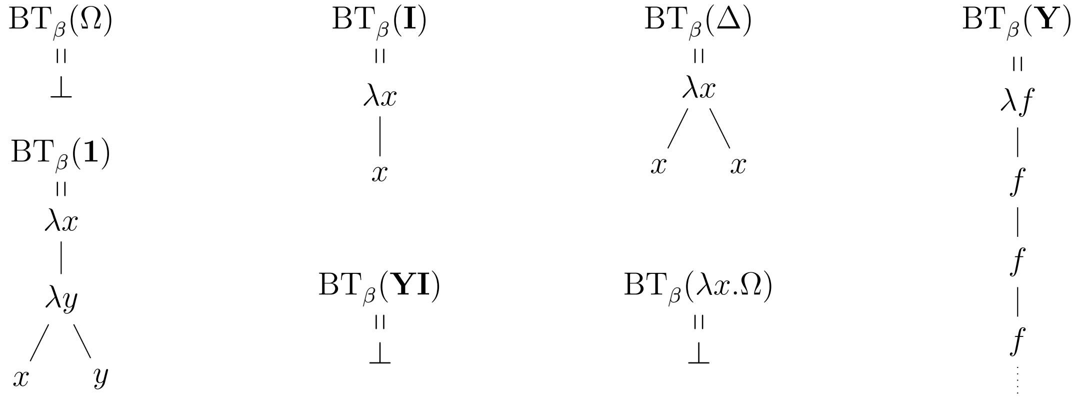

$\mathrm{BT}_\beta(\Omega) = \perp$
$\mathrm{BT}_\beta(\mathbf{1}) = \lambda x - \lambda y - x \; y$
$\mathrm{BT}_\beta(\mathbf{I}) = \lambda x - x$
$\mathrm{BT}_\beta(\mathbf{YI}) = \perp$
$\mathrm{BT}_\beta(\Delta) = \lambda x - x \; x$
$\mathrm{BT}_\beta(\lambda x.\Omega) = \perp$
$\mathrm{BT}_\beta(\mathbf{Y}) = \lambda f - f - f - f$

Figure 3.1.: Examples of Call-by-Name Böhm trees.

Notation 3.1.2. *For simplicity reason we will often omit the tree representation and write* $\lambda x_1 \dots x_n.x_i \mathrm{BT}_\beta(M_1) \cdots \mathrm{BT}_\beta(M_k)$ *for* $\mathrm{BT}_\beta(M)$.

Example 3.1.3. *The following are examples of Böhm trees (using combinators defined in Figure 2.1):*

1. $\mathrm{BT}_\beta(\mathbf{I}) = \lambda x.x$;
2. $\mathrm{BT}_\beta(\mathbf{1}) = \lambda xy.xy$;
3. $\mathrm{BT}_\beta(\Delta) = \lambda x.xx$;
4. $\mathrm{BT}_\beta(\Omega) = \perp$ *since* Ω *is unsolvable;*
5. $\mathrm{BT}_\beta(\lambda x.\Omega) = \perp$;
6. $\mathrm{BT}_\beta(\mathbf{YI}) = \perp$ *since* $\mathbf{YI}$ *is unsolvable;*
7. $\mathrm{BT}_\beta(\mathbf{Y}) = \lambda f.f(f(f(f(f(\cdots)))))$, *each step of reduction produce a new* f *which is stable in following reductions.*

You can see them as actual trees in Figure 3.1.

Remark 3.1.4. *More generally, if* M *is in* β*-nf then* $\mathrm{BT}_\beta(M) = M$.

Since $\mathrm{BT}_\beta(M)$ is defined coinductively, so is the equality between Böhm trees:

$$\mathrm{BT}_\beta(M_1) = \mathrm{BT}_\beta(M_2) \quad \textit{iff} \quad \begin{cases} M_1, M_2 \textit{ have both no HNF} \\ \qquad\qquad \textit{or} \\ \textit{for } i = 1,2 : M_i \twoheadrightarrow_h \lambda\vec{x}.yN_1^i \cdots N_k^i \\ \qquad\qquad \textit{and } \forall j \in 1, \dots k, \mathrm{BT}_\beta(N_j^1) = \mathrm{BT}_\beta(N_j^2). \end{cases}$$

3.1.2. Theory of Program Approximation

At the time when the notion of Böhm tree was introduced in [Barendregt, 1977] researchers also proposed an inductive theory of program approximation based on Scott-continuity and finite trees. The possibly infinite behaviour of a λ-term, represented by its Böhm tree, is then retrieved by performing a "limit" of its finite approximants that are potentially infinitely many.

Definition 3.1.5.

- *The set $\Lambda_\perp$ of* $\lambda_\perp$-terms *over $\mathbb{V}$ is inductively defined by the grammar:*

$$(\Lambda_\perp) \qquad M,N,L ::= \ \perp \ \mid \ x \ \mid \ MN \mid \ \lambda x.M \qquad \textit{(for } x \in \mathbb{V}\textit{)}$$

- *The* $\perp$-contexts $C(\!|-|\!)$ *are an extended version of the previous contexts (Definition 2.2.6) using $\perp$:*

$$C ::= (\!|-|\!) \mid CM \mid MC \mid \lambda x.C \qquad \textit{(for } M \in \Lambda_\perp \textit{ and } x \in \mathbb{V}\textit{)}$$

Definition 3.1.6. *Let $\sqsubseteq_\beta \subseteq \Lambda_\perp \times \Lambda_\perp$ denotes the least contextual closed preorder (see Definition 2.2.6*(4)*) generated by setting:*

$$\textit{for all } M \in \Lambda_\perp, \quad \perp \sqsubseteq_\beta M.$$

We can extend the β-reduction to this setting in the obvious way. And the $\lambda_\perp$-terms are endowed with the reduction $\to_{\beta\perp}$, which is the β-reduction extended with:

$$\begin{array}{lcl} \lambda x.\perp & \mapsto_\perp & \perp \\ \perp M_1 \cdots M_n & \mapsto_\perp & \perp \qquad \text{(for } n > 0\text{)} \end{array}$$

Definition 3.1.7. *The set $\mathcal{A}_\beta \subseteq \Lambda_\perp$ of* finite approximants *is defined by:*

$$(\mathcal{A}_\beta) \qquad A ::= \ \perp \ \mid \ \lambda x_1 \ldots x_n.yA_1 \cdots A_k \quad \textit{(for } n,k \geq 0 \textit{ and } x_1,\ldots x_n, y \in \mathbb{V}\textit{)}$$

We said that two approximants $A_1, A_2 \in \mathcal{A}_\beta$ are *compatible* if there exists $A \in \mathcal{A}_\beta$ such that $A_1 \sqsubseteq_\beta A \sqsupseteq_\beta A_2$.

Definition 3.1.8. *Given a λ-term M, the set $\mathcal{A}_\beta(M)$ of* finite approximants of M *is defined as follows:*

$$\mathcal{A}_\beta(M) = \{A \in \mathcal{A}_\beta \mid \exists N \in \Lambda\,,\ M \twoheadrightarrow_\beta N \textit{ and } A \sqsubseteq_\beta N\}.$$

Intuitively, the finite approximants of a λ-term M are obtained by cutting its Böhm tree into finite pieces, replacing the removed subtrees with $\perp$.

Example 3.1.9. *See Figure 2.1 for a definition of the useful combinators:*

1. $\mathcal{A}_\beta(\mathbf{I}) = \{\bot, \lambda x.x\}$;
2. $\mathcal{A}_\beta(\mathbf{1}) = \{\bot, \lambda xy.x\bot, \lambda xy.xy\}$;
3. $\mathcal{A}_\beta(\Omega) = \mathcal{A}_\beta(\mathbf{YI}) = \{\bot\}$;
4. $\mathcal{A}_\beta(\lambda x.x\Omega) = \{\bot, \lambda x.x\bot\} = \mathcal{A}_\beta(\lambda x.x(\mathbf{YI}))$;
5. $\mathcal{A}_\beta(\mathbf{Y}) = \{\bot\} \cup \{\lambda f.f^n(\bot) \mid n > 0\}$.

The following properties are well established (see [Amadio and Curien, 1998]).

Definition 3.1.10. *A set is an ideal if it is non-empty, downward closed and directed.*

Lemma 3.1.11 ([Amadio and Curien, 1998])**.**

1. *$M \in \Lambda_\bot$ is in $\beta\bot$-normal form if and only if $M \in \mathcal{A}_\beta$.*
2. *For $M \in \Lambda$, the set $\mathcal{A}_\beta(M)$ is an* ideal *and admits a supremum.*

The (syntactic) Approximation Theorem below shows that infinite Böhm trees can be recovered by taking the supremum of their finite approximants.

Theorem 3.1.12 (Approximation Theorem, [Amadio and Curien, 1998])**.** *For any λ-term M:*

$$\mathrm{BT}_\beta(M) = \bigsqcup \mathcal{A}_\beta(M)$$

Such a supremum always exists by Lemma 3.1.11(2)*.*

We deduce that for $M, N \in \Lambda$, $\mathrm{BT}_\beta(M) = \mathrm{BT}_\beta(N) \Leftrightarrow \mathcal{A}_\beta(M) = \mathcal{A}_\beta(N)$.

3.2. Call-by-Value

In this section we introduce an original definition for Call-by-Value Böhm trees.

3.2.1. Theory of Program Approximation

We will develop a theory of approximants similar at the one in Subsection 3.1.2 but in the Call-by-Value framework.

Similarly to the Call-by-Name case, we introduce $\bot$. However here $\bot$ is a constant representing an undefined value and not anymore a general λ-term.

Definition 3.2.1.

- *Let $\Lambda_\bot$ be the set of λ-terms extended with $\bot$, and $\Lambda_\bot^V$ the set of values extended with $\bot$:*

$$\begin{array}{llcll} (\Lambda_\bot) & M, N & ::= & V \mid MN & \\ (\Lambda_\bot^V) & U, V & ::= & \bot \mid x \mid \lambda x.M & \quad (\textit{for } x \in \mathbb{V}) \end{array}$$

- *The* $\perp$-contexts $C(\!|-|\!)$ *are an extended version of the previous contexts (Definition 2.2.6) using* $\perp$*:*

$$C ::= (\!|-|\!) \mid CM \mid MC \mid \lambda x.C \qquad (\textit{for } M \in \Lambda_\perp \textit{ and } x \in \mathbb{V})$$

They are the same as in the Call-by-Name case.

Definition 3.2.2. *The reduction* $\to_v$ *(from Definitions 2.3.2 and 2.3.3) generalises to terms in* $\Lambda_\perp$ *in the obvious way, with* $V \in \Lambda_\perp^V, M, N, L \in \Lambda_\perp$*:*

$$\begin{array}{llcll} (\beta_v) & (\lambda x.M)V & \mapsto & M[V/x] & \\ (\sigma_1) & (\lambda x.M)NL & \mapsto & (\lambda x.ML)N & \textit{with } x \notin \mathrm{FV}(L) \\ (\sigma_3) & V((\lambda x.M)N) & \mapsto & (\lambda x.VM)N & \textit{with } x \notin \mathrm{FV}(V) \end{array}$$

The *set of free variables* $\mathrm{FV}(-)$ is extended to approximants by setting $\mathrm{FV}(\perp) = \emptyset$.

Definition 3.2.3. *The set of* approximants *contains the terms* $A \in \Lambda_\perp$ *generated by the grammar:*

$$\begin{array}{llcll} (\mathcal{A}_v) & A & ::= & H \mid R & \\ & H & ::= & x \mid \lambda x.A \mid \perp \mid xHA_1 \cdots A_k & (\textit{for } k \geq 0 \textit{ and } x \in \mathbb{V}) \\ & R & ::= & (\lambda x.A)(yHA_1 \cdots A_k) & (\textit{for } k \geq 0 \textit{ and } y, x \in \mathbb{V}) \end{array}$$

Approximants of shape H *are called* head approximants *as they remind those used for building Call-by-Name Böhm trees, while approximants of shape* R *are called* redex-like *because they look like a* β*-redex. Let* $\mathcal{H}$ *(resp.* $\mathcal{R}$*) be the set of all head (resp. redex-like) approximants.*

Definition 3.2.4. *Let* $\sqsubseteq_v \subseteq \Lambda_\perp \times \Lambda_\perp$ *be the least contextual closed preorder on* $\Lambda_\perp$ *(see Definition 2.2.6(4)) generated by setting:*

$$\forall V \in \Lambda_\perp^V, \qquad \perp \sqsubseteq_v V.$$

Example 3.2.5.

- $\lambda x.xy\perp \sqsubseteq_v \lambda x.xyx$*;*
- $\lambda x.xy\perp \not\sqsubseteq_v \lambda x.x\perp x$ *and* $\lambda x.x\perp x \not\sqsubseteq_v \lambda x.xy\perp$*.*

Remark 3.2.6. *Notice that, contrary to the Call-by-Name case,* $\perp$ *is only used to approximate values, not general* λ*-terms like* Ω*.*

Definition 3.2.7. *Given* $M \in \Lambda$*,* the set of approximants of M *is defined as follows:*

$$\mathcal{A}_v(M) = \{A \in \mathcal{A}_v \mid \exists N \in \Lambda, M \twoheadrightarrow_v N \textit{ and } A \sqsubseteq_v N\}.$$

Example 3.2.8. *See Figure 2.1 for a definition of useful combinators:*

1. $\mathcal{A}_v(\mathbf{I}) = \{\bot, \lambda x.\bot, \lambda x.x\}$ *since both* $\lambda x.x$ *and* x *are values;*

2. $\mathcal{A}_v(\Omega) = \mathcal{A}_v(\Xi) = \emptyset$ *since those terms never reduced to* λ*-terms that can be compared to an approximant;*

3. $\mathcal{A}_v(\lambda x.\Omega) = \{\bot\}$ *since the term is a value we can compared it to* $\bot$*, but due to* $\mathcal{A}_v(\Omega) = \emptyset$ *we cannot have any other approximant;*

4. $\mathcal{A}_v(\mathbf{I}(\Delta(xx))) = \{(\lambda z.(\lambda y.Y)(zZ))(xX) \mid Y \in \{y, \bot\} \wedge Z \in \{z, \bot\} \wedge X \in \{x, \bot\}\}$. *Notice that neither* $(\lambda z.\bot)(xx)$ *nor* $(\lambda z.\bot)(x\bot)$ *belong to this set, since* $\bot \not\sqsubseteq_v \mathbf{I}(zz)$*;*

5. $$\begin{aligned} \mathcal{A}_v(\mathbf{Z}) = {} & \textstyle\bigcup_{n\in\mathbb{N}}\{\lambda f.f(\lambda z_0.f(\lambda z_1.f \cdots (\lambda z_n.f\bot\, Z_n)\cdots Z_1)Z_0) \mid \forall i, Z_i \in \{z_i, \bot\}\} \\ & \cup\{\bot\}; \end{aligned}$$

6. $\mathcal{A}_v(\mathbf{K}^*) = \{\lambda x_1 \ldots x_n.\bot \mid n \geq\ 0\}$*;*

7. $$\begin{aligned} \mathcal{A}_v(\mathbf{ZB}) = {} & \{\bot, \lambda f_0.\bot\} \\ & \cup\{\lambda f_0 x_0.(\cdots(\lambda f_{n-1}x_{n-1}.(\lambda f_n.\bot)(f_{n-1}X_{n-1}))\cdots)(f_0 X_0) \\ & \qquad\qquad \mid n > 0, \forall i\,,\ X_i \in \{x_i, \bot\}\}. \end{aligned}$$

Lemma 3.2.9. *A* $\Lambda_\bot$*-term* M *is in* v*-normal form if and only if* $M \in \mathcal{A}_v$*.*

Proof. Proof analogous to the one of Lemma 2.3.6. □

We will show that the external shape of an approximant is fixed under reduction. For instance, if $A = (\lambda x.A_0)(yHA_1 \cdots A_k) \sqsubseteq_v M$ then all approximants $A' \in \mathcal{A}_v(M)$ have shape $(\lambda x.A'_0)(yH'A'_1 \cdots A'_k)$ for some $H', A'_0, \ldots, A'_k \in \mathcal{A}_v$.

Lemma 3.2.10. *Let* $C\llparenthesis - \rrparenthesis$ *be a (single-hole)* $\bot$*-context and* $V \in \Lambda^V$*. Then* $C\llparenthesis \bot \rrparenthesis \in \mathcal{A}_v$ *and* $C\llparenthesis V \rrparenthesis \to_v N$ *entails that there exists a value* V' *such that* $V \twoheadrightarrow_v V'$ *and* $N = C\llparenthesis V' \rrparenthesis$*.*

Proof. Let $A = C\llparenthesis \bot \rrparenthesis \in \mathcal{A}_v$. By Lemma 3.2.9, A cannot have any v-redex. Clearly, substituting V for an occurrence of $\bot$ in A does not create any new β_v-redex, so if $C\llparenthesis V \rrparenthesis \to_{\beta_v} N$ then the contracted redex must occur in V. As V is a value, it can only v-reduce to a value V'.

We have also to check by induction on $C\llparenthesis - \rrparenthesis$ that such an operation does not introduce any σ-redex. Most cases are direct due to the peculiar shape of approximants. In particular we can observed that:

- if $C\llparenthesis \bot \rrparenthesis = (\lambda x.A')(xHA_1 \cdots C'\llparenthesis \bot \rrparenthesis \cdots A_k)$ with $C'\llparenthesis \bot \rrparenthesis \in \mathcal{A}_v$, direct by induction hypothesis. Indeed no matter the shape of $C'\llparenthesis V \rrparenthesis$ no rule will apply.

- if $C\llparenthesis \bot \rrparenthesis = (\lambda x.A')(xC'\llparenthesis \bot \rrparenthesis A_1 \cdots A_k)$ where $C'\llparenthesis \bot \rrparenthesis$ is in $\mathcal{H}$. Since $x \in \Lambda^V$, $xC'\llparenthesis V \rrparenthesis$ would be a σ_3-redex for $C'\llparenthesis V \rrparenthesis = (\lambda y.M)N$ but this is impossible since $C'\llparenthesis \bot \rrparenthesis$ is in $\mathcal{H}$ so it cannot have this shape. □

Lemma 3.2.11. *For $M \in \Lambda$ and $A \in \mathcal{A}_v$, $A \sqsubseteq_v M$ and $M \to_v N$ entails $A \sqsubseteq_v N$.*

Proof. If $A \sqsubseteq_v M$ then M can be obtained from A by substituting each occurrence of $\perp$ for the appropriate subterm of M, and such subterm must be a value. Hence, the redex contracted in M to obtain N must occur in a subterm V of M corresponding to an occurrence $C(\!|-|\!)$ of $\perp$ in A. So we have $C(\!|\perp|\!) = A$ and $C(\!|V|\!) \to_v N'$ implies, by Lemma 3.2.10, that $N' = C(\!|V'|\!)$ for a V' such that $V \to_v V'$. So we conclude that $A = C(\!|\perp|\!) \sqsubseteq_v N$, as desired. □

Lemma 3.2.12. *For $M, N \in \Lambda$, $M \to_v N$ entails $\mathcal{A}_v(M) = \mathcal{A}_v(N)$.*

Proof. Straightforward from Definition 3.2.7 and Lemma 3.2.11. □

Proposition 3.2.13. *For all $M \in \Lambda$, the set $\mathcal{A}_v(M)$ is either empty or an ideal with regards to $\sqsubseteq_v$ (Definition 3.1.10).*

Proof. Assume $\mathcal{A}_v(M)$ is non-empty. We check the remaining two conditions:

- To show that $\mathcal{A}_v(M)$ is directed, we need to prove that every $A_1, A_2 \in \mathcal{A}_v(M)$ have an upper bound $A_3 \in \mathcal{A}_v(M)$. We proceed by induction on A_1:
  - In case $A_1 = \perp$ (resp. $A_2 = \perp$) simply take $A_3 = A_2$ (resp. $A_3 = A_1$).

  Let us assume that $A_1, A_2 \neq \perp$.
  - Case $A_1 = x$, then $M = A_3 = A_2 = x$.
  - Case $A_1 = \lambda x.A'_1$ use the induction hypothesis.
  - Case $A_1 = xH^1A^1_1 \cdots A^1_k$. In this case we must have $M \twoheadrightarrow_v N_1$ for $N_1 = xN'_0 \cdots N'_k$ with $H^1 \sqsubseteq_v N'_0$ and $A^1_i \sqsubseteq_v N'_i$ for $i = 1, \dots, k$.

    As $A_2 \in \mathcal{A}_v(M)$, there exists a λ-term N_2 such that $M \twoheadrightarrow_v N_2$ and $A_2 \sqsubseteq_v N_2$.

    By Proposition 2.3.7, N_1 and N_2 have a common reduct N.

    Since $A_1 \sqsubseteq_v N_1$, by Lemma 3.2.11 we get $A_1 \sqsubseteq_v N$ thus $N = xN_0 \cdots N_k$, with $H^1 \sqsubseteq_v N_0$ and $A^1_i \sqsubseteq_v N_i$ for $i = 1, \dots, k$.

    By Lemma 3.2.11 again, $A_2 \sqsubseteq_v N$ whence $A_2 = xH^2A^2_1 \cdots A^2_k$ for some approximants $H^2 \sqsubseteq_v N_0$ and $A^2_i \sqsubseteq_v N_i$ for $i = 1, \dots, k$.

    Now, by definition, $H^1, H^2 \in \mathcal{A}_v(N_0)$ and $A^1_i, A^2_i \in \mathcal{A}_v(N_i)$ for $i = 1, \dots, k$.

    By induction hypothesis, there exist $H_3 \in \mathcal{A}_v(N_0)$ and $A^3_i \in \mathcal{A}_v(N_i)$ such that $H^1 \sqsubseteq_v H^3 \sqsupseteq H^2$ and $A^1_i \sqsubseteq_v A^3_i \sqsupseteq A^2_i$ from which it follows that the upper bound $xH^3A^3_1 \cdots A^3_k$ of A_1, A_2 belongs to $\mathcal{A}_v(xN_0 \cdots N_k)$.

    By Lemma 3.2.12, we conclude that $xH^3A^3_1 \cdots A^3_k \in \mathcal{A}_v(M)$, as desired.
  - Case $A_1 = (\lambda x.A'_1)(yH^1A^1_1 \cdots A^1_k)$. In this case we must have $M \twoheadrightarrow_v N_1$ for $N_1 = (\lambda x.M')(yM_0 \cdots M_k)$ with $A'_1 \sqsubseteq_v M'$, $H^1 \sqsubseteq_v M_0$ and $A^1_i \sqsubseteq_v M_i$ for $i = 1, \dots, k$.

    Reasoning as above, $A_2 \in \mathcal{A}_v(M)$ implies there exists a λ-term N_2 such that $M \twoheadrightarrow_v N_2$ and $A_2 \sqsubseteq_v N_2$.

By Proposition 2.3.7, N_1 and N_2 have a common reduct N.

Since $A_1 \sqsubseteq_v N_1$, by Lemma 3.2.11 we get $A_1 \sqsubseteq_v N$ thus $N = (\lambda x.N')(yN_0 \cdots N_k)$, with $A'_1 \sqsubseteq_v N'$, $H^1 \sqsubseteq_v N_0$ and $A^1_i \sqsubseteq_v N_i$ for $i = 1, \dots, k$.

By Lemma 3.2.11 again, $A_2 \sqsubseteq_v N$ whence $A_2 = (\lambda x.A'_2)(yH^2A^2_1 \cdots A^2_k)$ where $A'_2 \sqsubseteq_v N'$, $H^2 \sqsubseteq_v N_0$ and $A^2_i \sqsubseteq_v N_i$ for $i = 1, \dots, k$.

By induction hypothesis we get $A'_3 \in \mathcal{A}_v(N')$ such that $A'_1 \sqsubseteq_v A'_3 \sqsupseteq A'_2$, $H^3 \in \mathcal{A}_v(N_0)$ such that $H^1 \sqsubseteq_v H^3 \sqsupseteq H^2$ and $A^3_i \in \mathcal{A}_v(N_i)$ such that $A^1_i \sqsubseteq_v A^3_i \sqsupseteq A^2_i$ for $i = 1, \dots, k$.

It follows that the upper bound $(\lambda x.A'_3)(yH^3A^3_1 \cdots A^3_k)$ of A_1, A_2 belongs to $\mathcal{A}_v((\lambda x.N')(yN_0 \cdots N_k))$.

By Lemma 3.2.12, we conclude that $(\lambda x.A'_3)(yH^3A^3_1 \cdots A^3_k) \in \mathcal{A}_v(M)$.

- To prove that $\mathcal{A}_v(M)$ is downward closed, we need to show that for all $A_1, A_2 \in \mathcal{A}_v$, if $A_1 \sqsubseteq_v A_2 \in \mathcal{A}_v(M)$ then $A_1 \in \mathcal{A}_v(M)$, but this follows directly from the definition. □

The supremum of a set $\mathcal{A}_v(M)$ of approximants of a λ-term M, might not belong to $\mathcal{A}_v$. This is the case for $\mathcal{A}_v(\lambda x.\mathbf{K}^*) = \{\bot, \lambda x_0.\bot, \lambda x_0x_1.\bot, \lambda x_0x_1x_2.\bot, \dots\}$. In fact, such a supremum only exists in the ideal completion of $\mathcal{A}_v$, that we describe below coinductively.

Definition 3.2.14.

1. *The set $\mathcal{B}_v$ of* Call by Value Böhm-like trees *is generated by the following grammar, where inductive and coinductive non-terminal symbols coexist:*

$$\begin{array}{llll} (\mathcal{B}_v) & T & ::= & \varnothing \mid A \\ & A & ::= & H \mid R \\ & H & ::= & x \mid L^{\text{co-ind}} \mid \bot \mid xHA_1 \cdots A_k \qquad \textit{(for } k \geq 0 \textit{ and } x \in \mathbb{V}\textit{)} \\ & L^{\text{co-ind}} & ::= & \lambda x.A \\ & R & ::= & L^{\text{co-ind}}(xHA_1 \cdots A_k) \qquad \textit{(for } k \geq 0 \textit{ and } x \in \mathbb{V}\textit{)} \end{array}$$

Intuitively, this means that in the construction of a production $T \in \mathcal{B}_v$ one can apply the rule $L^{\text{co-ind}} \Rightarrow \lambda x.A$ infinitely many times, while all other rules remain inductive as usual. It is easy to check that $\mathcal{A}_v \subsetneq \mathcal{B}_v$.

2. *The order $\sqsubseteq_v$ is extended from $\mathcal{A}_v$ to $\mathcal{B}_v$ in the obvious way.*

3. *The* (Call-by-Value) Böhm tree *of a λ-term M, is defined as follows (where the supremum is taken in $\mathcal{B}_v$, and we assume that $\bigsqcup \emptyset = \varnothing$):*

$$\mathrm{BT}_v(M) = \bigsqcup \mathcal{A}_v(M).$$

Therefore, the resulting structure is a possibly infinite labelled tree T.

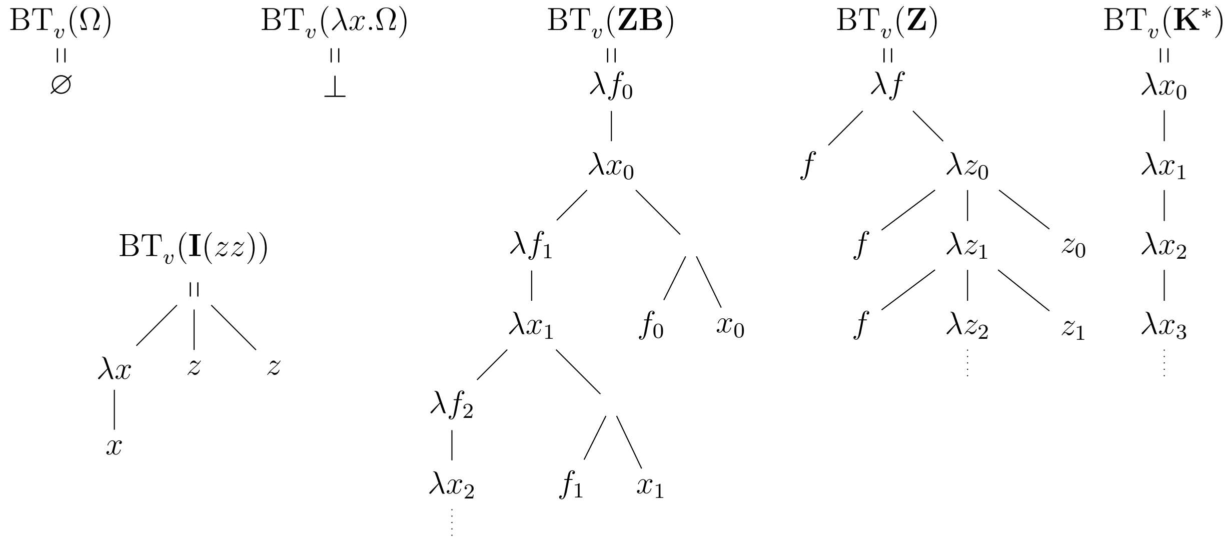

$\mathrm{BT}_v(\Omega) = \varnothing$
$\mathrm{BT}_v(\lambda x.\Omega) = \bot$
$\mathrm{BT}_v(\mathbf{ZB})$
$\mathrm{BT}_v(\mathbf{Z})$
$\mathrm{BT}_v(\mathbf{K}^*)$
$\mathrm{BT}_v(\mathbf{I}(zz))$

Figure 3.2.: Examples of Call-by-Value Böhm trees.

Given a λ-term M, the free variables of its Böhm tree are generated by

$$\mathrm{FV}(\mathrm{BT}_v(M)) = \bigcup_{A \in \mathcal{A}_v(M)} \mathrm{FV}(A).$$

Remark 3.2.15.

- *Notice that* $\mathcal{A}_v(M) = \mathcal{A}_v(N)$ *if and only if* $\mathrm{BT}_v(M) = \mathrm{BT}_v(N)$.
- $\mathrm{FV}(\mathrm{BT}_v(M)) \subseteq \mathrm{FV}(M)$ *and the inclusion can be strict:* $\mathrm{FV}(\mathrm{BT}_v(\lambda x.\Omega y)) = \mathrm{FV}(\bot) = \emptyset$ *but* $\mathrm{FV}(\lambda x.\Omega y) = \{y\}$.

Example 3.2.16. *Notable examples of Böhm trees of λ-terms are given in Figure 3.2. The λ-term Ξ (from Example 2.3.5(7)) satisfying*

$$\Xi =_v (\lambda y_1.((\lambda y_2.(\cdots(\lambda y_n.\Xi\mathbf{I}^n)(zz)\cdots))(zz)))(zz) \tag{3.1}$$

is such that $\mathrm{BT}_v(\Xi) = \bot$. *Indeed, substituting $\bot$ for a $\lambda y_n.\Xi\mathbf{I}^n$ in* (3.1) *never gives an approximant belonging to $\mathcal{A}_v$ by Definition 3.2.3.*

Property 3.2.17. *For $M, N \in \Lambda$, if $M =_v N$ then* $\mathrm{BT}_v(M) = \mathrm{BT}_v(N)$.

Proof. By Proposition 2.3.7, $M =_v N$ if and only if there exists a λ-term L such that $M \twoheadrightarrow_v L$ and $N \twoheadrightarrow_v L$. By an iterated application of Lemma 3.2.12 we get $\mathcal{A}_v(M) = \mathcal{A}_v(L) = \mathcal{A}_v(N)$, so we conclude $\mathrm{BT}_v(M) = \mathrm{BT}_v(N)$. □

3.2.2. Characterisation of Böhm Trees

In this subsection we will provide a characterisation of sets of approximants that correspond to the Böhm trees of λ-terms.

Remark 3.2.18. *A* Böhm-like tree *T is determined by any $\mathcal{X} \subseteq \mathcal{A}_v$ directed and downward closed such that $T = \bigsqcup \mathcal{X}$. In this case, we say that T* is generated by *$\mathcal{X}$.*

All Böhm trees are Böhm-like trees, but not the opposite. It is natural to wonder which characteristics on a Böhm-like tree T ensure that T is a Böhm tree, meaning there exists $M \in \Lambda$ such that $\mathrm{BT}_v(M) = T$. In the spirit of [Barendregt, 1984, Thm. 10.1.23] we will provide such a characterisation in Theorem 3.2.23.

To achieve this result, it will be convenient to consider a tree as a set of sequences closed under prefix.

Notation 3.2.19.

- *We denote by $\mathbb{N}^*$ the set of finite sequences of natural numbers.*
- *Given $n_1, \dots, n_k \in \mathbb{N}$, the corresponding sequence $\sigma \in \mathbb{N}^*$ of length k is represented by $\sigma = \langle n_1, \dots, n_k \rangle$. In particular, $\langle\rangle$ represents the empty sequence of length 0.*
- *Given $\sigma \in \mathbb{N}^*$ as above and $n \in \mathbb{N}$, we write $n :: \sigma$ for the sequence $\langle n, n_1, \dots, n_k \rangle$ and $\sigma; n$ for the sequence $\langle n_1, \dots, n_k, n \rangle$.*

Given a tree T, the sequence $i :: \sigma$ possibly determines a subtree that can be found going through the $(i+1)$-th children of T (if it exists) and then following the path σ. Of course this is only the case if $i :: \sigma$ actually belongs to the domain of the tree. We will use the symbol $\uparrow$ in case we try to reach an nonexistent node and need to backtrack in the tree.

Definition 3.2.20. *Let $\sigma \in \mathbb{N}^*, A \in \mathcal{A}_v$. The* subterm of A at σ, *written A_σ, is defined by:*

$$A_{\langle\rangle} = A \qquad (\lambda x.A)_\sigma = \begin{cases} A_\tau & \textit{if } \sigma = 0 :: \tau, \\ \uparrow & \textit{otherwise}, \end{cases}$$

$$\perp_\sigma = \uparrow \qquad (xA_0 \cdots A_k)_\sigma = \begin{cases} (A_{i-1})_\tau & \textit{if } 1 \leq i \leq k+1 \textit{ and } \sigma = i :: \tau, \\ \uparrow & \textit{otherwise}, \end{cases}$$

$$((\lambda x.A')(yA_0 \cdots A_k))_\sigma = \begin{cases} A'_\tau & \textit{if } \sigma = 0 :: 0 :: \tau, \\ (A_{i-1})_\tau & \textit{if } 1 \leq i \leq k+1 \textit{ and } \sigma = 1 :: i :: \tau, \\ \uparrow & \textit{otherwise.} \end{cases}$$

As a matter of notation, given an approximant A', a subset $\mathcal{X} \subseteq \mathcal{A}_v$ and a sequence $\sigma \in \mathbb{N}^*$, we write $\exists A_\sigma \simeq_{\mathcal{X}} A'$ whenever there exists $A \in \mathcal{X}$ such that A_σ is defined and $A_\sigma = A'$.

Notation 3.2.21. *Given a Böhm-like tree T generated by $\mathcal{X}$, we set*

$$\mathrm{FV}(T) = \mathrm{FV}(\mathcal{X}) = \bigcup_{A \in \mathcal{X}} \mathrm{FV}(A).$$

Definition 3.2.22. *Let E be a set of natural numbers.*

- *E is recursively enumerable if it is the domain of a partial recursive function.*
- *If both E and its complement are recursively enumerable then E is called decidable.*

Equivalently E is recursively enumerable (or semi-decidable) if there is an algorithm with input numbers that halts if and only if the input is in E.

The quote of Mogensen ([Mogensen, 1992]). An encoding of λ-terms as λ-terms in normal forms is introduced, using the representation schema:

- $\ulcorner x \urcorner = \lambda abc.ax$;
- $\ulcorner MN \urcorner = \lambda abc.b\ulcorner M \urcorner \ulcorner N \urcorner$;
- $\ulcorner \lambda x.M \urcorner = \lambda abc.c(\lambda x.\ulcorner M \urcorner)$;

where variables a, b, c do not occur free in the λ-term on the left-hand side of the equation.

This representation schema is injective and the representation is linear in the size of the represented λ-terms. For all M, we have $\mathrm{FV}(M) = \mathrm{FV}(\ulcorner M \urcorner)$.

Originally, the encoding was defined in Call-by-Name. In particular there is an *evaluator*: $\mathbf{E}_\beta = \mathbf{Y}R$ where $R = \lambda em.m(\lambda x.x)(\lambda mn.(em)(en))(\lambda mv.e(mv))$ and $\mathbf{Y}$ is Curry's Call-by-Name fixed point combinator (defined in Figure 2.1).
$\mathbf{E}_\beta$ is such that for any λ-term M, $\mathbf{E}_\beta \ulcorner M \urcorner =_\beta M$.

However, using the Call-by-Value fix-point operator $\mathbf{Z}$ (also defined in Figure 2.1), we can define a Call-by-Value evaluator: $\mathbf{E}_v = \mathbf{Z}R$ and we have $\mathbf{E}_v \ulcorner M \urcorner =_v M$. We can observe that the codings are not only normal forms, but values.

Theorem 3.2.23. *Let $\mathcal{X} \subseteq \mathcal{A}_v$ be a set of approximants. There exists $M \in \Lambda$ such that $\mathcal{A}_v(M) = \mathcal{X}$ if and only if the following three conditions hold:*

1. *$\mathcal{X}$ is directed and downward closed w.r.t. $\sqsubseteq_v$;*
2. *$\mathcal{X}$ is recursively enumerable;*
3. *$\mathrm{FV}(\mathcal{X})$ is finite.*

Proof.

($\Rightarrow$) Let $M \in \Lambda$ be such that $\mathcal{X} = \mathcal{A}_v(M)$, then (1) is satisfied by Proposition 3.2.13 and (3) by Remark 3.2.15.

Concerning (2) let us fix an effective bijective encoding $\# : \Lambda_\perp \to \mathbb{N}$.

We have $\{\#A \mid A \in \mathcal{X}\} = \{\#A \mid A \in \mathcal{A}_v(M)\} = \{\#A \mid \exists N, M \twoheadrightarrow_v N, A \sqsubseteq_v N\}$.

The set $\{\#N \mid M \twoheadrightarrow_v N\}$ is recursively enumerable, we can enumerate the reducts of M (since the calcul is confluent) and compare their code with the input.

The set $\{\#A \mid A \in \mathcal{A}_v\}$ is decidable, indeed for a given $\lambda_\perp$-term we can determine in finite time if it is an approximant, since its size if finite.

Finally, given $A \in \mathcal{A}_v, N \in \Lambda$, the question $A \sqsubseteq_v N$ is decidable since both A and N are of finite size and can then be compared in finite time.

($\Leftarrow$) Assume that $\mathcal{X}$ is a set of approximants satisfying the conditions (1-3).

If $\mathcal{X} = \emptyset$ then we can simply take $M = \Omega$ since $\mathcal{A}_v(\Omega) = \emptyset$.

If $\mathcal{X}$ is non-empty then it is an ideal.

Since $\mathcal{X}$ is recursively enumerable, if $A' \in \mathcal{A}_v$ and $\sigma \in \mathbb{N}^*$ are effectively given then the condition $\exists A_\sigma \simeq_{\mathcal{X}} A'$ is semi-decidable and a witness $\mathbf{A}$ can be computed.

Let $\ulcorner\sigma\urcorner$ be the numeral associated with σ under an effective encoding and $\ulcorner A\urcorner$ be the quote of A as defined by Mogensen, using a fresh variable $z_b \notin \mathrm{FV}(\mathcal{X})$ to represent the $\perp$ (such variable always exists because $\mathrm{FV}(\mathcal{X})$ is finite).

The Call-by-Value λ-calculus being Turing-complete, as shown in [Paolini, 2001], there exists a λ-term $P_{\mathcal{X}}$ satisfying:

$$P_{\mathcal{X}}\ulcorner\sigma\urcorner\ulcorner A'\urcorner =_v \begin{cases} \ulcorner\mathbf{A}\urcorner, & \text{if } \exists A_\sigma \simeq_{\mathcal{X}} A' \text{ holds,} \\ \text{not potentially valuable,} & \text{otherwise.} \end{cases}$$

for some witness $\mathbf{A}$. Using the λ-terms $\mathbf{E}_v, P_{\mathcal{X}}$ so-defined and the recursion operator $\mathbf{Z}$ (see Figure 2.1 for definition of useful combinators), it is possible to define a λ-term F (also depending on $\mathcal{X}$) satisfying the following recursive equations:

$F\,\ulcorner\sigma\urcorner =_v$

$$\begin{cases} x & \text{if } \exists A_\sigma \simeq_{\mathcal{X}} x, \\ \lambda x.F\,\ulcorner\sigma;0\urcorner & \text{if } \exists A_\sigma \simeq_{\mathcal{X}} \lambda x.A_1, \\ x(F\,\ulcorner\sigma;1\urcorner)\cdots(F\,\ulcorner\sigma;k+1\urcorner) & \text{if } \exists A_\sigma \simeq_{\mathcal{X}} xA_0\cdots A_k, \\ \big(\lambda x.F\ulcorner\sigma;0;0\urcorner\big)\big(y(F\ulcorner\sigma;1;1\urcorner)\cdots(F\ulcorner\sigma;1;k+1\urcorner)\big) & \text{if } \exists A_\sigma \simeq_{\mathcal{X}} (\lambda x.A)(yA_0\cdots A_k), \\ \text{not valuable} & \text{otherwise.} \end{cases}$$

The fact that $\mathcal{X}$ is directed guarantees that, for a given sequence σ, exactly one of the cases above is applicable. It is now easy to see that $\mathcal{A}_v(F\,\ulcorner\langle\rangle\urcorner) = \mathcal{X}$. □

4. Taylor Expansion

The Taylor expansion is an alternative way of approximating λ-terms which was introduced in [Ehrhard and Regnier, 2003]. The idea is the same as the classical Taylor expansion for functions: expressing a function near a point as an infinite sum of terms depending of that function's derivatives at this point.

The Taylor expansion translates a λ-term M in a set of multi-linear terms, each one of those terms approximates a finite part of the behaviour of M. The potentially infinitary nature of the behaviour of M is captured by the set of multi-linear terms being potentially infinite.

Those multi-linear terms populate a *resource calculus* [Tranquilli, 2009] where λ-calculus application is replaced by the application of a term to a bag of resources that cannot be erased, nor duplicated and must all be consumed during the reduction. With such a target language for the Taylor expansion, the amount of resources needed by a λ-term to produce (a finite part of) a value is explicit. Therefore the Taylor expansion contains a very useful quantitative information, while approximations such as Böhm trees are only qualitative.

If those definitions of resource calculus and Taylor expansion are originally in the Call-by-Name case, the Call-by-Value analogues are unproblematic to define. They are driven by solid intuitions coming from the translation in Linear Logic. There are two ways to translate the intuitionistic arrow in Linear Logic: each corresponding respectively to either the Call-by-Name and the Call-by-Value λ-calculus. The implication $a \Rightarrow b$ becomes in Call-by-Name $!a \multimap b$ and in Call-by-Value $!(a \multimap b)$. This second transformation is called “boring” and is suitable for Call-by-Value (see [Ehrhard, 2012]).

In its original definition, the Taylor expansion is a power series of multi-linear terms taking coefficients in the semiring of non-negative rational numbers. In this work, similarly as in [Manzonetto and Pagani, 2011, Ehrhard, 2012, Boudes et al., 2013], we abuse language and call “Taylor expansion” the support of the actual Taylor expansion. This is done because we are interested in the usual observational equivalences between λ-terms that overlook such coefficients.

In the present chapter, we will introduce in Section 4.1 the Call-by-Value resource calculus denoted λ_r^σ, and in Section 4.2 the Taylor expansion in Call-by-Value. Those two notions have been defined in [Ehrhard, 2012, Carraro and Guerrieri, 2014].

Then, Section 4.3 contains an original characterisation of sets of resource terms corresponding to the Taylor expansion of some λ-term (Theorem 4.3.4): we define, following [Boudes et al., 2013], a coherence relation $\frown$ between resource terms and prove that a set of such terms corresponds to the Taylor expansion of a λ-term if and only if it is an infinite clique having finite height.

In Section 4.4, finally, we investigate normal forms of Taylor expansions, those normal forms can always be calculated since the resource calculus enjoys strong normalisation.

This chapter is derived from [Kerinec et al., 2020].

4.1. Resource Calculus

In this section we introduce the resource calculus from [Carraro and Guerrieri, 2014].

Definition 4.1.1. *The sets* Λ_r *of* resource terms*,* Λ_r^V *of* resource values*, and* Λ_s *of* simple terms *are generated by the following grammar:*

$$\begin{array}{llrcll} \textit{resource terms} & (\Lambda_r) & e & ::= & v \mid s & \\ \textit{resource values} & (\Lambda_r^V) & u, v & ::= & x \mid \lambda x.t & (\textit{for } x \in \mathbb{V}) \\ \textit{simple terms} & (\Lambda_s) & s, t & ::= & st \mid [v_1, \ldots, v_k] & (\textit{for } k \geq 0) \end{array}$$

The concepts of α-conversion *and* free variable *are inherited from classical Call-by-Value (and Call-by-Name) λ-calculus (Definition 2.2.4).*

Resource values are analogous to the values of Call-by-Value λ-calculus, namely variables and λ-abstractions. A simple term of the shape $[v_1, \ldots, v_n]$ is called a bag *and the order of the subterms do not matter.*

The idea behind λ_r^σ is to have a quantitative notion of resource. A bag $[v_1, \ldots, v_n]$ represents a finite multiset of linear resources. During a reduction we will see that every v_i must be used exactly once. Indeed, when a singleton bag $[\lambda x.t]$ is applied to a bag $[v_1, \ldots, v_n]$ of resource values, each v_i is substituted for exactly one free occurrence of x in t. This occurrence is chosen non-deterministically, and all possibilities are taken into account. Since we have to take different choices into account the result of a reduction will be presented as a set-theoretical union of resource terms. In case there is a mismatch between the cardinality of the bag and the number of occurrences of x in t, the result of the computation is the empty set $\emptyset$.

The height $\mathsf{ht}(e)$ of e *is the height of its syntax tree:*

$$\begin{array}{lcl} \mathsf{ht}(x) & = & 0, \\ \mathsf{ht}(\lambda x.t) & = & \mathsf{ht}(t) + 1, \\ \mathsf{ht}(st) & = & \max\{\mathsf{ht}(s), \mathsf{ht}(t)\} + 1, \\ \mathsf{ht}([v_1, \ldots, v_k]) & = & \max\{\mathsf{ht}(v_i) \mid i \leq k\} + 1. \end{array}$$

Let us introduce some notations concerning those sets of resource terms.

Notation 4.1.2. *Sets of resource values, simple terms and resource terms are denoted by:*

$$\mathcal{U},\mathcal{V}\in\mathscr{P}(\Lambda_r^V),\qquad \mathcal{S},\mathcal{T}\in\mathscr{P}(\Lambda_s),\qquad \mathcal{E}\in\mathscr{P}(\Lambda_r).$$

For the sake of simplicity, given $\mathcal{S},\mathcal{T}\in\mathscr{P}(\Lambda_s)$ and $\mathcal{V}_1,\ldots,\mathcal{V}_k\in\mathscr{P}(\Lambda_r^V)$ we fix the following notations:

$$\begin{array}{rcl} \lambda x.\mathcal{T} & = & \{\lambda x.t \mid t\in\mathcal{T}\}\in\mathscr{P}(\Lambda_r^V),\\ \mathcal{S}\mathcal{T} & = & \{st \mid s\in\mathcal{S}, t\in\mathcal{T}\}\in\mathscr{P}(\Lambda_s),\\ [\mathcal{V}_1,\ldots,\mathcal{V}_k] & = & \{[v_1,\ldots,v_k] \mid v_1\in\mathcal{V}_1,\ldots,v_k\in\mathcal{V}_k, k\geq 0\}\in\mathscr{P}(\Lambda_s). \end{array}$$

From the multi-linearity of λ_r^σ-constructors, we have the following equality:

$$\lambda x.\emptyset=\emptyset\mathcal{T}=\mathcal{S}\emptyset=[\emptyset,\mathcal{V}_1,\ldots,\mathcal{V}_k]=\emptyset.$$

We use $[x^n]$ for $[\underbrace{x,\ldots,x}_{n\ times}]$.

Definition 4.1.3. *Let $e\in\Lambda_r$ and $x\in\mathbb{V}$.*

1. *Define the* degree of x in e, *written $\deg_x(e)$, as the number of free occurrences of the variable x in the resource term e.*
2. *Given $e\in\Lambda_r$, $v_1,\ldots,v_n\in\Lambda_r^V$ and $x\in\mathbb{V}$. The* linear substitution of $v_1,\ldots,v_n$ for x in e, *denoted by $e\langle[v_1,\ldots,v_n]/x\rangle\in\mathscr{P}_{\mathrm{f}}(\Lambda_r)$, is defined as follows:*

$$e\langle[v_1,\ldots,v_n]/x\rangle=\begin{cases}\{e[v_{\sigma(1)}/x_1,\ldots,v_{\sigma(n)}/x_n]\mid\sigma\in\mathfrak{S}_n\} & \textit{if } \deg_x(e)=n,\\ \emptyset & \textit{otherwise.}\end{cases}$$

where $\mathfrak{S}_n$ is the group of permutations over $\{1,\ldots,n\}$ and $x_1,\ldots,x_n$ is an enumeration of the free occurrences of x in e, so that $e[v_{\sigma(i)}/x_i]$ denotes the resource term obtained from e by replacing the i-th free occurrence of x in e with the resource value $v_{\sigma(i)}$.

The following reductions for λ_r^σ correspond to the reductions of Call-by-Value λ-calculus (Definitions 2.3.2 and 2.3.3).

Definition 4.1.4.

1. *The* β_r-reduction *is a relation $\rightarrow_{\beta_r}\subseteq\Lambda_r\times\mathscr{P}_{\mathrm{f}}(\Lambda_r)$ defined as the contextual closure (using the first four rules in Figure 4.1) of the following rule:*

$$(\beta_r)\qquad [\lambda x.t][v_1,\ldots,v_n]\ \mapsto\ t\langle[v_1,\ldots,v_n]/x\rangle\qquad (\textit{for } v_1,\ldots,v_n\in\Lambda_r^V)$$

Similarly, the 0-reduction $\rightarrow_0\subseteq\Lambda_r\times\mathscr{P}_{\mathrm{f}}(\Lambda_r)$ *is defined by the rule:*

$$(0)\qquad [v_1,\ldots,v_n]\,t\ \mapsto\ \emptyset\qquad (\textit{for } n\neq 1)$$

The σ-reductions $\rightarrow_{\sigma_1},\rightarrow_{\sigma_3}\subseteq\Lambda_r\times\Lambda_r$*:*

$$\begin{array}{llll} (\sigma_1) & [\lambda x.t]s_1s_2 & \mapsto & [\lambda x.ts_2]s_1 \qquad (\textit{for } x\notin\mathrm{FV}(s_2))\\ (\sigma_3) & [v]([\lambda x.t]s) & \mapsto & [\lambda x.[v]t]s \qquad (\textit{for } x\notin\mathrm{FV}(v) \textit{ and } v\in\Lambda_r^V) \end{array}$$

$$\frac{t \;\to_{\mathsf{r}}\; \mathcal{T}}{\lambda x.t \;\to_{\mathsf{r}}\; \lambda x.\mathcal{T}} \qquad \frac{s \;\to_{\mathsf{r}}\; \mathcal{S}}{s\,t \;\to_{\mathsf{r}}\; \mathcal{S}\,t} \qquad \frac{t \;\to_{\mathsf{r}}\; \mathcal{T}}{s\,t \;\to_{\mathsf{r}}\; s\,\mathcal{T}}$$

$$\frac{v_0 \;\to_{\mathsf{r}}\; \mathcal{V}_0}{[v_0, v_1, \ldots, v_k] \;\to_{\mathsf{r}}\; [\mathcal{V}_0, v_1, \ldots, v_k]} \qquad \frac{e \;\to_{\mathsf{r}}\; \mathcal{E}_1 \quad e \notin \mathcal{E}_2}{\{e\} \cup \mathcal{E}_2 \;\to_{\mathsf{r}}\; \mathcal{E}_1 \cup \mathcal{E}_2}$$

Figure 4.1.: Contextual rules for a relation $\to_{\mathsf{r}} \subseteq \mathscr{P}_{\mathrm{f}}(\Lambda_r) \times \mathscr{P}_{\mathrm{f}}(\Lambda_r)$.

2. *The relation $\to_{\mathsf{r}} \subseteq \mathscr{P}_{\mathrm{f}}(\Lambda_r) \times \mathscr{P}_{\mathrm{f}}(\Lambda_r)$ is the contextual closure of the rules above, in other words $\to_{\mathsf{r}}$ is the smallest relation including $(\beta_r), (0), (\sigma_1), (\sigma_3)$ and satisfying the rules in Figure 4.1.*

Example 4.1.5.

1. $[\lambda x.[x][x]][\lambda y.[y], z] \to_{\beta_r} \{[\lambda y.[y]][z], [z][\lambda y.[y]]\} \to_{\beta_r} \{[z], [z][\lambda y.[y]]\}$*;*
2. $[\lambda x.[x, x]][\lambda y.[y], z] \to_{\beta_r} \{[\lambda y.[y], z], [z, \lambda y.[y]]\} = \{[\lambda y.[y], z]\}$*;*
3. $[\lambda y.[\lambda x.[x, x][y]]]([z][w])[\mathbf{I}, w] \to_{\sigma_1} [\lambda y.[\lambda x.[x, x][y]][\mathbf{I}, w]]([z][w])$
   $\to_{\beta_r} \{[\lambda y.[\mathbf{I}, w][y]]([z][w])\} \to_0 \emptyset$*. Notice that (σ_1) is used to unblock an otherwise stuck β_r-redex;*
4. $[\mathbf{I}]([\lambda x.[\lambda y.[x][y]]][z][w]) \to_{\beta_r} \{[\lambda x.[\lambda y.[x][y]]][z][w]\} \to_{\beta_r} \{[\lambda y.[z][y]][w]\} \to_{\beta_r} \{[z][w]\}$*;*
5. $[\mathbf{I}]([\lambda x.[\lambda y.[x][y]]][z][w]) \to_{\sigma_3} [\lambda x.[\mathbf{I}][\lambda y.[x][y]]][z][w] \to_{\sigma_1} [\lambda x.[\mathbf{I}][\lambda y.[x][y]][w]][z]$
   $\to_{\beta_r} \{[\mathbf{I}][\lambda y.[z][y]][w]\} \to_{\beta_r} \{[\lambda y.[z][y]][w]\} \to_{\beta_r} \{[z][w]\}$.

Remark that (4) *and* (5) *constitute two different reduction sequences originating from the same simple term.*

As shown in [Carraro and Guerrieri, 2014], this notion of reduction enjoys the following properties.

Property 4.1.6. *The reduction $\to_{\mathsf{r}}$ is confluent and strongly normalising.*

Proof.

- Strong normalisation is easy to prove:
  - 0-reduction erases the whole term;
  - σ-rules are strongly normalising;
  - contracting a β_r-redex in a resource term e produces a set (of finite size) of resource terms whose each size is strictly smaller than the one of e (smaller because no duplication is involved and a λ-abstraction is erased).
- λ_r^σ is a restriction of the original resource calculus in [Ehrhard, 2012] using coefficient in $\mathbb{N}$, from it can be deduced the confluence of the rule $\beta_r \cup 0$ in the present case.

  σ rules are locally confluent (the proof is simple by induction) and are strongly normalising, again we use Lemma 2.1.4. It is simple to see that they commute with β_r as well as 0. And using Lemma 2.1.5 we have the confluence. □

4.2. Call-by-Value Taylor Expansion

Now that we have recalled the resource calculus for Call-by-Value, we will present the Taylor expansion that translates a λ-term in a set of those resource terms.

Definition 4.2.1. *The Taylor expansion $\mathscr{T}(M) \subseteq \Lambda_s$ of a λ-term M is an infinite set of simple terms defined by induction as follows:*

$$\begin{array}{lcl} \mathscr{T}(x) & = & \{[x^n] \mid n \geq 0\}, \\ \mathscr{T}(NL) & = & \{st \mid s \in \mathscr{T}(N),\ t \in \mathscr{T}(L)\}, \\ \mathscr{T}(\lambda x.N) & = & \{[\lambda x.t_1, \ldots, \lambda x.t_n] \mid n \geq 0, i = 1, \ldots, n,\ t_i \in \mathscr{T}(N)\}. \end{array}$$

The Taylor expansion is a static object, indeed the resource terms associated with an abstraction are also all abstractions, the same for an application or a variable.

Remark 4.2.2.

1. *$[\,] \in \mathscr{T}(M)$ if and only if $M \in \Lambda^V$.*
2. *Each occurrence of a $\beta_r\sigma$-redex in $t \in \mathscr{T}(M)$ corresponds to a v-redex in M.*
3. *By exploiting Notation 4.1.2, we can rewrite the Taylor expansion of an application or an abstraction as follows:*

$$\begin{array}{lcl} \mathscr{T}(NL) & = & \mathscr{T}(N)\mathscr{T}(L), \\ \mathscr{T}(\lambda x.N) & = & \bigcup_{n\in\mathbb{N}}\{[\underbrace{\lambda x.\mathscr{T}(N), \ldots, \lambda x.\mathscr{T}(N)}_{n \text{ times}}]\}. \end{array}$$

Example 4.2.3. *Taylor expansion of some λ-terms (combinators are defined in Figure 2.1):*

1. $\mathscr{T}(\mathbf{I}) = \{[\lambda x.[x^{n_1}], \ldots, \lambda x.[x^{n_k}]] \mid k \geq 0, \forall i \leq k, n_i \geq 0\}$;
2. $\mathscr{T}(\Delta) = \{[\lambda x.[x^{n_1}][x^{m_1}], \ldots, \lambda x.[x^{n_k}][x^{m_k}]] \mid k \geq 0, \forall i \leq k, m_i, n_i \geq 0\}$;
3. $\mathscr{T}(\Delta\mathbf{I}) = \{st \mid s \in \mathscr{T}(\Delta), t \in \mathscr{T}(\mathbf{I})\}$;
4. $\mathscr{T}(\Omega) = \{st \mid s, t \in \mathscr{T}(\Delta)\}$;
5. $$\begin{array}{ll} \mathscr{T}(\lambda z.yyz) = & \{[\lambda z.[y^{\ell_1}][y^{m_1}][z^{n_1}], \ldots, \lambda z.[y^{\ell_k}][y^{m_k}][z^{n_k}]] \\ & \qquad \mid k \geq 0, \forall i \leq k, \ell_i, m_i, n_i \geq 0\}; \end{array}$$
6. $$\begin{array}{ll} \mathscr{T}(\lambda y.f(\lambda z.yyz)) = & \{[\lambda y.[f^{n_1}]t_1, \ldots, \lambda y.[f^{n_k}]t_k] \\ & \mid k \geq 0, \forall i \leq k, n_i \geq 0, t_i \in \mathscr{T}(\lambda z.yyz)\}; \end{array}$$
7. $\mathscr{T}(\mathbf{Z}) = \{[\lambda f.s_1t_1, \ldots, \lambda f.s_kt_k] \mid k \geq 0, \forall i \leq k, s_i, t_i \in \mathscr{T}(\lambda y.f(\lambda z.yyz))\}$.

From those examples we can deduce the following remarks and lemma.

Remark 4.2.4.

- *An element t belonging to the Taylor expansion of a v-normal form M might not be in r-nf, due to the possible presence of 0-redexes.*

  For example, take $[\lambda x.[x,x][x,x], \lambda x.[x][x,x,x]] \in \mathscr{T}(\Delta)$, clearly Δ is in v-normal form, but $[x,x][x,x] \to_0 \emptyset$ since the first bag is of size 2, so $\lambda x.[x,x][x,x] \to_0 \emptyset$ and then $[\lambda x.[x,x][x,x], \lambda x.[x][x,x,x]] \to_0 \emptyset$.

- *We can refine the definition of Taylor expansion eliminating all the 0-redexes. It is sufficient to substitute the application case with the following:*

  $$\mathscr{T}(VM_0 \cdots M_k) = \{[v]t_0 \cdots t_k \mid [v] \in \mathscr{T}(V), \forall i = 0, \ldots, k,\ t_i \in \mathscr{T}(M_i)\}.$$

  Rather than having some resource terms $[v^n]t_0 \cdots t_k \to_0 \emptyset$.

  However we prefer to keep Ehrhard's original idea because it has a simpler inductive definition.

Lemma 4.2.5. *For $M \in \Lambda$, the following are equivalent:*

1. *M is in v-normal form;*
2. *every $t \in \mathscr{T}(M)$ is in $\beta_r\sigma$-normal form.*

Proof.

($1 \Rightarrow 2$) Using Lemma 2.3.6, we proceed by induction on the normal structure of M:

- If $M = x$ then $t \in \mathscr{T}(M)$ entails $t = [x, \ldots, x]$ which is in v-nf.
- If $M = \lambda x.G$ then $t \in \mathscr{T}(M)$ implies that $t = [\lambda x.t_1, \ldots, \lambda x.t_n]$ where $t_i \in \mathscr{T}(G)$ for all $i \leq n$. By the induction hypothesis each t_i is in $\beta_r\sigma$-nf, hence, so is t.
- If $M = xHG_1 \cdots G_k$ then $t \in \mathscr{T}(M)$ entails $t = [x^n]st_1 \cdots t_k$ for some $n \geq 0$, $s \in \mathscr{T}(H)$ and for $i = 1, \ldots, k$, $t_i \in \mathscr{T}(G_i)$. By induction hypothesis $s, t_1, \ldots, t_k$ are in $\beta_r\sigma$-nf, so t is in β_r-nf. Concerning σ-rules, t could have a σ_3-redex in case $s = [\lambda x.s']t'$ but this is impossible since $s \in \mathscr{T}(H)$ and H cannot have shape $(\lambda x.L_1)L_2$.
- If $M = (\lambda x.G)(yHG_1 \cdots G_k)$ and $t \in \mathscr{T}(M)$ then $t = [\lambda x.s_1, \ldots, \lambda x.s_n]t'$ for some $n \geq 0$, $i = 1, \ldots, n$,$s_i \in \mathscr{T}(G)$, and $t' \in \mathscr{T}(yHG_1 \cdots G_k)$. By induction hypothesis, the resource terms $s_1, \ldots, s_n$ and t' are in $\beta_r\sigma$-nf. In principle, when $n = 1$, the simple term t might have the shape either of a β_r-redex or of a σ_3-redex. Both cases are impossible since $t' \in \mathscr{T}(yHG_1 \cdots G_k)$ entails $t' = [y^m]st_1 \cdots t_k$ which is neither a resource value nor a simple term of shape $[\lambda z.s]s'$. We conclude that t is in $\beta_r\sigma$-nf.

($2 \Rightarrow 1$) We prove the contrapositive. Assume that M is not in v-nf, then either M itself is a β_v- or σ-redex, or it contains one as a subterm. Let us analyse first the former case.

(β_v) If $M = (\lambda x.N)V$ for $V \in \Lambda^V$ then, by Remark 4.2.2(1), the β_r-redex $[\lambda x.s][\,]$ belongs to $\mathscr{T}(M)$ for every $s \in \mathscr{T}(N)$.

(σ_1) If $M = (\lambda x.N)L_1L_2$ then for all $s \in \mathscr{T}(N), t_1 \in \mathscr{T}(L_1), t_2 \in \mathscr{T}(L_2)$ we have $[\lambda x.s]t_1t_2 \in \mathscr{T}(M)$ and this simple term is a σ_1-redex.

(σ_3) If $M = V((\lambda x.N)L)$ for $V \in \Lambda^V$ then for all $[v] \in \mathscr{T}(V), s \in \mathscr{T}(N)$ and $t' \in \mathscr{T}(L)$ we have $[v]([\lambda x.s]t') \in \mathscr{T}(M)$ and this resource term is a σ_2-redex.

Otherwise $M = C(\!|M'|\!)$ where C is a context and M' is a v-redex having one of the shapes above; in this case there is $t \in \mathscr{T}(M)$ containing a $\beta_r\sigma$-redex $t' \in \mathscr{T}(M')$ as a subterm. □

4.3. Characterisation of the Taylor Expansion

We will now focus on a characterisation of all sets of simple terms that arise as the Taylor expansion of some λ-term.

We proceed similarly as in [Boudes et al., 2013], which was for the Call-by-Name case.

Definition 4.3.1.

1. *The* height *of a non-empty set $\mathcal{E} \subseteq \Lambda_r$, written* $\mathsf{ht}(\mathcal{E})$, *is the maximal height of its elements, if it exists, and in this case we say that $\mathcal{E}$ has* finite *height.*
Otherwise, we define $\mathsf{ht}(\mathcal{E}) = \aleph_0$ *and we say that $\mathcal{E}$ has* infinite *height.*

2. *Define a* coherence relation $\frown \subseteq \Lambda_r \times \Lambda_r$ *as the smallest relation satisfying:*

$$\frac{}{x \frown x} \qquad \frac{s \frown t}{\lambda x.s \frown \lambda x.t}$$

$$\frac{v_i \frown v_j \quad (\forall i, j \leq n)}{[v_1, \dots, v_k] \frown [v_{k+1}, \dots, v_n]} \qquad \frac{s_1 \frown s_2 \quad t_1 \frown t_2}{s_1t_1 \frown s_2t_2}$$

3. *A subset $\mathcal{E} \subseteq \Lambda_r$ is a* clique *whenever $e \frown e'$ holds for all $e, e' \in \mathcal{E}$.*

4. *A clique $\mathcal{E}$ is* maximal *if, for every $e \in \Lambda_r$, $\mathcal{E} \cup \{e\}$ is a clique entails $e \in \mathcal{E}$.*

The coherence relation above is inspired by Ehrhard and Regnier's work in the Call-by-Name setting [Ehrhard and Regnier, 2008]. Note that $\frown$ is symmetric, but neither reflexive as $[x, y] \not\frown [x, y]$ nor transitive since $[x] \frown [\,] \frown [y]$ but $[x] \not\frown [y]$.

Example 4.3.2. *In Example 4.2.3 all sets are maximal cliques of finite height. For instance,* $\mathsf{ht}(\mathscr{T}(\mathbf{I})) = 3$ *and by following the rules in Definition 4.3.1(2) we have $u \frown t$ for all $t \in \mathscr{T}(\mathbf{I})$ if and only if $u = [\lambda x.[x^{n_1}], \dots, \lambda x.[x^{n_k}]]$ for some $k, n_1, \dots n_k \in \mathbb{N}$, and such a u is in $\mathscr{T}(\mathbf{I})$. Therefore $\mathscr{T}(\mathbf{I})$ is maximal.*

The lemma follows from Definition 4.3.1(1) and Remark 4.2.2(3).

Lemma 4.3.3. *For $N, L \in \Lambda$, we have:*

1. $\mathsf{ht}(\mathscr{T}(\lambda x.N)) = \mathsf{ht}(\mathscr{T}(N)) + 2$;
2. $\mathsf{ht}(\mathscr{T}(NL)) = \mathsf{ht}(\mathscr{T}(N) \cup \mathscr{T}(L)) + 1$.

Proof. Easy calculations give:

$$\begin{array}{lrcl}
(1) & \mathsf{ht}(\mathscr{T}(\lambda x.N)) & = & \max\{\mathsf{ht}([\lambda x.t_1, \dots, \lambda x.t_n]) \mid n \geq 0, \forall i \leq n,\ t_i \in \mathscr{T}(N)\}, \\
& & = & \max\{\max\{\mathsf{ht}(\lambda x.t_1), \dots, \mathsf{ht}(\lambda x.t_n)\} + 1 \\
& & & \qquad\qquad\qquad \mid n \geq 0, \forall i \leq n,\ t_i \in \mathscr{T}(N)\}, \\
& & = & \max\{\max\{\mathsf{ht}(t_1), \dots, \mathsf{ht}(t_n)\} + 2 \mid n \geq 0, \forall i \leq n,\ t_i \in \mathscr{T}(N)\}, \\
& & = & \max\{\mathsf{ht}(t) + 2 \mid t \in \mathscr{T}(N)\}, \\
& & = & \mathsf{ht}(\mathscr{T}(N)) + 2.
\end{array}$$

$$\begin{array}{lrcl}
(2) & \mathsf{ht}(\mathscr{T}(NL)) & = & \max\{\mathsf{ht}([t_1t'_1, \dots, t_nt'_n]) \\
& & & \qquad \mid n \geq 0, \forall i \leq n,\ t_i \in \mathscr{T}(N),\ t'_i \in \mathscr{T}(L)\}, \\
& & = & \max\{\max\{\mathsf{ht}(t_1t'_1), \dots, \mathsf{ht}(t_nt'_n)\} + 1 \\
& & & \qquad \mid n \geq 0, \forall i \leq n,\ t_i \in \mathscr{T}(N),\ t'_i \in \mathscr{T}(L)\}, \\
& & = & \max\{\max\{\mathsf{ht}(t_1), \mathsf{ht}(t'_1), \dots, \mathsf{ht}(t_n), \mathsf{ht}(t'_n)\} + 1 \\
& & & \qquad \mid n \geq 0, \forall i \leq n,\ t_i \in \mathscr{T}(N),\ t'_i \in \mathscr{T}(L)\}, \\
& & = & \max\{\mathsf{ht}(t) + 1 \mid t \in \mathscr{T}(N) \cup \mathscr{T}(L)\}, \\
& & = & \mathsf{ht}(\mathscr{T}(N) \cup \mathscr{T}(L)) + 1.
\end{array}$$

This concludes the proof. □

The next theorem gives a characterisation of those sets of simple terms corresponding to the Taylor expansion of some λ-terms and constitutes the main result of this section.

Theorem 4.3.4. *For $\mathcal{E} \subseteq \Lambda_s$, the following are equivalent:*

1. *$\mathcal{E}$ is a maximal clique having finite height;*
2. *There exists $M \in \Lambda$ such that $\mathcal{E} = \mathscr{T}(M)$.*

Proof.

($1 \Rightarrow 2$) As $\mathcal{E}$ maximal entails $\mathcal{E} \neq \emptyset$, we can proceed by induction on $h = \mathsf{ht}(\mathcal{E})$. Depending on h:

- if $h = 0$ impossible because no simple term has height 0.
- if $h = 1$ then $t \in \mathcal{E}$ implies $t = [x_1, \dots, x_n]$ since variables are the only resource terms of height 0. Now, $t \frown t$ holds since $\mathcal{E}$ is a clique so the x_i's must be pairwise coherent with each other, but $x_i \frown x_j$ holds if and only if $x_i = x_j$ whence $t = [x_i, \dots, x_i]$ for some index i. From this, and the fact that $\mathcal{E}$ is maximal, we conclude $\mathcal{E} = \mathscr{T}(x_i)$.
- if $h > 1$ then split into cases depending on the form of $t \in \mathcal{E}$.

- ∗ Case $t = [\lambda x.s_1, \ldots, \lambda x.s_k]$. Since $\mathsf{ht}(\mathcal{E}) > 1$ we can assume wlog that $t \neq [\,]$, namely $k > 0$. Moreover, since $\mathcal{E}$ is a clique, all $t' \in \mathcal{E}$ must have shape $t' = [\lambda x.s_{k+1}, \ldots, \lambda x.s_n]$ for some n with $s_i \coh s_j$ for all $i, j \leq n$. It follows that the set $\mathcal{S} = \{s \mid [\lambda x.s] \in \mathcal{E}\}$ is a maximal clique, because $\mathcal{E}$ is maximal, and has height $h - 2$ since $\mathsf{ht}([\lambda x.s]) = \mathsf{ht}(s) + 2$. Moreover, $\mathcal{E} = \{[\lambda x.s_1, \ldots, \lambda x.s_k] \mid k \geq 0, \forall i \leq k \ .\ s_i \in \mathcal{S}\}$. By induction hypothesis there exists $N \in \Lambda$ such that $\mathcal{S} = \mathscr{T}(N)$, so we get $\mathcal{E} = \mathscr{T}(\lambda x.N)$.
- ∗ Otherwise, if $t = s_1 s_2$ then all $t' \in \mathcal{E}$ must be of the form $t' = s'_1 s'_2$ with $s_1 \coh s'_1$ and $s_2 \coh s'_2$. So, the set $\mathcal{E}$ can be written as $\mathcal{E} = \mathcal{S}_1 \mathcal{S}_2$ where $\mathcal{S}_1 = \{t \mid t s_2 \in \mathcal{E}\}$ and $\mathcal{S}_2 = \{t \mid s_1 t \in \mathcal{E}\}$. As $\mathcal{E}$ is a maximal clique, the sets $\mathcal{S}_1, \mathcal{S}_2$ are independent from the choice of s_2, s_1 (resp.), and they are maximal cliques themselves. Moreover, $\mathsf{ht}(\mathcal{E}) = \mathsf{ht}(\mathcal{S}_1 \cup \mathcal{S}_2) + 1$, whence the heights of $\mathcal{S}_1, \mathcal{S}_2$ are strictly smaller than h. By the induction hypothesis, there exists $N, L \in \Lambda$ such that $\mathcal{S}_1 = \mathscr{T}(N)$ and $\mathcal{S}_2 = \mathscr{T}(L)$, from which it follows $\mathcal{E} = \mathscr{T}(NL)$.

(2 ⇒ 1) We proceed by induction on the structure of M.

- If $M = x$ then $t, t' \in \mathscr{T}(M)$ entails $t = [x^k]$ and $t' = [x^n]$ for some $k, n \geq 0$, whence $\mathscr{T}(x)$ is a clique of height 1. It is moreover maximal because it contains $[x^i]$ for all $i \geq 0$.
- If $M = \lambda x.N$ then $t, t' \in \mathscr{T}(M)$ entails $t = [\lambda x.t_1, \ldots, \lambda x.t_k]$ and $t' = [\lambda x.t_{k+1}, \ldots, \lambda x.t_n]$ with $t_i \in \mathscr{T}(N)$ for all $i \leq n$. By induction hypothesis $\mathscr{T}(N)$ is a maximal clique of finite height $h \in \mathbb{N}$, in particular $t_i \coh t_j$ for all $i, j \leq n$ which entails $t \coh t'$. The maximality of $\mathscr{T}(M)$ follows from that of $\mathscr{T}(N)$ and, by Lemma 4.3.3(1), $\mathsf{ht}(\mathscr{T}(M))$ has finite height $h + 2$.
- If $M = NL$ then $t, t' \in \mathscr{T}(M)$ entails $t = s_1 t_1$ and $t' = s_2 t_2$ for $s_1, s_2 \in \mathscr{T}(N)$ and $t_1, t_2 \in \mathscr{T}(L)$. By induction hypothesis, $s_1 \coh s_2$ and $t_1 \coh t_2$ hold and thus $t \coh t'$. Also in this case, the maximality of $\mathscr{T}(M)$ follows from the same property of $\mathscr{T}(N), \mathscr{T}(L)$. Finally, by induction hypothesis, $\mathsf{ht}(\mathscr{T}(N)) = h_1$ and $\mathsf{ht}(\mathscr{T}(L)) = h_2$ for $h_1, h_2 \in \mathbb{N}$ then $\mathsf{ht}(\mathscr{T}(M)) = \max\{h_1, h_2\} + 1$ by Lemma 4.3.3(2), and this concludes the proof. □

4.4. Normal Forms of Taylor Expansions

We will now observe a dynamic version of the Taylor expansion. We define the normal form of the Taylor expansion of λ-terms. Finally we observe the link between reductions on a λ-term and reductions on its Taylor expansion.

Definition 4.4.1. *The* r-normal form *is extended element-wise to any subset* $\mathcal{E} \subseteq \Lambda_r$ *by setting*

$$\mathrm{NF}(\mathcal{E}) = \bigcup_{e \in \mathcal{E}} \mathrm{NF}_r(e).$$

We can now for any $M \in \Lambda$ refer to $\mathrm{NF}(\mathscr{T}(M))$ which is a well-defined subset of $\mathrm{NF}(\Lambda_s)$, possibly empty.

Example 4.4.2. *We calculate the* r*-normal form of the Taylor expansions from Example 4.2.3:*

1. $\mathrm{NF}(\mathscr{T}(\mathbf{I})) = \mathscr{T}(\mathbf{I}) = \{[\lambda x.[x^{n_1}], \dots, \lambda x.[x^{n_k}]] \mid k \geq 0, \forall i \leq k, n_i \geq 0\}$;
2. $\mathrm{NF}(\mathscr{T}(\Delta)) = \{[\lambda x.[x][x^{m_1}], \dots, \lambda x.[x][x^{m_k}]] \mid k \geq 0, \forall i \leq k, m_i \geq 0\}$;
3. $\mathrm{NF}(\mathscr{T}(\Delta\mathbf{I})) = \mathrm{NF}(\mathscr{T}(\mathbf{I}))$;
4. $\mathrm{NF}(\mathscr{T}(\Omega)) = \emptyset$;
5. $\mathrm{NF}(\mathscr{T}(\lambda x.\Omega)) = \{[\ ]\}$;
6. *for* $\mathbf{A} = (\lambda z.(\lambda y.y)(zz))(xx)$, *we obtain:*
$$\begin{aligned}\mathrm{NF}(\mathscr{T}(\mathbf{A})) = \ & \{[\lambda z.[[\lambda y.[y^{\ell_1}]]([z][z^{m_1}])]([x][x^{n_1}]), \dots, \lambda z.[[\lambda y.[y^{\ell_k}]]([z][z^{m_k}])]([x][x^{n_k}])] \\ & \mid k \geq 0, \forall i \leq k, \ell_i, m_i, n_i \geq 0\}.\end{aligned}$$

We need a connection between reductions of resource terms in a Taylor expansion and reductions of the corresponding λ-term. It is complicated to compute some normal forms of Taylor expansion without having such a result, for example Z and ZB.

Lemma 4.4.3 (Substitution Lemma). *Let $M \in \Lambda$, $V \in \Lambda^V$ and $x \in \mathbb{V}$. Then we have:*

$$\mathscr{T}(M[V/x]) = \bigcup_{t \in \mathscr{T}(M)} \ \bigcup_{[v_1,\dots,v_n] \in \mathscr{T}(V)} t\langle [v_1, \dots, v_n]/x \rangle.$$

Proof. Straightforward induction on the structure of M. □

Lemma 4.4.4. *Let $M, N \in \Lambda$ be such that $M \to_v N$. Then:*

1. *for all $t \in \mathscr{T}(M)$, there exists $\mathcal{T} \subseteq \mathscr{T}(N)$ such that $t \twoheadrightarrow_{\mathrm{r}} \mathcal{T}$;*
2. *for all $t' \in \mathscr{T}(N)$ such that $t' \not\twoheadrightarrow_0 \emptyset$, there exist $t \in \mathscr{T}(M)$ and $\mathcal{T} \in \mathscr{P}_{\mathrm{f}}(\Lambda_s)$ satisfying $t \twoheadrightarrow_{\mathrm{r}} \{t'\} \cup \mathcal{T}$. Moreover such a t is unique.*

Proof. We check that both (1) and (2) hold by induction on a derivation of $M \to_v N$, splitting into cases depending on the reduced redex:

(β_v) If $M = (\lambda x.L)V$ and $N = L[V/x]$ then items (1) and (2) follow by Lemma 4.4.3.

(σ_1) If $M = (\lambda x.M')L_1L_2$ and $N = (\lambda x.M'L_1)L_2$ then

$$\begin{aligned}\mathscr{T}(M) &= \{[\lambda x.t_1, \dots, \lambda x.t_n]s_1s_2 \\ &\qquad \mid n \geq 0, t_i \in \mathscr{T}(M'), s_1 \in \mathscr{T}(L_1), s_2 \in \mathscr{T}(L_2)\}, \\ \mathscr{T}(N) &= \{[\lambda x.t'_1s'_1, \dots, \lambda x.t'_ns'_n]s \\ &\qquad \mid n \geq 0, t'_i \in \mathscr{T}(M'), s'_i \in \mathscr{T}(L_1), s \in \mathscr{T}(L_2)\}.\end{aligned}$$

- For $n \neq 1$, we have $[\lambda x.t_1, \dots, \lambda x.t_n]s_1 s_2 \to_0 \emptyset \subseteq \mathscr{T}(N)$.
- For $n = 1$, we get $[\lambda x.t_1]s_1 s_2 \to_{\sigma_1} [\lambda x.t_1 s_2]s_1$ for $t_1 \in \mathscr{T}(M')$, $s_1 \in \mathscr{T}(L_1)$ and $s_2 \in \mathscr{T}(L_2)$, whence $[\lambda x.t_1 s_2]s_1 \in \mathscr{T}(N)$.

So (1) holds. Concerning (2), note that $[\lambda x.t'_1 s'_1, \dots, \lambda x.t'_n s'_n]s \not\to_0 \emptyset$ entails $n = 1$. Moreover, $\mathscr{T}(M) \ni [\lambda x.t'_1]s s'_1 \to_{\sigma_1} [\lambda x.t'_1 s'_1]s$ since $t'_1 \in \mathscr{T}(M'), s'_1 \in \mathscr{T}(L_2), s \in \mathscr{T}(L_1)$.

(σ_3) If $M = V((\lambda x.L_1)L_2)$ for $V \in \Lambda^V$ and $N = (\lambda x.V L_1)L_2$ then

$$\begin{aligned} \mathscr{T}(M) &= \{[v_1, \dots, v_n]([\lambda x.s_1, \dots, \lambda x.s_m]s) \\ &\qquad \mid n \geq 0, [v_1, \dots, v_n] \in \mathscr{T}(V), i \leq m,\ s_i \in \mathscr{T}(L_1), s \in \mathscr{T}(L_2)\}, \\ \mathscr{T}(N) &= \{[\lambda x.[v^1_1, \dots, v^1_{k_1}]s_1, \dots, \lambda x.[v^n_1, \dots, v^n_{k_n}]s_n]s \\ &\qquad \mid n \geq 0,\ i \leq n, s_i \in \mathscr{T}(L_1), s \in \mathscr{T}(L_2), [v^i_1, \dots, v^i_{k_i}] \in \mathscr{T}(V)\}, \end{aligned}$$

- For $m \neq 1$ or $n \neq 1$, we have $[v_1, \dots, v_n]([\lambda x.s_1, \dots, \lambda x.s_m]s) \to_0 \emptyset \subseteq \mathscr{T}(N)$.
- For $m = n = 1$, we get $[v_1]([\lambda x.s_1]s) \to_{\sigma_3} [\lambda x.[v_1]s_1]s \in \mathscr{T}(N)$.

So (1) holds.

- For $n \neq 1$ or $k_i \neq 1$ for some $i \leq n$, we have that $[\lambda x.[v^1_1, \dots, v^1_{k_1}]s_1, \dots, \lambda x.[v^n_1, \dots, v^n_{k_n}]s_n]s \to_0 \emptyset$.
- For $n = k_1 = 1$, we get $\mathscr{T}(M) \ni [v^1_1]([\lambda x.s_1]s) \to_{\sigma_3} [\lambda x.[v^1_1]s_1]s$.

This proves (2).

In the cases above it is easy to check that t is actually unique. The contextual cases follow straightforwardly from the induction hypothesis. □

As a consequence, we obtain the analogue of Property 3.2.17 for Taylor expansions.

Corollary 4.4.5. *For $M, N \in \Lambda$, $M =_v N$ entails* $\mathrm{NF}(\mathscr{T}(M)) = \mathrm{NF}(\mathscr{T}(N))$.

Proof. It is enough to prove $\mathrm{NF}(\mathscr{T}(M)) = \mathrm{NF}(\mathscr{T}(N))$ for M and N such that $M \to_v N$, indeed the general result follows by confluence of v-reduction. We show the two inclusions:

($\subseteq$) Consider $t \in \mathrm{NF}(\mathscr{T}(M))$, then there exists $t_0 \in \mathscr{T}(M)$ and $\mathcal{T} \in \mathscr{P}_{\mathrm{f}}(\Lambda_s)$ such that $t_0 \twoheadrightarrow_{\mathrm{r}} \{t\} \cup \mathcal{T}$. Since λ^σ_r is strongly normalising (Proposition 4.1.6), we assume wlog $\mathcal{T}$ in r-nf. By Lemma 4.4.4(1), we have $t_0 \twoheadrightarrow_{\mathrm{r}} \mathcal{T}_0 \subseteq \mathscr{T}(N)$ so by confluence of $\to_r$ we get $\mathcal{T}_0 \twoheadrightarrow_{\mathrm{r}} \{t\} \cup \mathcal{T}$ which entails $t \in \mathrm{NF}(\mathscr{T}(N))$ because t is in r-nf.

($\supseteq$) If $t \in \mathrm{NF}(\mathscr{T}(N))$ then there are $s \in \mathscr{T}(N)$ and $\mathcal{T} \in \mathscr{P}_{\mathrm{f}}(\Lambda_s)$ such that $s \twoheadrightarrow_{\mathrm{r}} \{t\} \cup \mathcal{T}$. By Lemma 4.4.4(2), there exists $s_0 \in \mathscr{T}(M)$ and $\mathcal{S} \in \mathscr{P}_{\mathrm{f}}(\Lambda_s)$ satisfying $s_0 \twoheadrightarrow_{\mathrm{r}} \{s\} \cup \mathcal{S}$. Composing the two reductions we get $s_0 \twoheadrightarrow_{\mathrm{r}} \{t\} \cup \mathcal{S} \cup \mathcal{T}$, thus $t \in \mathrm{NF}(\mathscr{T}(M))$ as well. □

We now prove a Context Lemma for Taylor expansions similar as [Barendregt, 1984, Cor. 14.3.20]. In the next lemma we consider head contexts, the same reasoning works for arbitrary contexts.

Lemma 4.4.6 (Context Lemma)**.** *Let $M, N \in \Lambda$. If* $\mathrm{NF}(\mathscr{T}(M)) = \mathrm{NF}(\mathscr{T}(N))$ *then, for all head contexts* $C\llparenthesis - \rrparenthesis$*,* $\mathrm{NF}(\mathscr{T}(C\llparenthesis M \rrparenthesis)) = \mathrm{NF}(\mathscr{T}(C\llparenthesis N \rrparenthesis))$.

Proof. Consider $C\llparenthesis - \rrparenthesis = (\lambda x_1 \dots x_n.\llparenthesis - \rrparenthesis)V_1 \cdots V_k$ for $n, k \geq 0$.

Let us take $t \in \mathrm{NF}(\mathscr{T}(C\llparenthesis M \rrparenthesis))$ and prove that t belongs to $\mathrm{NF}(\mathscr{T}(C\llparenthesis N \rrparenthesis))$, the other inclusion being symmetrical. There exists $t_0 \in \mathscr{T}(C\llparenthesis M \rrparenthesis)$ and $\mathcal{T} \in \mathscr{P}_{\mathrm{f}}(\mathrm{NF}(\Lambda_s))$ such that $t_0 \twoheadrightarrow_{\mathrm{r}} \{t\} \cup \mathcal{T}$. By definition of $C\llparenthesis - \rrparenthesis$ and $\mathscr{T}(-)$, t_0 must have the following shape:

$$t_0 = [\lambda x_1.[\cdots[\lambda x_n.s]\cdots]][v_1^1, \dots, v_{n_1}^1] \cdots [v_1^k, \dots, v_{n_k}^k]$$

where $s \in \mathscr{T}(M)$, $[v_1^i, \dots, v_{n_i}^i] \in \mathscr{T}(V_i)$ where $i \in \{1, \dots, k\}$ and $n_i = \deg_{x_i}(s)$ for otherwise $t_0 \to_0 \emptyset$, which is impossible. By confluence and strong normalisation of $\to_{\mathsf{r}}$ (Proposition 4.1.6), the reduction $t_0 \twoheadrightarrow_{\mathrm{r}} \{t\} \cup \mathcal{T}$ factorises as $t_0 \twoheadrightarrow_{\mathrm{r}} \mathcal{T}_0 \twoheadrightarrow_{\mathrm{r}} \{t\} \cup \mathcal{T}$ where

$$\mathcal{T}_0 = [\lambda x_1.[\cdots[\lambda x_n.\mathrm{NF}_r(s)]\cdots]][v_1^1, \dots, v_{n_1}^1] \cdots [v_1^k, \dots, v_{n_k}^k]$$

and $\mathrm{NF}_r(s) \in \mathscr{P}_{\mathrm{f}}(\mathrm{NF}(\mathscr{T}(M)))$. By hypothesis $\mathrm{NF}_r(s) \in \mathscr{P}_{\mathrm{f}}(\mathrm{NF}(\mathscr{T}(N)))$, therefore there are $\mathcal{S}_1 \in \mathscr{P}_{\mathrm{f}}(\mathscr{T}(N))$ such that $\mathcal{S}_1 \twoheadrightarrow_{\mathrm{r}} \mathrm{NF}_r(s) \cup \mathcal{S}'$, for some $\mathcal{S}'$, and $\mathcal{S}_0 \subseteq \mathscr{T}(C\llparenthesis N \rrparenthesis)$ of shape

$$\mathcal{S}_0 = [\lambda x_1.[\cdots[\lambda x_n.\mathcal{S}_1]\cdots]][v_1^1, \dots, v_{n_1}^1] \cdots [v_1^k, \dots, v_{n_k}^k].$$

So we conclude, for some $\mathcal{S}''$, that $\mathcal{S}_0 \twoheadrightarrow_{\mathrm{r}} \mathcal{T}_0 \cup \mathcal{S}'' \twoheadrightarrow_{\mathrm{r}} \{t\} \cup \mathcal{T} \cup \mathrm{NF}_r(\mathcal{S}'') \subseteq \mathrm{NF}(\mathscr{T}(C\llparenthesis N \rrparenthesis))$. □

5. Investigation of the Böhm Trees

In Chapter 3 we have introduced the Böhm trees of Call-by-Value λ-terms and, in Chapter 4 the Taylor expansion of those same λ-terms. The Taylor expansion has the advantage to contain the quantitative notion of resource while the Böhm trees only contain qualitative information, however there exists a strong connection between those notions of program approximation. Indeed, in [Ehrhard and Regnier, 2006a], the authors show that the Taylor expansion can actually be seen as a resource sensitive version of Böhm trees by demonstrating that the normal form of the Taylor expansion of a λ-term is actually equal to the Taylor expansion of its Böhm tree.

This relation has been proved in Call-by-Name, therefore it is natural to wonder if such a link exists in Call-by-Value. The present chapter aim at answering this question and further investigating the power of the Böhm trees introduced in Section 3.2.

Contrary to the Call-by-Name case, it turns out that for a general λ-term M the normal form of the Taylor expansion $\mathscr{T}(M)$ is different from the Taylor expansion of the Böhm tree $\mathscr{T}(\mathrm{BT}_v(M))$. Indeed $\mathscr{T}(\mathrm{BT}_v(M))$ may contain approximants that are not normal, but whose normal form is however empty. Such approximants, due to their specific shape, do disappear along the reduction. In Section 5.1, we define the normalised Taylor expansion $\mathscr{T}^\circ(\mathrm{BT}_v(M))$ of a Böhm tree (Definition 5.1.4) to solve this problem. And although the result from [Ehrhard and Regnier, 2006a] does not hold verbatim in Call-by-Value, we prove in Theorem 5.1.8 that for any λ-term M the normal form of $\mathscr{T}(M)$ coincide with $\mathscr{T}^\circ(\mathrm{BT}_v(M))$.

In Section 5.2 we investigate further those notions and the consequences of Theorem 5.1.8. In particular, we investigate potential valuability and solvability.

Solvability is a crucial property, namely a λ-term is solvable if it is capable of generating a completely defined result, like the identity, when plugged in a suitable context [Barendregt, 1974, Klop, 1975, Hyland, 1975]. The solvability notion underlines many fundamental concepts such as approximants, Böhm trees, sensible λ-theories...

In Call-by-Name, solvability has been characterised operationally using head reduction (see [Wadsworth, 1976]), and also logically by means of intersection types in [Coppo and Dezani-Ciancaglini, 1978, Coppo and Dezani-Ciancaglini, 1980].
Non-idempotent intersection types have also been used to obtain quantitative informations about solvable terms in [de Carvalho, 2007, de Carvalho, 2018].

Call-by-Value solvability was also characterised, but through Call-by-Name β-reduction

[Paolini and Ronchi Della Rocca, 1999] [Ronchi Della Rocca and Paolini, 2004]. This result was improved in [Paolini et al., 2005] by using a restriction of Call-by-Name reduction. It is really unsatisfactory to not get an internal characterisation of Call-by-Value solvability.

In [Accattoli and Paolini, 2012], the authors extend the Call-by-Value calculus in a calculus with explicit and delayed substitutions: *Value Substitution Calculus.* In this calculus they provide a characterisation of solvable terms as terms having a normal form for a reduction called stratified weak reduction. In [Accattoli and Guerrieri, 2022], the authors gave a quantitative analysis of solvability via intersection types in Value Substitution Calculus.

In Call-by-Name unsolvable terms are contextually equivalent when observing termination, but this is not the case in Call-by-Value. In [García-Pérez and Nogueira, 2016], the authors present the first proof that λ-theories equating all unsolvable terms are inconsistent, this was already mentioned in [Paolini and Ronchi Della Rocca, 1999].

Some of the properties satisfyed in Call-by-Name by solvable terms are in Call-by-Value satisfied by potentially valuable terms, this is the case for the contextual equivalence. Potentially valuable terms are λ-terms that reduce to a value when plugged in the right context. This notion was introduced in [Paolini and Ronchi Della Rocca, 1999], later studied in [Ronchi Della Rocca and Paolini, 2004] and is less restrictive than solvability. A first characterisation of potentially valuable term was given in [Accattoli and Paolini, 2012], in value substitution calculus as terminating terms with respect a weak reduction.[1]

In [Carraro and Guerrieri, 2014], the authors provide an operational characterisation of solvability and potential valuability in λ_v^σ using two subreductions of $\to_v$. They also provide a semantical characterisation of solvable and potentially valuable terms using the relational model of [Ehrhard, 2012]. An interesting consequence of the link between Taylor expansion and Böhm tree, among others, is that all denotational models satisfying the Taylor expansion (eg, the one in [Carraro and Guerrieri, 2014]) equate all λ-terms having the same Böhm tree. From this we can deduce a characterisation of potential valuability in Theorem 5.2.4.

Our Böhm tree definition is actually meaningful since all the λ-terms having the same Böhm tree are operationally indistinguishable (Theorem 5.2.6). As usual this theorem come from the Context Lemma for Böhm trees 5.2.5, but the specificity is that this lemma is proved as a corollary of the Context Lemma for Taylor expansions.

In 5.3, staying in λ_v^σ (see Section 2.3), we provide a partition of the approximants defined in Section 3.2. We use this partition to give a characterisation of solvability: Theorem 5.3.6.

We recall that most of the results we prove are the Call-by-Value analogues of results well-known in Call-by-Name and contained in [Barendregt, 1977, Ch. 10] (for Böhm trees),

[1] In [Accattoli and Paolini, 2012, Accattoli and Guerrieri, 2022] the authors refer at potentially valuable terms as scrutable terms.

in [Boudes et al., 2013] (for Taylor expansion) and in [Ehrhard and Regnier, 2006a] (for the relationship between the two).

The results presented in Sections 5.1 and 5.2 already appeared in [Kerinec et al., 2020] and those from Section 5.3 in [Kerinec et al., 2021].

5.1. Link Between Taylor Expansion and Böhm Trees

5.1.1. Approximants of Resource Calculus

First let's define approximants similar to the ones in Subsection 3.2.1 for the resource calculus (Definition 4.1).

As a consequence of Property 4.1.6, the r-*normal form* of $\mathcal{E} \in \mathscr{P}_{\mathrm{f}}(\Lambda_r)$ always exists. We denote it by $\mathrm{NF}_r(\mathcal{E})$, so $\mathcal{E} \twoheadrightarrow_{\mathrm{r}} \mathrm{NF}_r(\mathcal{E}) \in \mathscr{P}_{\mathrm{f}}(\Lambda_r)$ and there is no $\mathcal{E}'$ such that $\mathrm{NF}_r(\mathcal{E}) \to_{\mathsf{r}} \mathcal{E}'$.

In such a case there is no need to introduce $\perp$ for approximation and we will call *resource approximants* simple terms in r-nf. They admit the following syntactic characterisation:

Definition 5.1.1. *A* resource approximant $a \in \Lambda_s$ *is a simple term generated by the following grammar:*

$$\begin{array}{lcll} a & ::= & b \mid c & \\ b & ::= & [x^n] \mid [\lambda x.a_1, \dots, \lambda x.a_n] \mid [x]ba_1 \cdots a_k & \textit{(with } k, n \geq 0 \textit{ and } x \in \mathbb{V}\textit{)} \\ c & ::= & [\lambda x.a]([y]ba_1 \cdots a_k) & \textit{(with } k \geq 0 \textit{ and } y, x \in \mathbb{V}\textit{)} \end{array}$$

where $[x^n]$ *is the bag* $[\underbrace{x, \dots, x}_{n}]$.

It is easy to check that resource approximants are r-normal forms.

The structure is extremely similar to the one of our approximants for λ-calculus without resources, see Definition 3.2.3.

Example 5.1.2. *The following are examples of resource approximants:*

1. *both* $[\lambda x.[x], \lambda x.[x,x], \lambda x.[x,x,x]]$ *and* $[\lambda x.[x][x,x], \lambda x.[x][x,x,x]]$ *belong to the Taylor expansions of some* λ*-terms, respectively* I *and* Δ*;*

2. $[\lambda x.[x,x,x], \lambda x.[y,y,y]]$ *does not, since* x *and* y *cannot be approximants of the same subterm.*

Lemma 5.1.3. *Let* $t \in \Lambda_s$ *be such that* $t \subset t$ *(see Definition 4.3.1). Then* t *is in* r*-nf if and only if* t *is a resource approximant.*

Proof. Notice that $t \subset t$ guarantees that all terms in each bag occurring in t have similar shape. The proof of the absence of β_r- and σ- redexes, is analogous to the one of Lemma 2.3.6. The bags occurring in $[x]ba_1 \cdots a_k$ and $[\lambda x.a]([y]ba_1 \cdots a_k)$ must be singleton multisets, for otherwise they would have some 0-redexes. □

5.1.2. Taylor Expansion of Böhm Trees

We add $\mathscr{T}(\bot) = \{[\,]\}$ to the rules of definition 4.2.1 to extend the Taylor expansion to $\Lambda_\bot$, in fact to $\mathcal{A}_v$. And we will use a modify definition of the Taylor expansion to obtain resource terms in r-normal form, as discussed in Remark 4.2.4.

Definition 5.1.4.

1. *Let $A \in \mathcal{A}_v$. The* normalised Taylor expansion *of $A \in \mathcal{A}_v$ is defined by induction following 3.2.3:*

$$\begin{array}{rcl} \mathscr{T}^\circ(\bot) &=& \{[\,]\}, \\ \mathscr{T}^\circ(x) &=& \{[x^n] \mid n \geq 0\}, \\ \mathscr{T}^\circ(\lambda x.A') &=& \{[\lambda x.t_1, \ldots, \lambda x.t_n] \mid n \geq 0, \forall i \leq n,\ t_i \in \mathscr{T}^\circ(A')\}, \\ \mathscr{T}^\circ(xBA_1 \cdots A_n) &=& \{[x]t_0 \cdots t_n \mid t_0 \in \mathscr{T}^\circ(B), \forall i \leq n,\ t_i \in \mathscr{T}^\circ(A_i)\}, \\ \mathscr{T}^\circ((\lambda x.A')(yBA_1 \cdots A_n)) &=& \{[\lambda x.s]t \mid s \in \mathscr{T}^\circ(A'),\ t \in \mathscr{T}^\circ(yBA_1 \cdots A_n)\}. \end{array}$$

2. *The* normalised Taylor expansion of $\mathrm{BT}_v(M)$*, written $\mathscr{T}^\circ(\mathrm{BT}_v(M))$, is defined by setting:*

$$\mathscr{T}^\circ(\mathrm{BT}_v(M)) = \bigcup_{A \in \mathcal{A}_v(M)} \mathscr{T}^\circ(A)$$

Example 5.1.5.

1. *Recall from Example 3.2.8 that $\mathcal{A}_v(\mathbf{I}) = \{\bot, \lambda x.\bot, \lambda x.x\}$, therefore*

$$\mathscr{T}^\circ(\mathcal{A}_v(\mathbf{I})) = \{[\,]\} \cup \{[(\lambda x.[\,])^k] \mid k \geq 0\} \cup \{[\lambda x.[x^{n_1}], \ldots, \lambda x.[x^{n_k}]] \mid k, n_1, \ldots, n_k \geq 0\}$$

By Example 4.4.2(1) this is equal to $\mathrm{NF}(\mathscr{T}(\mathbf{I}))$;

2. *Since $\mathcal{A}_v(\Omega) = \emptyset$ we have $\mathscr{T}^\circ(\mathcal{A}_v(\Omega)) = \emptyset = \mathrm{NF}(\mathscr{T}(\Omega))$;*

3. *Also, $\mathcal{A}_v(\Delta) = \{\bot, \lambda x.\bot, \lambda x.xx\}$, so that*

$$\begin{array}{rl} \mathscr{T}^\circ(\mathcal{A}_v(\Delta)) = & \{[\,]\} \\ & \cup\{[(\lambda x.[\,])^k] \mid k \geq 0\} \\ & \cup\{[\lambda x.[x][x^{n_1}], \ldots, \lambda x.[x][x^{n_k}]] \mid k, n_1, \ldots, n_k \geq 0\} \end{array}$$

by Example 4.4.2(2) this is equal to $\mathrm{NF}(\mathscr{T}(\Delta))$;

4. *Finally, examples 3.2.8(4) and 4.4.2(3) and the above item (1) give us* $\mathscr{T}^\circ(\mathcal{A}_v(\Delta\mathbf{I})) = \mathscr{T}^\circ(\mathcal{A}_v(\mathbf{I})) = \mathrm{NF}(\mathscr{T}(\mathbf{I})) = \mathrm{NF}(\mathscr{T}(\Delta\mathbf{I}))$.

We will now prove that for any λ-term M, the normal form of its Taylor expansion is equal to the normalised Taylor expansion of its Böhm tree. This is a key result in the spirit of [Ehrhard and Regnier, 2006a].

Lemma 5.1.6. *Let $M \in \Lambda$:*

1. *If $t \in \mathscr{T}(M)$ and $t \to_{\mathsf{r}} \{t_1\} \cup \mathcal{T}_1$, then there exists $N \in \Lambda$ and $\mathcal{T}_2 \in \mathscr{P}_{\mathrm{f}}(\Lambda_s)$, such that $M \to_v N$ and $\{t_1\} \cup \mathcal{T}_1 \twoheadrightarrow_r \mathcal{T}_2 \subseteq \mathscr{T}(N)$.*

2. *If $t \in \mathrm{NF}(\mathscr{T}(M))$ then there exists M' such that $M \twoheadrightarrow_v M'$ and $t \in \mathscr{T}(M')$.*

3. *If $t, s \in \mathscr{T}(M)$ then $\mathrm{NF}(t) \cap \mathrm{NF}(s) \neq \emptyset$ entails $t = s$.*

4. *If $t \in \mathscr{T}(M) \cap \mathrm{NF}(\Lambda_s)$ then there exists $A \in \mathcal{A}_v$ such that $A \sqsubseteq_v M$ and $t \in \mathscr{T}^\circ(A)$.*

Proof.

(1) Note that $t \to_{\mathsf{r}} \{t_1\} \cup \mathcal{T}_1$ by contracting an r-redex arising from an occurrence of a v-redex in M, so $M \to_v N$ where N is obtained by contracting such a redex occurrence. By Lemma 4.4.4(1) and confluence of $\to_{\mathsf{r}}$, there exists $\mathcal{T}_2 \subseteq \mathscr{T}(N)$ such that $t \twoheadrightarrow_r \{t_1\} \cup \mathcal{T}_1 \twoheadrightarrow_r \mathcal{T}_2$.

(2) Assume that $t \in \mathrm{NF}(\mathscr{T}(M))$, then there are $t_0 \in \mathscr{T}(M)$ and $\mathcal{T} \in \mathscr{P}_{\mathrm{f}}(\Lambda_s)$ such that $t_0 \twoheadrightarrow_r \{t\} \cup \mathcal{T}$. Since $\to_{\mathsf{r}}$ is strongly normalising, we can assume $\mathcal{T} \subseteq \mathrm{NF}(\Lambda_s)$ and choose such a reduction to have maximal length n. We proceed by induction on n to show that the λ-term M' exists. If $n = 0$ then t_0 is in r-nf so just take $t_0 = t$, $\mathcal{T} = \emptyset$ and $M = M'$. Otherwise $n > 0$ and $t_0 \to_{\mathsf{r}} \{t_1\} \cup \mathcal{T}_1 \twoheadrightarrow_r \{t\} \cup \mathcal{T}$ where the second reduction is strictly shorter. By (1) and confluence there exists N such that $M \to_v N$ and $\{t_1\} \cup \mathcal{T}_1 \twoheadrightarrow_r \mathcal{T}_2 \twoheadrightarrow_r \{t\} \cup \mathcal{T}$ for some $\mathcal{T}_2 \subseteq \mathscr{T}(N)$. So, there are $t_2 \in \mathcal{T}_2$ and $\mathcal{T}' \in \mathscr{P}_{\mathrm{f}}(\Lambda_s)$ such that $t_2 \twoheadrightarrow_r \{t\} \cup \mathcal{T}' \subseteq \mathrm{NF}(\mathscr{T}(M))$ we then conclude by applying the induction hypothesis to this reduction shorter than n.

(3) Assume $t_0 \in \mathrm{NF}(t) \subseteq \mathrm{NF}(\mathscr{T}(M))$. By (2) there is a reduction $M \to_v M_1 \to_v \cdots \to_v M_k$ such that $t_0 \in \mathscr{T}(M_k)$. By an iterated application of Lemma 4.4.4(2), we get that t is the unique element in $\mathscr{T}(M)$ generating t_0. Therefore, $t_0 \in \mathrm{NF}(s)$ entails $s = t$.

(4) By structural induction on the normal structure of t (characterised in Lemma 5.1.3: notice that $t \coh t$ by Theorem 4.3.4).

- If $t = [\,]$ then $M \in \Lambda^V$ and there are two subcases: either $M = x$, or $M = \lambda x.M'$ so we simply take $A = \bot$.
- If $t = [x, \ldots, x]$ ($n > 0$ occurrences) then $M = A = x$.
- If $t = [\lambda x.a_1, \ldots, \lambda x.a_n]$ with $n > 0$ then $M = \lambda x.M'$ and $a_i \in \mathscr{T}(M')$ for $i \leq n$. By induction hypothesis, there are approximants $A_i \sqsubseteq_v M'$ such that $a_i \in \mathscr{T}^\circ(A_i)$. Then we set $A = \lambda x.A'$ for $A' = A_1 \sqcup \cdots \sqcup A_n$ which exists because the A_i's are pairwise compatible.
- If $t = [x]ba_1 \cdots a_k$ then $M = xM_0 \cdots M_k$ and $b \in \mathscr{T}(M_0)$ and $a_j \in \mathscr{T}(M_j)$ for $j = 1, \ldots, k$. By induction hypothesis, there are $A_0, \ldots, A_k$ such that $A_i \sqsubseteq_v M_i$ for all i ($i = 0, \ldots, k$), $b \in \mathscr{T}^\circ(A_0)$ and $a_j \in \mathscr{T}^\circ(A_j)$ for $j = 1, \ldots, k$. Moreover $b \in \mathscr{T}^\circ(A_0)$ entails that A_0 is a B-term from the grammar in Definition 3.2.3, therefore we may take $A = xA_0 \cdots A_k \in \mathcal{A}_v$.

- Finally, if $t = [\lambda x.a]([y]ba_1 \cdots a_k)$ then $M = (\lambda x.M')(yM_0 \cdots M_k)$ with $a \in \mathscr{T}(M')$ and $[y]ba_1 \cdots a_k \in \mathscr{T}(yM_0 \cdots M_k)$. Reasoning as in the previous case, we get $yA_0 \cdots A_k \in \mathcal{A}_v$ such that $[y]ba_1 \cdots a_k \in \mathscr{T}^\circ(yA_0 \cdots A_k)$. Moreover, by induction hypothesis, there is $A' \sqsubseteq_v M'$ such that $a \in \mathscr{T}^\circ(A')$. We conclude by taking $A = (\lambda x.A')(yA_0 \cdots A_k)$. □

Lemma 5.1.7. *Let $M \in \Lambda$ and $A \in \mathcal{A}_v$:*

1. *If $A \sqsubseteq_v M$ then $\mathscr{T}^\circ(A) \subseteq \mathrm{NF}(\mathscr{T}(M))$.*
2. *If $\mathscr{T}^\circ(A) \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M))$ then $A \in \mathcal{A}_v(M)$.*

Proof.

(1) If $A = \bot$ then $M \in \Lambda^V$ and $\mathscr{T}^\circ(\bot) = \{[\,]\} \subseteq \mathscr{T}(M) \cap \mathrm{NF}(\Lambda_r)$.

Otherwise, it follows by induction on A exploiting the fact that all simple terms in $\mathscr{T}^\circ(A)$ belong to $\mathscr{T}(M)$ and are already in r-nf.

(2) We proceed by structural induction on A,

- If $A = \bot$, then $\mathscr{T}^\circ(A) = \{[\,]\} \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M))$ entails $M \twoheadrightarrow_v V$ for some value V, therefore we get $\bot \in \mathcal{A}_v(V)$ and we conclude by Lemma 3.2.12.
- If $A = x$ then $\mathscr{T}^\circ(x) = \{[x^n] \mid n \geq 0\} \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M))$ entails $M \twoheadrightarrow_v x$ and we are done.
- If $A = \lambda x.A'$ then

  $\mathscr{T}^\circ(\lambda x.A') = \{[\lambda x.t_1, \ldots, \lambda x.t_n] \mid n \geq 0, \forall i \leq k \;.\; t_i \in \mathscr{T}^\circ(A')\}$.

  So, $\mathscr{T}^\circ(\lambda x.A') \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M))$ implies that $M \twoheadrightarrow_v \lambda x.M'$ for some M' such that $\mathscr{T}^\circ(A') \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M'))$. By induction hypothesis, we get $A' \in \mathcal{A}_v(M')$ and $\lambda x.A' \in \mathcal{A}_v(\lambda x.M')$. By Lemma 3.2.12 we obtain $\lambda x.A' \in \mathcal{A}_v(M)$.
- If $A = xBA'_1 \cdots A'_k$ then

  $\mathscr{T}^\circ(A) = \{[x]t_0 \cdots t_n \mid t_0 \in \mathscr{T}^\circ(B), i = 1, \ldots, k \;.\; t_i \in \mathscr{T}^\circ(A'_i)\}$. In this case, we must have $M \twoheadrightarrow_v xM_0 \cdots M_k$ with $\mathscr{T}^\circ(B) \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M_0))$ and $\mathscr{T}^\circ(A'_i) \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M_i))$ for $i = 1, \ldots, k$. By induction hypothesis $B \in \mathcal{A}_v(A)(M_0)$ and $A'_i \in \mathcal{A}_v(()M_i)$ $\forall i \in \{1, \ldots, k\}$, thus $xBA'_1 \cdots A'_k \in \mathcal{A}_v(xM_0 \cdots M_k) = \mathcal{A}_v(M)$ by Lemma 3.2.12.
- If $A = (\lambda x.A')(yBA'_1 \cdots A'_k)$, then

  $\mathscr{T}^\circ(A) = \{[\lambda x.s]t \mid s \in \mathscr{T}^\circ(A'),\ t \in \mathscr{T}^\circ(yBA'_1 \cdots A'_k)\}$. In this case we get $M \twoheadrightarrow_v (\lambda x.M')(yM_0 \cdots M_k)$ with

  $\mathscr{T}^\circ(A') \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M'))$ and $\mathscr{T}^\circ(yBA'_1 \cdots A'_k) \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(yM_0 \cdots M_k))$. By applying the induction hypothesis, we obtain $A \in \mathcal{A}_v((\lambda x.M')(yM_0 \cdots M_k))$ and once again we conclude by Lemma 3.2.12. □

From the two previous lemmas we deduce the following major theorem.

Theorem 5.1.8. *For all $M \in \Lambda$, we have $\mathscr{T}^\circ(\mathrm{BT}_v(M)) = \mathrm{NF}(\mathscr{T}(M))$.*

Proof.

($\subseteq$) Take $t \in \mathscr{T}^\circ(\mathrm{BT}_v(M))$, then there exists an approximant $A' \in \mathcal{A}_v(M)$ such that $t \in \mathscr{T}^\circ(A')$. As $A' \in \mathcal{A}_v(M)$, there is $M' \in \Lambda$ such that $M \twoheadrightarrow_v M'$ and $A' \sqsubseteq_v M'$. We can therefore apply Lemma 5.1.7(1) to conclude that $t \in \mathrm{NF}(\mathscr{T}(M'))$, which is equal to $\mathrm{NF}(\mathscr{T}(M))$ by Lemma 4.4.5.

($\supseteq$) Assume $t \in \mathrm{NF}(\mathscr{T}(M))$. By Lemma 5.1.6(2) there exists $M' \in \Lambda$ such that $M \twoheadrightarrow_v M'$ and $t \in \mathscr{T}(M')$. By Lemma 5.1.6(4), there is $A \sqsubseteq_v M'$ such that $t \in \mathscr{T}^\circ(A)$. By the conditions above we have $A \in \mathcal{A}_v(M)$, so we conclude that $t \in \mathscr{T}^\circ(\mathrm{BT}_v(M))$. □

5.2. Consequences

In the following we investigate some interesting consequences of Theorem 5.1.8.

Corollary 5.2.1. *For $M, N \in \Lambda$, the following are equivalent:*

1. $\mathrm{BT}_v(M) = \mathrm{BT}_v(N)$*;*
2. $\mathrm{NF}(\mathscr{T}(M)) = \mathrm{NF}(\mathscr{T}(N))$*.*

Proof.

($1 \Rightarrow 2$) If M, N have the same Böhm tree, we can apply Theorem 5.1.8 to get

$$\mathrm{NF}(\mathscr{T}(M)) = \mathscr{T}^\circ(\mathrm{BT}_v(M)) = \mathscr{T}^\circ(\mathrm{BT}_v(N)) = \mathrm{NF}(\mathscr{T}(N)).$$

($1 \Leftarrow 2$) We assume $\mathrm{NF}(\mathscr{T}(M)) = \mathrm{NF}(\mathscr{T}(N))$ and start showing $\mathcal{A}_v(M) \subseteq \mathcal{A}_v(N)$. Take any $A \in \mathcal{A}_v(M)$, by definition we have $\mathscr{T}^\circ(A) \subseteq \mathscr{T}^\circ(\mathrm{BT}_v(M))$, so Lemma 5.1.7(2) entails $A \in \mathrm{BT}_v(N)$.

The converse inclusion being symmetrical, we get $\mathcal{A}_v(M) = \mathcal{A}_v(N)$ which in its turn entails $\mathrm{BT}_v(M) = \mathrm{BT}_v(N)$ by Remark 3.2.15. □

5.2.1. A Short Semantical Digression

In [Carraro and Guerrieri, 2014] the relational model $\mathcal{U}$ of Call-by-Value λ-calculus and resource calculus introduced in [Ehrhard, 2012] is shown to satisfy the σ-rules, so it is actually a model of both λ_v^σ and λ_r^σ.

They also prove that $\mathcal{U}$ satisfies the Taylor expansion in the following technical sense (where $[\![-]\!]$ represents the interpretation function in $\mathcal{U}$):

$$[\![M]\!] = \bigcup_{t \in \mathscr{T}(M)} [\![t]\!] \tag{5.1}$$

As a consequence, we get that the theory of the model $\mathcal{U}$ is included in the theory equating all λ-terms having the same Böhm tree. We conjecture that the two theories coincide.

Theorem 5.2.2. *For $M, N \in \Lambda$, we have:*

$$\mathrm{BT}_v(M) = \mathrm{BT}_v(N) \;\Rightarrow\; [\![M]\!] = [\![N]\!].$$

Proof. Indeed, we have the following chain of equalities:

$$\begin{array}{rll} [\![M]\!] = & \bigcup_{t\in\mathscr{T}(M)}[\![t]\!] & \text{by Equation (5.1)},\\ = & \bigcup_{t\in\mathrm{NF}(\mathscr{T}(M))}[\![t]\!] & \text{as } [\![t]\!] = \bigcup_{s\in\mathrm{NF}_r(t)}[\![s]\!],\\ = & \bigcup_{t\in\mathscr{T}^\circ(\mathrm{BT}_v(M))}[\![t]\!] & \text{by Theorem 5.1.8},\\ = & \bigcup_{t\in\mathscr{T}^\circ(\mathrm{BT}_v(N))}[\![t]\!] & \text{as } \mathrm{BT}_v(M) = \mathrm{BT}_v(N),\\ = & \bigcup_{t\in\mathrm{NF}(\mathscr{T}(N))}[\![t]\!] & \text{by Theorem 5.1.8},\\ = & \bigcup_{t\in\mathscr{T}(N)}[\![t]\!] & \text{as } [\![t]\!] = \bigcup_{s\in\mathrm{NF}_r(t)}[\![s]\!],\\ = & [\![N]\!] & \text{by Equation (5.1)}. \end{array}$$

This concludes the proof. □

Remark 5.2.3. *Observe that $[\,] \in \mathscr{T}(\bot)$ and that $[\![[\,]]\!]_{\{x_1,\dots,x_n\}} = \{([\,]^n, [\,])\}$.*

In the paper [Carraro and Guerrieri, 2014], the authors also prove that $[\![M]\!] \neq \emptyset$ exactly when M is potentially valuable (Definition 2.3.8). From this result, we obtain easily the theorem below.

Theorem 5.2.4. *For $M \in \Lambda$, the following are equivalent:*

1. *M is potentially valuable;*
2. *$\mathrm{BT}_v(M) \neq \varnothing$.*

Proof. It is easy to check that all resource approximants t have non-empty interpretation in $\mathcal{U}$, that is $t \in \mathrm{NF}(\Lambda_s)$ entails $[\![t]\!] \neq \emptyset$. Therefore we have the following chain of equivalences:

$$\begin{array}{rlr} M \text{ potentially valuable} \iff & [\![M]\!] \neq \emptyset & \text{[Carraro and Guerrieri, 2014, Thm. 24]},\\ \iff & \exists t \in \mathscr{T}(M), [\![t]\!] \neq \emptyset & \text{by Equation (5.1)},\\ \iff & \exists s \in \mathrm{NF}(\mathscr{T}(M)), [\![s]\!] \neq \emptyset & \text{as } [\![t]\!] = \bigcup_{s\in\mathrm{NF}_r(t)}[\![s]\!],\\ \iff & \exists s \in \mathrm{NF}(\mathscr{T}(M)) & \text{since } s \in \mathrm{NF}(\Lambda_s) \Rightarrow [\![s]\!] \neq \emptyset,\\ \iff & \exists s \in \mathscr{T}^\circ(\mathrm{BT}_v(M)) & \text{by Theorem 5.1.8},\\ \iff & \exists A \in \mathcal{A}_v(M). & \end{array}$$

This is equivalent to say that $\mathrm{BT}_v(M) \neq \varnothing$. □

5.2.2. About Adequacy

All λ-terms having the same Böhm tree are indistinguishable from an observational point of view (see Definition 2.3.12). As in the Call-by-Name setting, this result follows from the Context Lemma for Böhm trees. The classical proof of this lemma in Call-by-Name is obtained by developing an interesting, but complicated, theory of syntactic continuity (see [Barendregt, 1984, §14.3] and [Amadio and Curien, 1998, §2.4]). Here we bypass this problem completely, and obtain such a result as a corollary of the Context Lemma for Taylor expansions by applying Theorem 5.1.8.

Lemma 5.2.5 (Context Lemma for Böhm trees)**.** *Let $M, N \in \Lambda$. If $\mathrm{BT}_v(M) = \mathrm{BT}_v(N)$ then, for all head contexts $C(\!|-|\!)$, we have $\mathrm{BT}_v(C(\!|M|\!)) = \mathrm{BT}_v(C(\!|N|\!))$.*

Proof.

$$\begin{array}{lrcl}
\textit{Let} & \mathrm{BT}_v(M) = \mathrm{BT}_v(N), & & \\
 & \mathrm{NF}(\mathscr{T}(M)) = \mathrm{NF}(\mathscr{T}(N)) & & \textit{by Corollary 5.2.1}, \\
\forall C(\!|-|\!), & \mathrm{NF}(\mathscr{T}(C(\!|M|\!))) = \mathrm{NF}(\mathscr{T}(C(\!|N|\!))) & & \textit{by Lemma 4.4.6}, \\
\forall C(\!|-|\!), & \mathrm{BT}_v(C(\!|M|\!)) = \mathrm{BT}_v(C(\!|N|\!)) & & \textit{by Corollary 5.2.1.}
\end{array}$$

□

As mentioned in the discussion before Lemma 4.4.6, both the statement and the proof generalise to arbitrary contexts. Thanks to Remark 2.3.14, we only need head contexts in order to prove the following theorem stating that the Böhm tree model we defined is adequate for Plotkin's Call-by-Value λ-calculus:

Theorem 5.2.6. *Let $M, N \in \Lambda$. If $\mathrm{BT}_v(M) = \mathrm{BT}_v(N)$ then $M \equiv N$ (see Definition 2.3.12).*

Proof. Assume, by the way of contradiction, that $\mathrm{BT}_v(M) = \mathrm{BT}_v(N)$ but $M \not\equiv N$. Then, there exists a head context $C(\!|-|\!)$ such that $C(\!|M|\!), C(\!|N|\!) \in \Lambda^o$ and, say, $C(\!|M|\!)$ is valuable while $C(\!|N|\!)$ is not. Since they are closed λ-terms, this is equivalent to say that $C(\!|M|\!)$ is potentially valuable while $C(\!|N|\!)$ is not. By Theorem 5.2.4, $\mathrm{BT}_v(C(\!|M|\!)) \neq \bot$ and $\mathrm{BT}_v(C(\!|N|\!)) = \bot$. As a consequence, we obtain $\mathrm{BT}_v(C(\!|M|\!)) \neq \mathrm{BT}_v(C(\!|N|\!))$ thus contradicting the Context Lemma for Böhm trees (Lemma 5.2.5). □

Remark 5.2.7. *Notice that the converse implication does not hold — for instance it is easy to check that $\Delta(yy) \equiv yy(yy)$ holds, but the two λ-terms have different Böhm trees.*

5.3. Solvability

Our previous approximants are useful to characterise potential valuability (Theorem 5.2.4). However in order to study CbV-solvability (Definition 2.3.8) we will refine this notion.

Definition 5.3.1. *The subsets $\mathcal{S}, \mathcal{U} \subseteq \mathcal{A}_v$ are defined inductively by the grammars:*

$$\begin{array}{llcll}
(\mathcal{S}) & S & ::= & H' \mid R' & \\
 & H' & ::= & x \mid \lambda x.S \mid xHA_1 \cdots A_n & \textit{(for } n \geq 0 \textit{ and } x \in \mathbb{V}\textit{)} \\
 & R' & ::= & (\lambda x.S)(yHA_1 \cdots A_n) & \textit{(for } n \geq 0 \textit{ and } x, y \in \mathbb{V}\textit{)} \\
 & & & & \\
(\mathcal{U}) & U & ::= & \bot \mid \lambda x.U & \\
 & & \mid & (\lambda x.U)(yHA_1 \cdots A_n) & \textit{(for } n \geq 0 \textit{ and } x, y \in \mathbb{V}\textit{)}
\end{array}$$

Note that $\{\mathcal{S}, \mathcal{U}\}$ constitutes a partition of $\mathcal{A}_v$, namely $\mathcal{A}_v = \mathcal{S} \cup \mathcal{U}$ and $\mathcal{S} \cap \mathcal{U} = \emptyset$.

Example 5.3.2.

- $x, \mathbf{I}, x\mathbf{K}\bot, \mathbf{I}(zz), \mathbf{\Delta}(zz), \mathbf{K}(y\mathbf{I}\bot), (\lambda x.(\mathbf{I}(yz)))(zy\bot) \in \mathcal{S}$;
- $\bot, \lambda x_0 \dots x_n.\bot, (\lambda x.\bot)(zz), (\lambda x.\bot)(y\mathbf{II}), (\lambda x.(\lambda y.\bot)(wz))(zw) \in \mathcal{U}$;
- *finally, notice that* $\mathcal{A}_v(\mathbf{\Omega}), \mathcal{A}_v(\mathbf{ZI}), \mathcal{A}_v(\lambda x.\mathbf{\Omega}), \mathcal{A}_v(\mathbf{K}^\star) \subseteq \mathcal{U}$.

We are going to show that the existence of an approximant $A \in \mathcal{A}_v(M)$ of shape S is enough to ensure the CbV-solvability of M. Conversely, when M is unsolvable, $\mathcal{A}_v(M)$ is only populated by approximants of shape U or empty.

Lemma 5.3.3 (Substitution Lemma)**.** *Let $M \in \Lambda$, $\vec{x} = \{x_1, \dots, x_i\} \supseteq \mathrm{FV}(M)$ and $\mathcal{A}_v(M) \neq \emptyset$. Then, for all $j \geq 0$ large enough and $n_1, \dots, n_i \geq j$, we have*

$$M[\mathbf{P}_{n_1}/x_1, \dots, \mathbf{P}_{n_i}/x_i] \twoheadrightarrow_v V, \text{ for some } V \in \Lambda^V \cap \Lambda^o.$$

Moreover, if $x_m H A_1 \cdots A_n \in \mathcal{A}_v(M)$ then we can take $V = \mathbf{P}_\ell$, for $\ell = n_m - n - 1 \geq 0$.

Proof. If $A \in \mathcal{A}_v(M)$, then there is $N \in \Lambda$ such that $M \twoheadrightarrow_v N$ and $A \sqsubseteq_\bot N$. By Lemma 2.3.4, setting $\vartheta = [\mathbf{P}_{n_1}/x_1, \dots, \mathbf{P}_{n_i}/x_i]$, we have $M^\vartheta \twoheadrightarrow_v N^\vartheta \in \Lambda^o$. It suffices to check $N^\vartheta \twoheadrightarrow_v V$.

By structural induction on A:

- if $A = x_m$ (for some $m \in \{1, \dots, i\}$), then $N = x_m$, so $N^\vartheta = \mathbf{P}_{n_m}$ and we are done.
- if $A = \lambda y.A_0$, then $N = \lambda y.N_0$ with $y \notin \vec{x}$ (wlog), whence $N^\vartheta = \lambda y.N_0^\vartheta \in \Lambda^V$.
- if $A = \bot$, and since $\bot \sqsubseteq_\bot N$ entails $N \in \Lambda^V$,then we have either $N = x_m$ or $N = \lambda y.N_0$. Therefore, we proceed as above.
- if $A = x_m H A_1 \cdots A_n$ (for some $m \in \{1, \dots, i\}$), then $A \sqsubseteq_\bot N$ entails $N = x_m N_0 \cdots N_n$ with $H \in \mathcal{A}_v(N_0)$ and $A_r \in \mathcal{A}_v(N_r)$ for all $r\,(r = 1, \dots, n)$. Assuming $j > n$, we obtain

$$\begin{array}{rll} N^\vartheta & = \mathbf{P}_{n_m} N_0^\vartheta \cdots N_n^\vartheta, & \text{by definition of } \vartheta, \\ & \twoheadrightarrow_v \mathbf{P}_{n_m} V_0 \cdots V_n, & \text{by I.H. (induction hypothesis)}, \\ & \twoheadrightarrow_v \mathbf{P}_{n_m - n - 1}, & \text{with } n_m - n - 1 \geq 0, \text{ since } n_m \geq j > n. \end{array}$$

- if $A = (\lambda y.A_0)(xHA_1 \cdots A_n)$ with $x \in \vec{x}$ and, wlog, $y \notin \vec{x}$, then from $A \sqsubseteq_\bot N$, we derive $N = (\lambda y.N_0)N_1$ where $A_0 \in \mathcal{A}_v(N_0)$ and $xHA_1 \cdots A_n \in \mathcal{A}_v(N_1)$. Easy calculations give:

$$\begin{array}{rll} N^\vartheta & = (\lambda y.N_0^\vartheta)N_1^\vartheta & \text{since } y \notin \mathrm{dom}(\vartheta), \text{ then for some } \ell_1 \geq 0 \text{ we get:} \\ & \twoheadrightarrow_v (\lambda y.N_0^\vartheta)\mathbf{P}_{\ell_1} & \text{as the I.H. on } N_1 \text{ and } xHA_1 \cdots A_n \in \mathcal{A}_v(N_1), \\ & \twoheadrightarrow_v N_0^\vartheta[\mathbf{P}_{\ell_1}/y] & \text{by } (\beta_\mathsf{v}), \\ & \twoheadrightarrow_v V & \text{by applying the I.H. to } N_0 \text{ and } \vartheta \circ [\mathbf{P}_{\ell_1}/y]. \end{array}$$

□

Proposition 5.3.4 (Context Lemma)**.** *Let $M \in \Lambda$ and $\{x_1, \dots, x_i\} \supseteq \mathrm{FV}(M)$.*

If $A \in \mathcal{A}_v(M) \cap \mathcal{S}$ then, for all $j \geq 0$ large enough, there is $k \geq 0$ such that for all $n_1, \dots, n_{i+k} \geq j$ we have

$$M[\mathbf{P}_{n_1}/x_1, \dots, \mathbf{P}_{n_i}/x_i]\mathbf{P}_{n_{i+1}} \cdots \mathbf{P}_{n_{i+k}} \twoheadrightarrow_v \mathbf{P}_\ell, \text{ for some } \ell \geq 0.$$

Proof. Since $A \in \mathcal{A}_v(M)$, there exists $N \in \Lambda$ such that $M \twoheadrightarrow_v N$ and $A \sqsubseteq_\perp N$. Now, setting $\vartheta = [\mathbf{P}_{n_1}/x_1, \dots, \mathbf{P}_{n_i}/x_i]$, we have $M^\vartheta \twoheadrightarrow_v N^\vartheta$. Proceed by structural induction on $A \in \mathcal{S}$:

- Case $A = x$. Take $k = 0$ and proceed as in the proof of Lemma 5.3.3.
- Case $A = xHA_1 \cdots A_n$. Again, take $k = 0$ and apply Lemma 5.3.3.
- Case $A = \lambda y.S$. Then $N = \lambda y.N_0$ with $y \notin \vec{x}$ and $S \in \mathcal{A}_v(N_0)$. By induction hypothesis, there is $k' \geq 0$ such that $n_1, \dots, n_{i+k'+1} \geq j$ entails $N_0^\vartheta[\mathbf{P}_{n_{i+1}}/y]\mathbf{P}_{n_{i+2}} \cdots \mathbf{P}_{n_{i+k'+1}} \twoheadrightarrow_v \mathbf{P}_\ell$, for some $\ell \geq 0$. Taking $k = k' + 1$, easy calculations give $(\lambda y.N_0)^\vartheta \mathbf{P}_{n_{i+1}} \cdots \mathbf{P}_{n_{i+k}} \twoheadrightarrow_v \mathbf{P}_\ell$.
- Case $A = (\lambda y.S)(x_m HA_1 \cdots A_n)$ with $m = 1, \dots, i$ and, wlog, $y \notin \vec{x}$. From $A \sqsubseteq_\perp N$, we obtain $N = (\lambda y.N_0)N_1$ with $S \in \mathcal{A}_v(N_0)$, $\mathrm{FV}(N_0) \subseteq \{\vec{x}, y\}$, and $x_m HA_1 \cdots A_n \in \mathcal{A}_v(N_1)$. By induction hypothesis, for all j' large enough, there is k' such that for all $h_1, \dots, h_{i+k'+1} \geq j'$ we have
$N_0[\mathbf{P}_{h_1}/x_1, \dots, \mathbf{P}_{h_i}/x_i, \mathbf{P}_{h_{i+1}}/y]\mathbf{P}_{h_{i+2}} \cdots \mathbf{P}_{h_{i+k'+1}} \twoheadrightarrow_v \mathbf{P}_\ell$, for some $\ell \geq 0$. Therefore, taking $k = k'+1$, we obtain, for all $j \geq j'+n+1$ and $n_1, \dots, n_{i+k} \geq j$, the following:

$$\begin{array}{rcll} N^\vartheta \mathbf{P}_{n_{i+1}} \cdots \mathbf{P}_{n_{i+k}} & = & (\lambda y.N_0^\vartheta)N_1^\vartheta \mathbf{P}_{n_{i+1}} \cdots \mathbf{P}_{n_{i+k}}, & \text{as } y \notin \mathrm{dom}(\vartheta), \\ & \twoheadrightarrow_v & (\lambda y.N_0^\vartheta)\mathbf{P}_{n_m-n-1}\mathbf{P}_{n_{i+1}} \cdots \mathbf{P}_{n_{i+k}}, & \text{by Lemma 5.3.3}, \\ & \to & N_0^\vartheta[\mathbf{P}_{\ell'}/y]\mathbf{P}_{n_{i+1}} \cdots \mathbf{P}_{n_{i+k}}, & \text{setting } \ell' = n_m - n - 1, \\ & \twoheadrightarrow_v & \mathbf{P}_\ell, & \text{by I.H. since } ell' \geq j'. \end{array}$$

□

Corollary 5.3.5. *Let $M \in \Lambda$ and $A \in \mathcal{A}_v(M)$. If $A \in \mathcal{S}$ then M is CbV-solvable.*

Proof. Assume $A \in \mathcal{A}_v(M) \cap \mathcal{S}$ and $\mathrm{FV}(M) = \{\vec{x}\}$. By Proposition 5.3.4, there are $P_1, \dots, P_k \in \Lambda^o$ such that $(\lambda \vec{x}.M)\vec{P} \twoheadrightarrow_v \mathbf{P}_n$ for some $n \geq 0$. By applying the identity n times, we get $(\lambda \vec{x}.M)\vec{P}\mathbf{I}^n \twoheadrightarrow_v \mathbf{I}$. We conclude that M is CbV-solvable. □

Using the previous corollary, we finally provide a characterisation of Call-by-Value solvability.

Theorem 5.3.6 (CbV-solvability)**.** *For $M \in \Lambda$, the following are equivalent:*

1. *M is CbV-solvable.*
2. *There exists an approximant $A \in \mathcal{A}_v(M) \cap \mathcal{S}$.*

Proof.

(1⇒ 2) We will prove the contrapositive : $\mathcal{A}_v(M) \subseteq \mathcal{U}$ or $\mathcal{A}_v(M) = \emptyset \Rightarrow M$ unsolvable.

We will proceed by induction on M:

- if $M = x$ then $\mathcal{A}_v(M) = \{\bot, x\}$. We have $x \in \mathcal{S}$,
- if $M = \lambda x.N$ then $\mathcal{A}_v(M) = \{\bot\} \cup \{\lambda x.A' \mid A' \in \mathcal{A}_v(N)\}$. $\bot \in \mathcal{U}$ and $\{\lambda x.A' \mid A' \in \mathcal{A}_v(N)\} \subseteq \mathcal{U}$ iff $\mathcal{A}_v(N) \subseteq \mathcal{U}$ so by induction hypothesis N is unsolvable, so is M,
- if $M = N_1 N_2$ then
  * either $M \twoheadrightarrow_v x$ and similar to previous case,
  * either $M \twoheadrightarrow_v \lambda x.N$ and similar to previous case,
  * either $M \twoheadrightarrow_v N = xN_0 \dots N_n$ with $N_0 = x$ or $\lambda x.N'_0$. So $\mathcal{A}_v(M) = \mathcal{A}_v(N) = \{xHA_1 \dots A_n \mid H \in \mathcal{A}_v(N_0), A_1 \in \mathcal{A}_v(N_1), \dots, A_n \in \mathcal{A}_v(N_n)\} \subseteq \mathcal{S}$.
  * either $M \twoheadrightarrow_v N = (\lambda x.N')(xN_0 \dots N_n)$ with $N_0 = x$ or $\lambda x.N'_0$. So $\mathcal{A}_v(M) = \mathcal{A}_v(N) = \{(\lambda x.A')(xHA_1 \dots A_n) \mid A' \in \mathcal{A}_v(N'), H \in \mathcal{A}_v(N_0), A_1 \in \mathcal{A}_v(N_1), \dots, A_n \in \mathcal{A}_v(N_n)\}$. $\mathcal{A}_v(N) \subseteq \mathcal{U}$ iff $\mathcal{A}_v(N') \subseteq \mathcal{U}$ and by induction hypothesis N' is unsolvable. So N and M are unsolvable.
  * or $\mathcal{A}_v(M) = \emptyset$, so by Theorem 5.2.4, M is not potentially valuable which implies that M is not solvable.

(2⇒ 1) By Corollary 5.3.5.

□

6. Conclusion

In previous chapters we developed a theory of program approximation in Call-by-Value.

We followed Carraro and Guerrieri who have introduced permutation rules, arising from linear logic proof-nets, in [Carraro and Guerrieri, 2014]. Those rules allow to unblock premature normal forms, due to the reduction rule β_v accepting only values as arguments. This extended calculus is proved, in [Guerrieri et al., 2017], to have the same operational semantics as the traditional Plotkin's Call-by-Value λ-calculus, and can therefore be used to study it.

In this setting, we proposed an original notion of Böhm tree, the first one for Call-by-Value. This definition is made in a similar way as the one in [Amadio and Curien, 1998]:

- introduce $\perp$, representing here an undefined value subterm, and not a general subterm as traditionally;
- define approximants as λ-terms with $\perp$ in normal forms for β_v and σ-rules;
- define a preorder $\sqsubseteq_v$ on approximants (generated by considering $\perp$ smaller than any value);
- using the preorder, define approximants of a term as the set of approximants smaller than this term and its reducts. Such approximants enjoy some properties:
  - Lemma 3.2.11: the "external shape" of an approximant of a given λ-term is stable under reduction;
  - Lemma 3.2.12: two interconvertible λ-terms have the same approximants;
  - Lemma 3.2.13: the set of approximants of a λ-term is empty or an ideal with respect to $\sqsubseteq_v$;
- finally, define the Böhm tree of a term as the supremum of its approximants (which exists due to aforementioned lemma) in Definition 3.2.14.

We showed that sets of approximants $\mathcal{A}_v(M)$ for some λ-term M are exactly the sets of approximants that are directed, downward closed, recursively enumerable and with a finite number of free variables (Theorem 3.2.23).

We compared our approach with Ehrhard's theory of approximation based on differentiation, where the Taylor expansion translates λ-terms in (potentially) infinite sets of terms with resources (see [Ehrhard, 2012]). In [Carraro and Guerrieri, 2014], the Call-by-Value resource calculus was extended with σ-rules to mimic the σ-reductions occurring in M at the level of its resource approximants. Using this extension of resource calculus, we provided a characterisation on sets of resource terms corresponding to the Taylor expansion of a λ-term. To this end, we define a coherence relation $\frown$ between resource terms and proved, in Theorem 4.3.4, that a set of such terms corresponds to the Taylor expansion of a λ-term if and only if it is an infinite clique (with respect to $\frown$) of finite height.

We investigated the relation between Böhm tree and Taylor expansion. In Call-by-Name, the normal form of the Taylor expansion of a λ-term is equal to the Taylor expansion of its Böhm tree (see [Ehrhard and Regnier, 2006a]), this is not the case in Call-by-Value, due to the presence of resource approximants reducing to 0. To take them into account, we defined the normalised Taylor expansion of the Böhm tree of a λ-term. And in Theorem 5.1.8, we proved that it is equal to the normal form of the Taylor expansion: $\forall M \in \Lambda,\ \mathscr{T}^{\circ}(\mathrm{BT}_v(M)) = \mathrm{NF}(\mathscr{T}(M))$.

An interesting consequence, among others, is that all denotational models satisfying the Taylor expansion (for example the one in [Carraro and Guerrieri, 2014]) equate all λ-terms with the same Böhm tree.

Another consequence is a characterisation of potential valuability: a term is potentially valuable if and only if its Böhm tree is not empty (Theorem 5.2.4). Recall that a λ-term is potentially valuable if and only if it reduces to a value when plugged in the right context, this is a larger notion than solvability.

We also provided a characterisation on solvable terms, which are traditionally considered as the meaningful ones. In Call-by-Name they are strongly linked with Böhm trees, since the unsolvable λ-terms are the λ-terms with Böhm tree equal to $\bot$. But the connection is less obvious in Call-by-Value, and in order to characterise solvable terms, we introduce a refinement of our approximants. We partition them into subsets $\mathcal{S}$ and $\mathcal{U}$. A λ-term is then Call-by-Value solvable if and only if it has (at least) one approximant of type $\mathcal{S}$ (Theorem 5.3.6).

Both Call-by-Value potentially valuable and solvable terms already have characterisations in [Accattoli and Paolini, 2012] and in [Carraro and Guerrieri, 2014], respectively using reductions in a version of Call-by-Value with explicit substitutions, and subreductions of $\to_v$ in λ_v^{σ}. We believe that our own characterisations bring a new light on those notions, and that the link with the new definition of Böhm tree is promising.

In particular, the significance of the notion of potential valuability in Call-by-Value is highlighted by the fact that the non-potentially valuable terms are exactly those with empty Böhm trees. We can make a parallel between Call-by-Value non-potentially valuable terms having the smallest Böhm tree possible in Call-by-Value (the empty tree), and Call-by-Name unsolvable terms having the smallest Böhm tree possible in Call-by-Name (a tree containing only $\bot$).

Our definition of Böhm trees provides a syntactic model of Call-by-Value λ-calculus which is adequate: all λ-terms having the same Böhm tree are operationally indistinguish-

able (Theorem 5.2.6). But this model is not fully abstract. Indeed, there are operationally indistinguishable λ-terms with different Böhm trees. An example of such terms is $\Delta(yy)$ and $yy(yy)$ with $\mathrm{BT}_v(\Delta(yy)) = \Delta(yy)$ and $\mathrm{BT}_v(yy(yy)) = yy(yy)$.
The situation looks similar in Call-by-Name where one needs to consider Böhm trees up to possibly infinite η-expansions to capture the λ-theory $\mathcal{H}^*$ and obtain a fully abstract model [Barendregt, 1984, Cor. 19.2.10].

Developing a notion of extensionality for Call-by-Value Böhm trees is certainly interesting and a necessary step, as it might help to describe the equational theory of some extensional denotational model. However, contrary to what happens in Call-by-Name, this will not be sufficient to achieve full abstraction. Indeed, in the previous counterexample extensionality plays no role.

Part II.

Semantics

7. Category Theory

The sole aim of this chapter is to introduce a number of categorical and bicategorical concepts that will be widely used in the remaining chapters. Hence, there is little to no interest reading it in isolation, and it should be better used as a reference. We refer at [Johnson and Yau, 2021] for a more general introduction of those notions.

Section 7.1 contains the categorical background that is needed in the rest of the thesis. We present the principle of coend and a basic theorem of the associated *coend calculus*, i.e. the so-called Yoneda Lemma for coends in Section 7.2. We then provide the construction of a free algebra for an endofunctor in Cat Section 7.3. We conclude this chapter by an introduction of bicategories in Section 7.4.

7.1. Categorical Preliminaries

We provide the definitions of categories (§7.1.1), functors (§7.1.2), Cartesian closed and Seely categories (§7.1.4), and conclude with the Kleisli construction (§7.1.5).

7.1.1. Category

In this section we recall basic definitions of *categories* and *morphisms.*

Definition 7.1.1. *A category $\mathcal{C}$ is given by:*

- *a collection of objects $A, B, \dots \in \mathcal{C}$;*
- *for each pair of objects $A, B \in \mathcal{C}$ a hom-set: a collection $\mathcal{C}(A, B)$ of morphisms $f : A \to B$;*
- *for each triple of objects $A, B, C \in \mathcal{C}$, a composition operation:*

  $$\circ : \mathcal{C}(B, C) \times \mathcal{C}(A, B) \to \mathcal{C}(A, C)$$

  satisfying associativity: for all morphisms f, g, h of the appropriate type:

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

- *for every object $A \in \mathcal{C}$, a morphism $Id_A \in \mathcal{C}(A, A)$ called the identity of A such that:*

  $$\forall f \in \mathcal{C}(A, B),\ Id_B \circ f = f = f \circ Id_A.$$

Notation 7.1.2. *When needed to avoid confusion, we will use the notation $A, B \in Ob(\mathcal{C})$, for A, B being objects of $\mathcal{C}$.*

A category is called *small* if its collection of objects is a set and *locally small* if the hom-sets are sets.

Example 7.1.3.

- *The category* Set *with objects being sets and morphisms being set-theoretic functions.*
- *A preorder category: a category such that for any pair of objects (A, B) there is at most one morphism from A to B. The existence of such a morphism corresponds to the truth of the relation $A \leq B$.*
- *The category* Rel *with objects being sets and morphisms being relations, composition of $R \in \mathrm{Rel}(B, C)$ and $S \in \mathrm{Rel}(A, B)$ is given by their relational product:*

$$R \circ S = \{(a, c) \in A \times C \mid \exists b \in B, (a, b) \in R, (b, c) \in S\}.$$

Definition 7.1.4. *A morphism $f : A \to B$ is called an* isomorphism *if there exists a morphism $f' : B \to A$, such that $f' \circ f = Id_A$ and $f \circ f' = Id_B$.*
In this case we say:

- *that f' is the inverse of f, often denoted by f^{-1};*
- *that the objects A and B are isomorphic, written $A \cong B$;*
- *that f, f^{-1} forms an iso-pair.*

Given a category $\mathcal{C}$ the opposite category $\mathcal{C}^{op}$ is obtained by reversing the morphisms (and no other change). A dual statement on a category, usually denoted with the prefix "co" corresponds to the statement in the opposite category.

Definition 7.1.5. *An* initial object *in a category $\mathcal{C}$ is an object $\bot \in \mathcal{C}$ such that any object $A \in \mathcal{C}$ there is a unique morphism $\bot \to A$.*

In a given category, initial objects (if they exists) are equal up to isomorphism, therefore we can consider them as one unique object: the initial object.

7.1.2. Functors

We can see the different categories as themselves objects of a category. In this case the notion of morphism becomes the one of *functor*. A functor is a morphism between categories preserving the identity and composition.

Definition 7.1.6. *Given two categories $\mathcal{C}, \mathcal{D}$ a* functor *$F : \mathcal{C} \to \mathcal{D}$ consists of:*

- *a map F_0 from the objects of $\mathcal{C}$ to the objects of $\mathcal{D}$;*
- *for each pair of objects A, B of $\mathcal{C}$, a map $F_{A,B} : \mathcal{C}(A, B) \to \mathcal{D}(F_0(A), F_0(B))$;*

- *such that for all $f \in \mathcal{C}(A, B)$ and $g \in \mathcal{C}(B, C)$:*
  - $F_{A,C}(g \circ f) = F_{B,C}(g) \circ F_{A,B}(f)$;
  - $F_{A,A}(Id_A) = Id_{F_0(A)}$.

Given functors $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{E}$ their composition $G \circ F : \mathcal{C} \to \mathcal{E}$ is defined by:

- $(G \circ F)_0 = G_0 \circ F_0$;
- $(G \circ F)_{A,B} = G_{F_0(A),F_0(B)} \circ F_{A,B}$.

We will often omit indexes of functors.

Definition 7.1.7.

- *A functor $F : \mathcal{C} \to \mathcal{D}$ is called* full and faithful *if for each pair $A, B \in \mathcal{C}$,*

  $F : \mathcal{C}(A, B) \to \mathcal{D}(F(A), F(B))$ is bijective. (full if surjective and faithful if injective) "Full and faithful" is sometimes shortened to "fully faithful".

- *A functor going from a category to itself, $F : \mathcal{C} \to \mathcal{C}$, is called an* endofuntor.

- *A functor $F : \mathcal{C} \to \mathcal{D}$ is* injective on objects *if its object function is injective: $F(A) = F(B) \Rightarrow A = B$.*

- *A functor $F : \mathcal{C} \to \mathcal{D}$ is* essentially surjective on objects *if for every object $D \in \mathcal{D}$ there exist an object $C \in \mathcal{C}$ and an isomorphism $F(C) \cong D$ in $\mathcal{D}$.*

- *A functor fully faithful and essentially surjective $F : \mathcal{C} \to \mathcal{D}$ is an* equivalence *between categories $\mathcal{C}$ and $\mathcal{D}$.*

- *An* embedding *is a functor that is both injective on objects and faithful.*

Remark 7.1.8. *Functors preserve isomorphisms: if (f, g) is an iso-pair in $\mathcal{C}$ and $F : \mathcal{C} \to \mathcal{D}$ is a functor, then $(F(f), F(g))$ is an iso-pair in $\mathcal{D}$.*

Definition 7.1.9.

- *A* subcategory *$\mathcal{D}$ of a category $\mathcal{C}$ is a category such that $Ob(\mathcal{D}) \subseteq Ob(\mathcal{C})$ and for all $f : A \to B$ in $\mathcal{D}$, f also in $\mathcal{C}$. There is a inclusion functor $\mathcal{D} \hookrightarrow \mathcal{C}$ that is faithful and injective on objects, so it is an embedding.*

- *A subcategory $\mathcal{D}$ of a category $\mathcal{C}$ is a* full subcategory *if for any A and B in $\mathcal{D}$, every morphism $f : A \to B$ of $\mathcal{C}$ is also in $\mathcal{D}$. In such a case, the inclusion functor $\mathcal{D} \hookrightarrow \mathcal{C}$ is full, so it is a full embedding.*

We call *natural transformations* morphisms between functors:

Definition 7.1.10.

- *Let $\mathcal{C}, \mathcal{D}$ be categories and $F, G : \mathcal{C} \to \mathcal{D}$ be functors. A natural transformation $\mu : F \to G$ is given by a family of morphisms:*

$$\mu_A : F(A) \to G(A) \qquad\qquad \forall A \in \mathcal{C}$$

  such that for all $f : A \to B$ the naturality diagram below commutes:

$$\begin{tikzcd} F(A) \arrow[r, "\mu_A"] \arrow[d, "F(f)"'] & G(A) \arrow[d, "G(f)"] \\ F(B) \arrow[r, "\mu_B"'] & G(B) \end{tikzcd}$$

- *The composition of natural transformations $\mu : F \to G$ and $\rho : G \to H$ is defined componentwise: $(\rho \circ \mu)_A = \rho_A \circ \mu_A$.*
- *A natural transformation μ such that μ_A is an isomorphism for every object A is called a* natural isomorphism*.*

7.1.3. Colimit

Definition 7.1.11 (Cocone, colimit, filtered colimit and cocomplete category)**.**

- *Given a functor $F : \mathcal{D} \to \mathcal{C}$ a* cocone *to F is an object $A \in \mathcal{C}$ and a family $\psi_B : F(B) \to A$ ($\forall B \in \mathcal{C}$) such that $\forall f : B \to C \in \mathcal{D}$ the following diagram commutes:*

$$\begin{tikzcd} F(B) \arrow[rr, "F(f)"] \arrow[dr, "\psi_B"'] & & F(C) \arrow[dl, "\psi_C"] \\ & A & \end{tikzcd}$$

- *A* colimit *for $F : \mathcal{D} \to \mathcal{C}$ is a cocone (L, ϕ) of F such that for any other cocone (A, ψ) of F, there is a unique morphism $u : L \to A$ such that $\forall B \in \mathcal{D}, u \circ \phi_B = \psi_B$, that is:*

$$\begin{tikzcd} F(B) \arrow[rr, "F(f)"] \arrow[dr, "\phi_B"] \arrow[ddr, "\psi_B"'] & & F(C) \arrow[dl, "\phi_C"'] \arrow[ddl, "\psi_C"] \\ & L \arrow[d, dotted, "u"] & \\ & A & \end{tikzcd}$$

- *A* filtered category *$\mathcal{C}$ is a category where every functor has a cocone.*
- *A* filtered colimit *is a colimit of a functor $F : \mathcal{D} \to \mathcal{C}$ where $\mathcal{D}$ is a filtered category.*
- *A category $\mathcal{C}$ is* cocomplete *if every functor $F : \mathcal{D} \to \mathcal{C}$ with $\mathcal{D}$ small has a colimit in $\mathcal{C}$.*

7.1.4. Special Categories

In this subsection we will define, among others, categories that are Cartesian, monoidal, closed...

Definition 7.1.12. *The* product category *$\mathcal{C} \times \mathcal{D}$ of categories $\mathcal{C}$ and $\mathcal{D}$ is such that:*

- *objects are pairs of objects (C, D), where C is an object of $\mathcal{C}$ and D of $\mathcal{D}$;*
- *arrows from (C_1, D_1) to (C_2, D_2) are pairs of arrows (f, g), where $f : C_1 \to C_2$ in $\mathcal{C}$ and $g : D_1 \to D_2$ in $\mathcal{D}$;*
- *composition is defined componentwise: $(f_2, g_2) \circ (f_1, g_1) = (f_2 \circ f_1, g_2 \circ g_1)$;*
- *identities are pairs of identities from the contributing categories: $id_{(C,D)} = (id_C, id_D)$.*

A *bifunctor* is a functor whose domain is the product of two categories: $F : \mathcal{C} \times \mathcal{D} \to \mathcal{E}$.

Definition 7.1.13.

- *A* product of two objects *A_1, A_2 in a category $\mathcal{C}$ is given by:*
  - *an object $A_1 \times A_2$ in $\mathcal{C}$;*
  - *together with two projections $\pi_i : A_1 \times A_2 \to A_i$ (for $i = 1, 2$);*

  such that, for every object B and pair of arrows $f_1 : B \to A_1$ and $f_2 : B \to A_2$, there exists a unique morphism $\langle f_1, f_2 \rangle : B \to A_1 \times A_2$ such that the following diagram commutes:

$$\begin{tikzcd}
 & B \arrow[dl, "f"'] \arrow[d, "\langle f,g \rangle"] \arrow[dr, "g"] & \\
A_1 & A_1 \times A_2 \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & A_2
\end{tikzcd}$$

- *A* terminal object *in a category $\mathcal{C}$ is an object $\mathbf{1}$ such that, for every object $A \in \mathcal{C}$, there exists a unique morphism $!_A : A \to \mathbf{1}$.*
- *A category $\mathcal{C}$ is* Cartesian *if it has a terminal object and every pair of objects admits a product.*

  In this case, for every pair of morphisms $f : A \to C$ and $g : B \to D$, the product map $f \times g : A \times B \to C \times D$ is defined by setting $f \times g = \langle f \circ \pi_1, g \circ \pi_2 \rangle$.

Tensor products are similar to categorical products, but are not unique up to isomorphism and do not necessarily have projections. They equipe monoidal categories.

Definition 7.1.14. *A* monoidal category *is a 6-tuple $\langle \mathcal{C}, \otimes, \mathbf{1}, \gamma, \lambda, \rho \rangle$, where $\mathcal{C}$ is a category, $- \otimes - : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ is a bifunctor called the tensor product, $\mathbf{1}$ is the unit of the tensor product, and γ, ρ, λ are natural isomorphisms (called respectively associator, left unitor and right unitor):*

$$\begin{aligned} \gamma_{A,B,C} :&\quad (A \otimes B) \otimes C &&\to A \otimes (B \otimes C), \\ \lambda_A :&\quad \mathbf{1} \otimes A &&\to A, \\ \rho_A :&\quad A \otimes \mathbf{1} &&\to A. \end{aligned}$$

such that the following diagrams commute:

$$\begin{tikzcd}
(A \otimes \mathbf{1}) \otimes B \arrow[rr, "\gamma_{A,\mathbf{1},B}"] \arrow[dr, "\rho_A \otimes \mathbf{1}_B"'] & & A \otimes (\mathbf{1} \otimes B) \arrow[dl, "\mathbf{1}_A \otimes \lambda_B"] \\
& A \otimes B &
\end{tikzcd}$$

$$\begin{tikzcd}
& (A \otimes B) \otimes (C \otimes D) \arrow[dr, "\gamma_{A \otimes B,C,D}"] & \\
A \otimes (B \otimes (C \otimes D)) \arrow[ur, "\gamma_{A,B,C \otimes D}"] & & ((A \otimes B) \otimes C) \otimes D \arrow[d, "\gamma_{A,B,C} \otimes \mathbf{1}_D"] \\
A \otimes ((B \otimes C) \otimes D) \arrow[u, "\mathbf{1}_A \otimes \gamma_{B,C,D}"'] & & (A \otimes (B \otimes C)) \otimes D \arrow[ll, "\gamma_{A,B \otimes C,D}"']
\end{tikzcd}$$

Definition 7.1.15.

- *A monoidal category is* strict *if γ, ρ, λ are identity natural transformations (i.e. the natural transformations that map each object to the appropiate identity morphism).*

- *An object A^* in a monoidal category is called the* dual of A *if it is equipped with two morphisms called the unit and the counit:*

  $$\eta_A : \mathbf{1} \to A^* \otimes A \qquad\qquad \epsilon_A : A \otimes A^* \to \mathbf{1}$$

  such that:

  $$\lambda_A \circ (\epsilon_A \otimes A) \circ \gamma^{-1}_{A,A^*,A} \circ (A \otimes \eta_A) \circ \rho^{-1}_A = \mathrm{id}_A$$

  $$\rho_{A^*} \circ (A^* \otimes \epsilon_A) \circ \gamma_{A^*,A,A^*} \circ (\eta_A \otimes A^*) \circ \lambda^{-1}_{A^*} = \mathrm{id}_{A^*}.$$

Definition 7.1.16. *A* Symmetric Monoidal Category *is a monoidal category* $\langle \mathcal{C}, \otimes, \mathbf{1}, \gamma, \lambda, \rho \rangle$ *having a natural isomorphism:*

$$\sigma_{A,B} : A \otimes B \cong B \otimes A$$

and and that satisfies the usual conditions on a braiding, which are the commutation of the following diagrams:

$$\begin{tikzcd}
(A \otimes B) \otimes C \arrow[r, "\gamma_{A,B,C}"] \arrow[d, "\sigma_{A,B} \otimes Id_C"'] & A \otimes (B \otimes C) \arrow[r, "\sigma_{A,B \otimes C}"] & (B \otimes C) \otimes A \arrow[d, "\gamma_{B,C,A}"] \\
(B \otimes A) \otimes C \arrow[r, "\gamma_{B,A,C}"'] & B \otimes (A \otimes C) \arrow[r, "Id_B \otimes \sigma_{A,C}"'] & B \otimes (C \otimes A)
\end{tikzcd}$$

$$\begin{tikzcd}
A \otimes \mathbf{1} \arrow[r, "\sigma_{A,\mathbf{1}}"] \arrow[dr, "\lambda_A"'] & \mathbf{1} \otimes A \arrow[d, "\rho_A"] \\
 & A
\end{tikzcd}
\qquad
\begin{tikzcd}
A \otimes B \arrow[r, "\sigma_{A,B}"] \arrow[dr, equal] & B \otimes A \arrow[d, "\sigma_{B,A}"] \\
 & A \otimes B
\end{tikzcd}$$

If each oject in a symmetric monoidal category $\mathcal{C}$ has a dual, $\mathcal{C}$ is called a *compact closed category.*

Definition 7.1.17. *A* Monoidal Closed Category *is a monoidal category* $(\mathcal{C}, \otimes, \mathbf{1})$ *such that for all objects* A, B*:*

- *there exists an object* $A \multimap B$ *that internalised* $\mathcal{C}(A, B)$*;*
- *there exists a morphism* $\mathrm{ev}_{A,B} : A \otimes (A \multimap B) \to B$ *called evaluation;*

such that, for every morphism $f : A \otimes X \to B$ *there exists a unique morphism* $h : X \to (A \multimap B)$ *making the next diagram commutes*

$$\begin{tikzcd}
A \otimes X \arrow[d, "Id_A \otimes h"'] \arrow[dr, "f"] & \\
A \otimes (A \multimap B) \arrow[r, "\mathrm{ev}_{A,B}"'] & B
\end{tikzcd}$$

Definition 7.1.18. *A* Cartesian Closed Category $\mathcal{C}$ *is a category with finite products which is closed with respect to its Cartesian monoidal structure* $(\mathcal{C}, \times, \mathbf{1})$*:*

- *for each* $A, B \in \mathcal{C}$ *there is* $A \times B \in \mathcal{C}$*;*
- *there is a terminal object* $\mathbf{1}$*;*
- *for each* $A, B \in \mathcal{C}$ *there is an* exponential object *internalising* $\mathcal{C}(A, B)$ *and denoted by* A^B*;*

- *for each $A, B \in \mathcal{C}$ there is an evaluation morphism* $\mathrm{ev}_{A,B} : (A^B) \times A \to B$ *similar as the one in Definition 7.1.17.*

  Given $f : C \times A \to B$, we denote by $\Lambda(f) : C \to (A^B)$ the currying *of f, namely the unique morphism such that* $\mathrm{ev}_{A,B} \circ \langle \Lambda(f) \circ \pi_1, \pi_2 \rangle = f$.

In a cartesian closed category a reflexive object *is an object U equipped with morphisms* $\mathrm{app} : U \to U^U$ *and* $\mathrm{lam} : U^U \to U$ *such that* $\mathrm{app} \circ \mathrm{lam} = \mathbf{1}$.

7.1.5. Kleisli Category

In the following, we will use & for the Cartesian product and $\top$ for the terminal object to respect the tradition concerning Seely category. Kleisli categories are naturally associated to a monad T and equivalent to the the categories of free T-algebras (see Section 7.3).

Definition 7.1.19. *A* comonad *over a category $\mathcal{C}$ is a triple $(!, der, dig)$ where:*

- $! : \mathcal{C} \to \mathcal{C}$ *is an endofunctor;*
- *dereliction* $\mathrm{der} : ! \to Id_{\mathcal{C}}$ *and digging* $\mathrm{dig} : ! \to !^2$ *are natural transformations, such that the following diagrams commute:*

$$\begin{tikzcd}
!A \arrow[r, "\mathrm{dig}_A"] \arrow[d, "\mathrm{dig}_A"'] \arrow[dr, equal] & !!A \arrow[d, "!\mathrm{der}_A"] \\
!!A \arrow[r, "\mathrm{der}_{!A}"'] & !A
\end{tikzcd}
\qquad
\begin{tikzcd}
!A \arrow[r, "\mathrm{dig}_A"] \arrow[d, "\mathrm{dig}_A"'] & !!A \arrow[d, "!\mathrm{dig}_A"] \\
!!A \arrow[r, "\mathrm{dig}_{!A}"'] & !!!A
\end{tikzcd}$$

Remark 7.1.20. *A monad is similar but with natural transformations $id_{\mathcal{C}} \to !$ and $!^2 \to !$.*

Definition 7.1.21. *A* Seely category *is a symmetric monoidal closed category $(\mathcal{C}, \otimes, \mathbf{1})$ with finite products $(\mathcal{C}, \&, \top)$ with:*

- *a comonad* $(!, \mathrm{der}, \mathrm{dig})$*;*
- *two natural isomorphisms:*

$$\begin{aligned} m^2_{A,B} :\ !A \otimes !B &\cong\ !(A \& B), \\ m^0 :\ \mathbf{1} &\cong\ !\top, \end{aligned}$$

  making $(!, m) : (\mathcal{C}, \&, \top) \to (\mathcal{C}, \otimes, \mathbf{1})$ a symmetric monoidal functor;
- *such that the following coherence diagram commutes*

$$\begin{tikzcd}
!A \otimes !B \arrow[r, "m"] \arrow[dd, dash, "\mathrm{der}_A \otimes \mathrm{der}_B"'] & !(A \& B) \arrow[d, "\mathrm{der}_{A \otimes B}"] \\
 & !!(A \& B) \arrow[d, "!\langle !\pi_1, !\pi_2 \rangle"] \\
!!A \otimes !!B \arrow[r, "m"'] & !(!A \& !B)
\end{tikzcd}$$

Given a comonad $(!, \mathrm{der}, \mathrm{dig})$ over a category $\mathcal{C}$ you can define for all $g : \ !A \to B$, a morphism $g^! : \ !A \to !B$ by setting $g^! = \ !f \circ \mathrm{dig}_{!A}$.

Definition 7.1.22. *The* coKleisli *category* $\mathcal{C}_!$ *of a comonad* $(!, \mathrm{der}, \mathrm{dig})$ *over a category* $\mathcal{C}$ *is defined by:*

- $\mathcal{C}_!$ *has the same objects as* $\mathcal{C}$*;*
- *for all objects* $A, B \in \mathcal{C}_!$*:* $\mathcal{C}_!(A, B) = \mathcal{C}(!A, B)$*;*
- *the composition of* $\mathcal{C}_!$ *is given by* $f \circ_! g = f \circ g^!$*.*

Theorem 7.1.23 ([Seely, 1989])**.** *Given a Seely category* $\mathcal{C}$ *with a comonad* $(!, \mathrm{der}, \mathrm{dig})$*, its coKleisli category* $\mathcal{C}_!$ *is Cartesian closed.*

7.2. The Coend Calculus

Coends are a universal categorical construction which is at the foundation of several structures. In the particular case we will consider, coends correspond to appropriate quotient sums of sets.

Definition 7.2.1. *Given* $f, g : A \to B$ *two morphisms in a category* $\mathcal{C}$*, their* coequaliser*, if it exists, is the colimit of the diagram formed by these two morphisms.*

Given $f, g : A \to B$ in a category $\mathcal{C}$, we say that they are parallel, since they are both in $\mathcal{C}(A, B)$.

Definition 7.2.2. *Given a category* $\mathcal{C}$ *and a functor* $F\colon \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathrm{Set}$*, the* coend of F *is the coequaliser of the following diagram:*

$$\sum_{A,B \in \mathcal{C}} \mathcal{C}(A, B) \times F(A, B) \rightrightarrows \sum_{A \in \mathcal{C}} F(A, A) \to \int^{A \in \mathcal{C}} F(A, A)$$

where the parallel arrows $\rightrightarrows$ *are given by left and right actions of* F *on morphisms* $f \in \mathcal{C}(B, A)$*.*

Since we work with coends in the category of sets, we have that this coequaliser is actually given by the quotient $\sum_{A \in \mathcal{C}} F(A, A)/\sim$ *where the equivalence relation* $\sim$ *is generated by the rule*

$$C \sim D \iff F(f, B)(C) = F(A, f)(D), \text{ for some } f : B \to A.$$

We denote the coend of F as $\int^{c \in \mathcal{C}} F(c, c)$.

We refer to [Loregian, 2021] for a more detailed presentation of this calculus.

A basic theorem of coend calculus is the *Yoneda Lemma for coends*:

Theorem 7.2.3 (Yoneda, Density Theorem)**.** *Let* $K : \mathcal{C}^{\mathrm{op}} \to \mathcal{D}$ *and* $H : \mathcal{C} \to \mathcal{D}$ *be two functors. We have canonical natural isomorphisms:*

$$K(-) \cong \int^{A \in \mathcal{C}} K(A) \times \mathcal{C}(-, A) \qquad H(-) \cong \int^{A \in \mathcal{C}} H(A) \times \mathcal{C}(A, -).$$

7.3. Algebras of Cat Endofunctors

We will present a construction of free algebra.

Definition 7.3.1. *Let* $\mathbf{F} : \mathrm{Cat} \to \mathrm{Cat}$ *be an endofunctor.*

- *An* algebra for $\mathbf{F}$ *consists of a small category* A *equipped with a functor* $F : \mathbf{F}A \to A$.
- *A* partial $\mathbf{F}$-algebra on a small category A *consists of a pair of a functor and a full embedding* $A \overset{F}{\leftarrow} H \overset{G}{\hookrightarrow} \mathbf{F}(A)$.

Definition 7.3.2 (Free $\mathbf{F}$-Algebras [Kelly, 1980])**.** *Given a functor* $\mathbf{F} : \mathrm{Cat} \to \mathrm{Cat}$ *that preserves colimits of* ω*-chains and a small category* A*, we construct a canonical* $\mathbf{F}$*-algebra as follows. Given a coproduct* $A \sqcup B$*, we denote by* in_A *and* in_B *the associated injections.*

- *First, we define an inductive family of small categories:*
$$D_0 = A, \qquad D_{n+1} = \mathbf{F}\, D_n \sqcup A.$$
- *Then, we construct a family of functors* $\iota_n : D_n \hookrightarrow D_{n+1}$*, again by induction:*
$$\iota_0 = \mathsf{in}_A, \qquad \iota_{n+1} = \mathbf{F}(\iota_n) \sqcup A.$$
- *Finally, we define*
$$D_A = \varinjlim_{n \in \mathbb{N}} D_n.$$

We obtain a canonical algebra map $\iota_A : \mathbf{F}(D_A) \to D_A$.
The small category D_A *is in particular the* free $\mathbf{F}$-algebra on A.

7.4. Bicategorical Preliminaries

We recall the definitions of bicategories (§7.4.1) and of pseudoreflexive objects living in a Cartesian closed bicategory (§7.4.2). We refer to [Borceux, 1994] for a more complete introduction of those notions.

7.4.1. Bicategory

Intuitively, bicategories are categories with “morphisms between morphisms”. The associativity and identity laws for composition of morphisms in a bicategory hold just up to coherent isomorphisms, for this reason we associate the prefix ”pseudo” to usual categorical notions.

Definition 7.4.1. *A* bicategory $\mathbf{C}$ *consists of:*

- *a collection* $\mathrm{ob}(\mathbf{C})$ *of* objects *(denoted by* $A, B, C, \ldots$*), also called* 0-cells*;*
- *for all* $A, B \in \mathrm{ob}(\mathbf{C})$*, a category* $\mathbf{C}(A, B)$*:*

- *objects F in $\mathbf{C}(A,B)$, also written $F\colon A \to B$, are called* 1-cells *or* morphisms *from A to B;*
- *arrows in $\mathbf{C}(A,B)$ are called* 2-cells *or* 2-morphisms *and denoted $\gamma : A \Rightarrow B$;*
- *composition of 2-cells is denoted by $- \bullet -$ and generally called* vertical composition*;*

- *for every $A, B, C \in \mathrm{ob}(\mathbf{C})$, a bifunctor called* horizontal composition*:*

$$\circ_{A,B,C}\colon \mathbf{C}(B,C) \times \mathbf{C}(A,B) \to \mathbf{C}(A,C)$$

  (often the indices A, B, C in $\circ_{A,B,C}$ are omitted).

- *for every $A \in \mathrm{ob}(\mathbf{C})$, a functor $1_A\colon 1 \to \mathbf{C}(A,A)$, where 1 is the category with one object $*$ and one arrow. We slightly abuse notation and identify $1_A(*)$ with the identity 1_A of A;*

- *for all $A, B, C, D \in \mathrm{ob}(\mathbf{C})$ a natural isomorphism $\gamma_{A,B,C,D}$ such that:*

$$\begin{tikzcd}
\mathbf{C}(C,D) \times \mathbf{C}(B,C) \times \mathbf{C}(A,B) \arrow[r, "1_D \times \circ_{A,B,C}"] \arrow[d, "\circ_{B,C,D} \times 1_A"'] & \mathbf{C}(D,C) \times \mathbf{C}(A,C) \arrow[d, "\circ_{A,C,D}"] \arrow[dl, Rightarrow, "\gamma_{A,B,C,D}"'] \\
\mathbf{C}(B,D) \times \mathbf{C}(A,B) \arrow[r, "\circ_{A,B,D}"'] & \mathbf{C}(A,D)
\end{tikzcd}$$

- *for every 1-cell $F\colon A \to B$, two families of invertible 2-cells expressing the identity law:*

$$\lambda_F\colon 1_B \circ F \cong F, \qquad \rho_F\colon F \cong F \circ 1_A.$$

  Such that:

$$\begin{tikzcd}
\mathbf{C}(A,B) \times 1 \arrow[r] \arrow[d, "1 \times 1_A"'] & \mathbf{C}(A,B) \arrow[dl, Rightarrow, shorten=15pt, "\rho_A"] \\
\mathbf{C}(A,B) \times \mathbf{C}(A,A) \arrow[ur, bend right, "\circ_{A,A,B}"'] &
\end{tikzcd}
\qquad
\begin{tikzcd}
1 \times \mathbf{C}(A,B) \arrow[r] \arrow[d, "1_B \times 1"'] & \mathbf{C}(A,B) \arrow[dl, Rightarrow, shorten=15pt, "\lambda_A"] \\
\mathbf{C}(B,B) \times \mathbf{C}(A,B) \arrow[ur, bend right, "\circ_{A,B,B}"'] &
\end{tikzcd}$$

Moreover two additional coherence axioms similar to the ones for monoidal categories Definition 7.1.14 should be satisfied.

Given a bicategory $\mathbf{C}$, we name *opposite bicategory* the bicategory obtained by reversing only the 1-cells, we denote it $\mathbf{C}^{op}$.

A *2-category* is a bicategory where associativity and unit 2-cells are identities.

Example 7.4.2. *A famous example of 2-category is* Cat*: the 2-category where* 0*-cells are the small categories,* 1*-cells are the functors and* 2*-cells are the natural transformations.*

Definition 7.4.3. *Let* $\mathbf{C}$, $\mathbf{D}$ *be two bicategories. A* lax functor $\phi : \mathbf{C} \to \mathbf{D}$ *is:*

- *A function* $\phi : ob(\mathbf{C}) \to ob(\mathbf{D})$*.*
- *For each pair of objects* $A, B \in ob(\mathbf{C})$ *a functor* $\phi_{A,B} : \mathbf{C}(A,B) \to \mathbf{D}(\phi_0(A), \phi_0(B))$*.*
- *For all* $A, B, C \in ob(\mathbf{C})$ *a natural transformation* $\phi_{A,B,C}$*:*

$$\begin{tikzcd}
\mathbf{C}(B,C) \times \mathbf{C}(A,B) \arrow[r, "\phi_{A,B} \times \phi_{B,C}"] \arrow[d, "\circ_{A,B,C}"'] & \mathbf{D}(\phi B, \phi C) \times \mathbf{D}(\phi A, \phi C) \arrow[d, "\circ_{\phi A,\phi B,\phi C}"] \arrow[dl, Rightarrow, "\phi_{A,B,C}"'] \\
\mathbf{C}(A,C) \arrow[r, "\phi_{A,C}"'] & \mathbf{D}(\phi A, \phi C)
\end{tikzcd}$$

  with components $\phi_{F,G} : \phi(G) \circ \phi(F) \to \phi(G \circ F)$*.*

- *For all* $A \in ob(\mathbf{C})$ *a natural transformation* ϕ_A *such that:*

$$\begin{tikzcd}
1 \arrow[r, "1_A"] \arrow[d, equal] & \mathbf{C}(A,A) \arrow[d, "\circ_{\phi A,A}"] \arrow[dl, Rightarrow, "\phi A"'] \\
1 \arrow[r, "1_{\phi A}"'] & \mathbf{D}(\phi A, \phi A)
\end{tikzcd}$$

  with components $\phi_A : \phi(1_A) \to 1\phi_A$

- *It also respects some coherence axioms (see [Leinster, 1998]).*

If the two natural transformation are isomorphisms, ϕ is called a *pseudofunctor*.

Definition 7.4.4. *Let* $\psi, \xi : \mathbf{C} \to \mathbf{D}$ *be two pseudofunctors. A* pseudonatural transformation $P : \psi \Rightarrow \xi$ *is the collection of the following data:*

- *A family of 1-cells* $(P_A : \psi A \to \xi A)_{A \in ob(\mathbf{C})}$*.*
- *for each 1-cell* $F : A \to B$ *an invertible 2-cell* PF *:*

$$\begin{tikzcd}
\psi A \arrow[r, "P_A"] \arrow[d, "\psi F"'] \arrow[dr, phantom, "P_F \Downarrow"] & \xi A \arrow[d, "\xi F"] \\
\psi B \arrow[r, "P_B"'] & \xi B
\end{tikzcd}$$

- *for all $F : A \to B$ and $G : B \to C$:*

$$\begin{tikzcd}
\psi A \arrow[r, "P_A"] \arrow[d, "\psi F"'] \arrow[dr, phantom, "P_F \Downarrow"] & \xi A \arrow[d, "\xi F"] \\
\psi B \arrow[r, "P_B"] \arrow[d, "\psi G"'] \arrow[dr, phantom, "\Downarrow P_G"] & \xi B \arrow[d, "\xi G"] \\
\psi C \arrow[r, "P_C"'] & \xi C
\end{tikzcd}
=
\begin{tikzcd}
\psi A \arrow[r, "P_A"] \arrow[d, "\psi G \circ F"'] \arrow[dr, phantom, "P_{G \circ F} \Downarrow"] & \xi A \arrow[d, "\xi G \circ F"] \\
\psi C \arrow[r, "P_C"'] & \xi C
\end{tikzcd}$$

- *for all $A \in ob(\mathbf{C})$:*

$$\begin{tikzcd}
\psi A \arrow[r, "P_A"] \arrow[d, "\psi 1_A"'] \arrow[dr, phantom, "P_{1_A} \Downarrow"] & \xi A \arrow[d, "\xi 1_A"] \\
\psi B \arrow[r, "P_B"'] & \xi B
\end{tikzcd}
=
\begin{tikzcd}
\psi A \arrow[r, "P_A"] \arrow[d, "1_{\psi A}"'] & \xi A \arrow[d, "1_{\xi A}"] \\
\psi B \arrow[r, "P_B"'] & \xi B
\end{tikzcd}$$

Definition 7.4.5. *Let $P, Q : \psi \Rightarrow \xi$ be two pseudo-natural transformations. A* modification *$\sigma : P \Rightarrow Q$ consists in a family of 2-cells $\sigma_A : P(A) \to Q(A)$ such that:*

$$\begin{tikzcd}
\psi A \arrow[r, bend left, "P_A"] \arrow[r, bend right, "Q_A"'] \arrow[r, phantom, "\sigma \Downarrow"] \arrow[d, "\psi F"'] \arrow[dr, phantom, "Q_F \Downarrow"] & \xi A \arrow[d, "\xi F"] \\
\psi B \arrow[r, bend right, "Q_B"'] & \xi B
\end{tikzcd}
=
\begin{tikzcd}
\psi A \arrow[r, bend left, "P_A"] \arrow[d, "\psi F"'] \arrow[dr, phantom, "P_F \Downarrow"] & \xi A \arrow[d, "\xi F"] \\
\psi B \arrow[r, bend left, "P_B"] \arrow[r, bend right, "Q_B"'] \arrow[r, phantom, "\sigma \Downarrow"] & \xi B
\end{tikzcd}$$

Definition 7.4.6. *Let $\mathbf{C}$ be a bicategory, $C, D \in$ ob$(\mathbf{C})$ and $i : C \to D$ a* 1*-cell, then:*

- *A* pseudoretraction for *i consists of a 1-cell $j : D \to C$ with an invertible 2-cell $\gamma : 1_C \cong j \circ i$.*
- *A* pseudosection for *i consists of a 1-cell $j : D \to C$ with an invertible 2-cell $\beta : i \circ j \cong 1_D$.*

- *A 1-cell $j : D \to C$ is* right adjoint to i *when there exist 2-cells* $\eta : 1_C \Rightarrow j \circ i$ *and* $\epsilon : i \circ j \Rightarrow 1_D$*, satisfying* $(\epsilon i) \circ (i\eta) = 1_C$ *and* $(j\epsilon) \circ (\eta j) = 1_D$*.*

  In this case, we say that i is left adjoint to j *and that the tuple* $\langle i, j, \eta, \epsilon \rangle$ *is an* adjunction*.*

- *If j is both a pseudoretraction and a pseudosection for i, we say that $\langle i, j \rangle$ is an* equivalence *(this is similar to the Definition 7.1.6).*

- *An equivalence that is also an adjunction is called an* adjoint equivalence*.*

Definition 7.4.7. *A* pseudomonad *over a bicategory* $\mathbf{C}$ *is constructed by:*

- *A triple $\langle T, \eta, \mu \rangle$ where $T : \mathbf{C} \to \mathbf{C}$ is a pseudofunctor, and $\eta : 1 \to T$ and $\mu : T^2 \to T$ are pseudonatural transformations, called the multiplication and unit of the pseudomonad.*

- *Invertible modifications:*

$$\begin{tikzcd}
T^3 \arrow[r, "T\mu"] \arrow[d, "\mu T"'] \arrow[dr, phantom, "\sigma \Downarrow"] & T^2 \arrow[d, "\mu"] \\
T^2 \arrow[r, "\mu"'] & T
\end{tikzcd}
\qquad
\begin{tikzcd}
T \arrow[r, "T\eta"] \arrow[d, "\eta T"'] \arrow[dr, equal] & T^2 \arrow[d, "\mu"] \arrow[dl, Rightarrow, shorten >=50%, "\tau 1"'] \\
T^2 \arrow[r, "\mu"'] \arrow[ur, Rightarrow, shorten >=50%, "\tau 2"'] & T
\end{tikzcd}$$

- *And satisfying two additional coherence conditions from [Lack, 2000].*

Every pseudomonad has also an associated *Kleisli bicategory*, which can be defined in complete analogy with the one-dimensional case (see [Cheng et al., 2003]).

In [Fiore et al., 2017], the authors introduce *relative pseudomonads* which are a generalisation of pseudomonads, and admits Kleisli bicategories.

7.4.2. Cartesian Closed Bicategories

We will give a short insight about Cartesian closed bicategories, for a more detailed version of this kind of structures we refer to [Saville, 2020].

Given bicategories $\mathbf{C}_1, \ldots, \mathbf{C}_n$ with $n \in \mathbb{N}$ there exists the *finite product bicategory* $\Pi_{i=1}^{n} \mathbf{C}_i$, defined in the natural way. Given a bicategory $\mathbf{C}$, we define the n-ary diagonal pseudo-functor:

$$\Delta^n : \mathbf{C} \to \underbrace{\mathbf{C} \times \cdots \times \mathbf{C}}_{n}$$

such that:

- on objects $\Delta^n(\mathbf{C}) = \langle \mathbf{C}, \ldots, \mathbf{C} \rangle$;

- for all $C, D \in ob(\mathbf{C})$:

$$\begin{gathered}\Delta^n_{C,D} : \mathbf{C}(C,D) \to \mathbf{C} \times \mathbf{C}(\Delta^n(C), \Delta^n(D)) \\ (F,G) \mapsto ((F,\dots,F),(G,\dots,G)) \\ (\alpha,\beta) \mapsto ((\alpha,\dots,\alpha),(\beta,\dots,\beta)).\end{gathered}$$

The natural isomorphisms $\Delta^n_{F,G}$ and Δ^n_A are the identities.

Definition 7.4.8. *We say that a bicategory* admits all finite products *if for every $n \in \mathbb{N}$ the pseudofunctor Δ^n admits a right pseudoadjoint, this means:*

- *a function $\Pi_n(-) : ob(\mathbf{C}^n) \to ob(\mathbf{C})$;*
- *for every $A_1, \dots, A_n \in ob(\mathbf{C})$, a 1-cell $\pi_{n,i} : \Pi_n\langle A_1, \dots, A_n\rangle \to A_i$ the i-th projection;*
- *for every $B \in ob(\mathbf{C})$ an adjoint equivalence:*

$$\begin{tikzcd} \Pi^n_{i=1}\mathbf{C}(B, A_i) \arrow[r, bend right, "(-)"'] \arrow[r, phantom, "\perp"] & \mathbf{C}(B, \Pi^n_{i=1}\langle A_1, \dots, A_n\rangle) \arrow[l, bend right, "\langle \pi_{n,1}\circ -, \dots, \pi_{n,n}\circ -\rangle"'] \end{tikzcd}$$

where the right adjoint $(-)$ is called the tupling.

Notation 7.4.9. *We use notations:*

- *if $n = 2$, then $\Pi_2(A,B) = A\&B$;*
- *otherwise $\Pi_n\langle A_1, \dots, A_n\rangle = \&^n_{i=1}A_i$.*

Definition 7.4.10. *A* Cartesian Closed Bicategory *is a bicategory that admits all finite products and is closed, that corresponds to the pseudofunctor $-\&B : \mathbf{C} \to \mathbf{C}$ admitting a right pseudoadjoint $-^B$ i.e.:*

- *for every $B \in ob(\mathbf{C})$ function $-^B : ob(\mathbf{C}) \to ob(\mathbf{C})$;*
- *for every $A, B \in ob(\mathbf{C})$, a 1-cell $\mathrm{ev}_{A,B} : A^B\&A \to B$ called the evaluation morphism;*
- *for every $A, B, C \in ob(\mathbf{C})$ an adjoint equivalence:*

$$\begin{tikzcd} \Pi^n_{i=1}\mathbf{C}(A\&B, C) \arrow[r, bend right, "\Lambda(-)"'] \arrow[r, phantom, "\perp"] & \mathbf{C}(A, C^B) \arrow[l, bend right, "\mathrm{ev}_{B,C}\circ(-\times B)"'] \end{tikzcd}$$

where $\Lambda(-)$ denotes the currying functor.

Definition 7.4.11. *A* pseudoreflexive object *in a Cartesian closed bicategory* $\mathbf{C}$ *is given by a tuple $\langle D, i \colon D^D \to D, j \colon D \to D^D, \gamma\rangle$, where D is an object and $\langle j, \gamma\rangle$ a pseudoretraction for i.*

8. 2-dimensional Semantics

A possible generalisation of well known relational semantics is given by *categorification*, where set-theoretic notions are replaced by categorical ones. The concept of a bidimensional semantics for λ-calculus was first presented in [Seely, 1987] and further studied in [Hirschowitz, 2013]. This refines the denotational semantics viewpoint and allows to categorically model *rewriting* (as a 2-cell) ([Fiore and Saville, 2019, Hilken, 1996]).

Here we are interested in a bicategorical model (see Definition 7.4.1), in particular in the bicategory of *distributors* ([Bénabou, 2000]) where sets are replaced with small categories and relations with distributors. Distributors are a natural choice since they can be seen as a categorification of relations between sets. A relation $R \subseteq A \times B$ is equivalent to its characteristic function:

$$\begin{array}{rrcl} \mathcal{X}_R : & A \times B & \to & \{0,1\} \\ & (a,b) & \to & \begin{cases} 1 & \text{if } aRb, \\ 0 & \text{otherwise.} \end{cases} \end{array}$$

The former function naturally induces a functor from $A \times B$ to the 2 elements category. It is then natural to relax the hypothesis and consider functors of the shape $F : B^{\mathrm{op}} \times A \to \mathrm{Set}$ where A and B are arbitrary small categories (see Section 7.1.1). These functors are what we call distributors.

Distributors are *proof-relevant*, in the sense that two objects a, b are mapped to the set $F(a, b)$ of "witnesses" of their relationship. They determine a *weak* 2-dimensional categorical structure in the sense that many former equalities now hold only up to coherent isomorphisms. This is the case for the interpretation of two β-convertible λ-terms.

In [Cattani and Winskel, 2005], the authors presented a distributor induced model of linear logic. Inspired from that work, Fiore, Gambino, Hyland, and Winskel introduced, in [Fiore, 2005, Fiore et al., 2008], the *generalized species of structure* (also studied in [Gambino and Joyal, 2017]). This is a Kleisli bicategory of distributors categorifying the standard multiset-based semantics of λ-calculus as well as Joyal's species of structures ([Joyal, 1986]).

Then in [Tsukada et al., 2017] was presented a syntactic counterpart: the *rigid Taylor expansion* of λ-terms which refines the usual Taylor expansion. The target language of the rigid Taylor expansion is the *rigid resource calculus*, where a permutation of resource terms in a bag leads to a bag distinct but isomorphic to the first one. In particular, this semantics is used to enumerate the reduction paths to normal forms for non-deterministic programs (subsequently, generalised to other effects in [Tsukada et al., 2018]).

Independently, [Mazza et al., 2017] presents a higher categorical approach of intersection types systems using multicategories and discrete distributors, building on the work done in [Melliès and Zeilberger, 2015] and [Hyland, 2017].

Following those works, in [Olimpieri, 2021, Olimpieri, 2020], the author considered a class of bicategories generalising the construction of [Fiore et al., 2008]. He proved that they actually determine categorical models of λ-calculus and can be syntactically presented via intersection types.

Our work builds on the semantic techniques introduced by [Olimpieri, 2021] and should be seen as a step further towards the categorification of the classical theory of λ-calculus, in the sense of [Hyland, 2017].

In this chapter, we aim at setting the scene for the investigations that will happen in future chapters. We introduce some general definitions and results of 2-dimensional categorical semantics of untyped λ-calculus in Section 8.1.

We also present the bicategory of distributors in Section 8.2, originally introduced in [Bénabou, 1973], which will constitute the core of our bicategorical investigations.

The notions presented in this chapter and the followings are issued from a conjoint work with Federico Olimpieri and Giulio Manzonetto, published in [Kerinec et al., 2023].

8.1. Bicategorical Interpretation

Traditionally models of λ-calculus are given in a 1-dimensional categorical framework: a model being a reflexive object of a Cartesian Closed Category (see Definition 7.1.18).

Definition 8.1.1. *Let $\mathcal{U} = (U, Ap, \lambda)$ be a categorical model in the category $\mathcal{C}$. For any λ-term M and adequate $x_1, \dots, x_n \in \mathbb{V}$, the interpretation of M (in $x_1, \dots, x_n$) is a morphism:*

$$[\![M]\!]_{\{x_1,\dots,x_n\}} \in \mathcal{C}(U^n, U) \qquad \textit{(where } U^n = \underbrace{U \times \cdots \times U}_{n}\textit{)}$$

defined by structural induction on M:

$$\begin{array}{lcll} [\![x]\!]_{\{x_1,\dots,x_n\}} & = & \pi_x^{\{x_1,\dots,x_n\}} & \\ [\![\lambda y.N]\!]_{\{x_1,\dots,x_n\}} & = & \lambda \circ \Lambda([\![N]\!]_{\{x_1,\dots,x_n\}\cup\{y\}}) & \textit{(wlog } y \notin \{x_1,\dots,x_n\}\textit{)}, \\ [\![NP]\!]_{\{x_1,\dots,x_n\}} & = & \mathrm{ev}_{U,U} \circ \langle Ap \circ [\![N]\!]_{\{x_1,\dots,x_n\}}, [\![P]\!]_{\{x_1,\dots,x_n\}} \rangle. & \end{array}$$

For more details you can refer to [Amadio and Curien, 1998, §4.6].

Here we explore the counterpart in a 2-dimensional setting. The categorical framework for our semantic investigations is a Cartesian closed bicategory $\mathbf{C}$ (see Definition 7.4.10), where each hom-category $\mathbf{C}(A, B)$ admits all filtered colimits and an initial object $\perp_{A,B}$.

Definition 8.1.2.

- *A* bicategorical model *of λ-calculus, similarly to a categorical one, is given by any pseudoreflexive object $\mathcal{D} = \langle D, i, j, \alpha\rangle$ in $\mathbf{C}$, where $\langle i, j\rangle$ represents the retraction pair and $\alpha : id_{D^D} \cong j \circ i$ (see Definition 7.4.11).*
- *An* extensional bicategorical model *is a bicategorical model where the pseudoretraction carries the structure of an adjoint equivalence (see Definition 7.4.6):*

$$\begin{tikzcd} D^D \arrow[r, bend right, "i"'] \arrow[r, phantom, "\perp"] & D \arrow[l, bend right, "j"'] \end{tikzcd}$$

In this setting, λ-terms are interpreted by mimicking the standard 1-dimensional categorical definition.

Fix a bicategorical model $\mathcal{D} = \langle D, i, j, \gamma\rangle$ living in the bicategory $\mathbf{C}$. For $x_1, \dots, x_n \in \mathbb{V}$, define $\Lambda^o(x_1, \dots, x_n) = \{M \in \Lambda \mid \mathrm{FV}(M) \subseteq \{x_1, \dots, x_n\}\}$ and $\mathsf{len}(\{x_1, \dots, x_n\}) = n$.

Definition 8.1.3. *The* interpretation of a λ-term $M \in \Lambda^o(\vec{x})$ in $\mathcal{D}$ *is a 1-cell:*

$$[\![M]\!]_{\vec{x}} : D^{\&\mathsf{len}(\vec{x})} \to D \qquad\qquad (= (\underbrace{D \,\&\, \cdots \,\&\, D}_{\mathsf{len}(\vec{x})}) \to D)$$

defined by induction on M as follows:

$$\begin{array}{lcll} [\![x_i]\!]_{\vec{x}} & = & \pi^n_i, & \\ [\![\lambda y.N]\!]_{\vec{x}} & = & i \circ \Lambda\big([\![N]\!]_{\vec{x},y}\big) & \quad\textit{(wlog assume } y \notin \vec{x}\textit{)}, \\ [\![NP]\!]_{\vec{x}} & = & \mathrm{ev}_{D,D} \circ \langle j \circ [\![N]\!]_{\vec{x}}, [\![P]\!]_{\vec{x}}\rangle. & \end{array}$$

The definition of the interpretation extends to $\lambda_\perp$-terms (see Chapter 3, more precisely Definition 3.1.5) by setting $[\![\perp]\!]_{\vec{x}} = \perp_{D^{\&n},D}$.

Since we are dealing with Cartesian closed bicategories, the denotation of a λ-term is invariant under β-conversion only up to canonical coherent isomorphisms. The following lemma and theorem are proved in [Olimpieri, 2021].

Lemma 8.1.4 ((de)Substitution). *Consider $M \in \Lambda^o(\vec{x}, y)$ and $N \in \Lambda^o(\vec{x})$, where $y \notin \vec{x} = x_1, \dots, x_n$. The following canonical invertible 2-cell is built out of the Cartesian closed structure:*

$$\mathsf{sub}^{M,y,N} : [\![M[N/y]]\!]_{\vec{x}} \cong [\![M]\!]_{\vec{x},y} \circ \langle 1_{D^{\&n}}, [\![N]\!]_{\vec{x}}\rangle$$

Theorem 8.1.5 (Soundness). *Let $M, N \in \Lambda^o(\vec{x})$ and $\mathcal{D} = \langle D, i, j, \gamma\rangle$ be a bicategorical model.*

- *If $M \to_\beta N$ then we have a canonical invertible 2-cell (interpreting the β-reduction step):*

  $$[\![M \to_\beta N]\!]_{\vec{x}} : [\![M]\!]_{\vec{x}} \cong [\![N]\!]_{\vec{x}}$$

  which is built out of the Cartesian closed structure and the 2-cell α.

- *If $M \to_\eta N$ and the model $\mathcal{D}$ is extensional, then we also have a canonical invertible 2-cell:*

$$[\![M \to_\eta N]\!]_{\vec{x}} : [\![M]\!]_{\vec{x}} \cong [\![N]\!]_{\vec{x}}$$

  built out of the Cartesian closed structure and the 2-cell α.

8.2. Distributors

For now and in following chapters our investigations will take place in the bicategory of *symmetric categorical sequences*. It is the Kleisli bicategory associated to a pseudo-comonad over the bicategory of distributors. We will present its construction in this section.

Definition 8.2.1. *The bicategory* Dist *of distributors (following [Bénabou, 1973] and see Definition 7.4.1):*

- *0-cells are small categories $A, B, C, \ldots$*
- *1-cells $F : A \nrightarrow B$ are functors $F : A^{\text{op}} \times B \to \text{Set}$ called* distributors.
- *2-cells $\alpha : F \Rightarrow G$ are natural transformations.*
- *For fixed 0-cells A and B, the 1 and 2-cells are organised as a category* $\text{Dist}(A, B)$.
- *For $A \in \text{Dist}$, the identity $1_A : A \nrightarrow A$ is defined by $1_A(a, a') = A(a, a')$.*
- *For 1-cells $F : A \nrightarrow B$ and $G : B \nrightarrow C$, the* horizontal composition *is given by*

$$(G \circ F)(a, c) = \int^{b \in B} G(b, c) \times F(a, b).$$

  Associativity and identity laws for this composition are only up to canonical isomorphism. For this reason Dist *is a bicategory [Borceux, 1994].*

- *There is a symmetric monoidal structure on* Dist *given by the Cartesian product of categories: $A \otimes B = A \times B$.*
- *The bicategory of distributors is compact closed and orthogonality is given by taking the opposite category $A^\perp = A^{\text{op}}$.*
- *The linear exponential object between two objects A and B is then defined as $A^{\text{op}} \times B$.*
- $\text{Dist}(A, B) = \text{Cat}(A^{\text{op}} \times B, \text{Set})$ *is a locally small cocomplete category.*

  For $A, B \in \text{Dist}$ the initial object $\bot_{A,B} \in \text{Dist}(A, B)$ is given by the zero distributor *defined by: for all $\langle a, b\rangle \in A \times B, \bot_{A,B}(a, b) = \emptyset$.*

Definition 8.2.2.

1. *Given a functor* $F : A \to B$ *we can define distributors* $\overline{F} : A \nrightarrow B$ *and* $\underline{F} : B \nrightarrow A$ *by setting*
$$\begin{array}{rcl} \overline{F}(a,b) & = & B(F(a),b), \\ \underline{F}(b,a) & = & B(b,F(a)). \end{array}$$
They are adjoint 1-cells in the bicategory Dist.

2. *Given a distributor* $F : A \nrightarrow B$ *the* web *of* F *is the set*
$$|F| = \bigsqcup_{\langle a,b\rangle \in A\times B} F(a,b).$$

Given distributors $F, G : A \nrightarrow B$, we write $F \subseteq G$ if there is a pointwise inclusion $F(a,b) \subseteq G(a,b)$.

Integers and permutations. Given $n \in \mathbb{N}$, define $[n] = \{1,\ldots,n\}$. In particular, we have $[0] = \emptyset$. We denote by $\mathfrak{S}_n$ the *set of permutations over* $[n]$.

Definition 8.2.3. *The category* $\mathbb{P}$ *of* integers and permutations *is defined as follows:*

- *the objects of* $\mathbb{P}$ *are sets of the form* $\{[n] \mid n \in \mathbb{N}\}$*;*
- *the morphisms from* $[n]$ *to* $[m]$ *are given by*
$$\mathbb{P}([n],[m]) = \begin{cases} \mathfrak{S}_n & \textit{if } n = m, \\ \emptyset & \textit{otherwise;} \end{cases}$$
- *composition of* $\mathbb{P}$ *is simply composition of functions and the identity on* $[n]$ *is denoted by* 1_n*.*

The category $\mathbb{P}$ is symmetric strict monoidal, with tensor product given by addition: $[n] \oplus [m] = [n+m]$ meaning:
given $k_1, k_2 \in \mathbb{N}$, $\sigma \in \mathfrak{S}_{k_1}$ and $\tau \in \mathfrak{S}_{k_2}$, we define $\sigma \oplus \tau \in \mathfrak{S}_{k_1+k_2}$ as
$$(\sigma \oplus \tau)(i) = \begin{cases} \sigma(i) & \text{if } 1 \le i \le k_1, \\ \tau(i-k_1)+k_1 & \text{otherwise.} \end{cases}$$

Given $k_1, \ldots, k_n \in \mathbb{N}$ and $\sigma \in \mathfrak{S}_n$, define:
$$\overline{\sigma} \colon [\textstyle\sum_{i\in[n]} k_i] \to [\sum_{i\in[n]} k_{\sigma(i)}] \quad \text{as} \quad \overline{\sigma}(\sum_{r=1}^{l-1} k_r + p) = \sum_{r=1}^{l-1} k_{\sigma(r)} + p,$$
where $l \in [n]$ and $1 \le p \le k_{\sigma(l)}$.

Symmetric strict monoidal completion. Given two lists $\vec{a}$ and $\vec{b}$ over a set A, their *concatenation* is denoted by $\vec{a} \oplus \vec{b}$.

Let A be a small category. For each object $a \in \mathrm{ob}(A)$, the identity morphism on a is denoted by 1_a. The *symmetric strict monoidal completion* $!A$ of A is the category:

- $\mathrm{ob}(!A) = \{\langle a_1, \dots, a_n\rangle \mid a_i \in A \text{ and } n \in \mathbb{N}\}$;
- $!A[\langle a_1, \dots, a_n\rangle, \langle a'_1, \dots, a'_{n'}\rangle] = \begin{cases} \{\langle \sigma, f_i\rangle_{i\in[n]} \mid f_i : a_i \to a'_{\sigma(i)}\ ,\ \sigma \in \mathfrak{S}_n\} & \text{if } n = n', \\ \emptyset & \text{otherwise;} \end{cases}$
- for $f = \langle \sigma, f_i\rangle_{i\in[n]} \colon \vec{a} \to \vec{b}$ and $g = \langle \tau, g_i\rangle_{i\in[n]} \colon \vec{b} \to \vec{c}$ their composition is defined as follows
$$g \circ f = \langle \tau\sigma, g_{\sigma(1)} \circ f_1, \dots, g_{\sigma(n)} \circ f_n\rangle,$$
- for $\vec{a} = \langle a_1, \dots, a_n\rangle \in \mathrm{ob}(!A)$, the identity on $\vec{a}$ is given by $1_{\vec{a}} = \langle 1_n, 1_{a_1}, \dots, 1_{a_n}\rangle$;
- the monoidal structure is given by list concatenation. The tensor product is symmetric, with symmetries given by the morphisms of the shape (for $\sigma \in \mathfrak{S}_n$):
$$\langle \sigma, \vec{1}\rangle \colon \langle a_1, \dots, a_n\rangle \to \langle a_{\sigma(1)}, \dots, a_{\sigma(n)}\rangle.$$

Definition 8.2.4. *Given $\sigma \in \mathfrak{S}_n$ and $\vec{a}_1, \dots, \vec{a}_n \in \mathrm{ob}(!A)$ with $\mathsf{len}(\vec{a}_i) = k_i$, define*
$$\sigma^\star \colon \bigoplus_{i=1}^n \vec{a}_i \to \bigoplus_{i=1}^n \vec{a}_{\sigma(i)} \textit{ as } \langle \overline{\sigma}, 1_{a_1}, \dots, 1_{a_k}\rangle \qquad (\textit{where } k = \sum_{i\in[n]} k_i).$$

Notation 8.2.5. *We introduce the following abbreviations:*
$$!A^n = (!A)^n \quad \textit{and} \quad !A^{\mathrm{op}} = (!A)^{\mathrm{op}}.$$

The former construction naturally determines an endofunctor $! : \mathrm{Cat} \to \mathrm{Cat}$ (i.e. the 2-monad on Cat for strict monoidal categories). We denote by CatSym the Kleisli bicategory of the pseudocomonad over Dist, obtained by lifting ! as shown in [Fiore et al., 2008, Gambino and Joyal, 2017]. This is the bicategory of symmetric categorical sequences.

Definition 8.2.6. *The bicategory of symmetric categorical sequences,* CatSym*:*

- *objects of* CatSym *are the small categories;*
- *for $A, B \in Dist$, we have* $\mathrm{CatSym}(A, B) = Dist(B, !A)$*;*
- *the identity $1_A(\vec{a}, a) =!A(\vec{a}, \langle a\rangle)$;*
- *For $F : A \nrightarrow B$ and $G : B \nrightarrow C$, composition is given by*
$$(G \circ F)(\vec{a}, c) = \int^{\vec{b}\in !B} G(\vec{b}, c) \times F^\#(\vec{a}, \vec{b})$$
where
$$F^\#(\vec{a}, \vec{b}) = \int^{\vec{a}_1, \dots, \vec{a}_{\mathsf{len}(\vec{b})}} \prod_{i=1}^{\mathsf{len}(\vec{b})} F(\vec{a_i}, b_i) \times !A(\bigoplus_{i=1}^{\mathsf{len}(\vec{b})} \vec{a_i}, a);$$

- CatSym *is Cartesian, with Cartesian product the disjoint union* $A \sqcup B$.
  The terminal object is the empty category,
  and the projections are: $\pi_i(\vec{c}, a) = !(A \sqcup B)(\vec{c}, \langle \rho_i(a) \rangle)$*;*
- CatSym *is Cartesian closed, the exponential object being given by* $B^A = !A^{\mathrm{op}} \times B$.

Remark 8.2.7. *This bicategory is biequivalent to the* generalised species of structures *[Fiore et al., 2008, Fiore et al., 2017]. Generalised species of structures is a powerful construction used to categorify, among others, Joyal's Combinatorial Species [Joyal, 1986].*

A functor $F : A \to B$ determines also a pair of distributors:

$$F^\star : !A \nrightarrow B, \qquad\qquad F_\star : !B \nrightarrow A$$

defined by precomposing $\overline{F}, \underline{F}$ (see Definition 8.2.21) with the counit of !.

Proposition 8.2.8 (Seely equivalence)**.** *For all* $A, B \in \mathrm{Cat}$*, we have an equivalence of categories (see Definition 7.4.6)*

$$!(A \sqcup B) \simeq !A \times !B.$$

The proposition above extends to finite products and coproducts of categories $!(A_1 \sqcup \cdots \sqcup A_n) \simeq !A_1 \times \cdots \times !A_n$. We denote the two components of this equivalence respectively as:

$$\begin{array}{rlcl} \mu_0 : & !(A_1 \sqcup \cdots \sqcup A_n) & \to & !A_1 \times \cdots \times !A_n, \\ \mu_1 : & !A_1 \times \cdots \times !A_n & \to & !(A_1 \sqcup \cdots \sqcup A_n). \end{array}$$

9. Intersection Type Distributors

Relational Graph Models, introduced in [Manzonetto and Ruoppolo, 2014], constitute a subclass of relational models, an important class of "traditional" semantics (see [Berline, 2000] for an overview). Their definition is similar to that of usual graph models inside Scott-continuous semantics ([Engeler, 1981]). They allow to characterise qualitative properties such as termination, as well as quantitative ones like the amount of resource needed during computations. In [de Carvalho, 2018], the author showed that, from an element in the interpretation of a λ-term M it is possible to deduce if M is solvable and an upper bound on the number of head reductions needed to reach the principal head normal form.

Inspired by this result, the authors of [Breuvart et al., 2018] proved by simple induction that relational graph models satisfy an Approximation Theorem. It follows that the theory of any relational graph models includes $\mathcal{B}$. They also constructed a relational graph model $\mathcal{E}$ whose theory is exactly $\mathcal{B}$: the proof of $\mathrm{Th}(\mathcal{E}) \subseteq \mathcal{B}$ relies on the fact that $\mathcal{E}$ has countably many atoms, thus the system admits a kind of principal typings. Relational graph models also contains models of $\mathcal{H}^+$ and $\mathcal{H}^\star$ ([Manzonetto and Ruoppolo, 2014]), and a characterisation on relational graph models that are fully abstract for $\mathcal{H}^+$ and $\mathcal{H}^\star$ was given in [Breuvart et al., 2018].

Since the pioneering work of [de Carvalho, 2007] it is clear that relational models can be presented as intersection type assignment systems where the type operator is associative, commutative but not idempotent (see [Bucciarelli et al., 2017, Paolini et al., 2017]). The relational interpretation of a λ-term M is usually given by the set of pairs (Γ, a), composed by an environment Γ and a type a and, such that the model assigns a to M in the environment Γ by following a bunch of rules.

Our present work builds on the semantic techniques introduced in [Olimpieri, 2021]. In that paper, the author presents a type-theoretic bicategorical semantics of λ-calculus, where the models under consideration are free-algebra constructions for an appropriate endofunctor. We extend this approach to a considerably more general notion of bicategorical models by introducing *categorified graph models* (Definition 9.1.1), which are a generalisation of relational graph models (see Section 9.1). The free-algebra models are then just particular (non-extensional) instances of our construction. Categorified graph models can possibly be extensional and we provide some canonical examples, categorifying classical filter models of λ-calculus (see Remark 9.2.7).

In Section 9.2, we observe that, similarly as in [Olimpieri, 2021], those models can be presented via an intersection type system where the intersection is neither commutative nor idempotent. However, permutative actions on the type derivations allow to restore commutativity "up to isomorphism".

In Section 9.3, we prove that the interpretation of a λ-term can be seen as an intersection type distributor. We then observe that the semantics defined by our models is proof-relevant: the interpretation of a λ-term can be thought of as the set of its type derivations and not only its typings. We can see those derivations as witnesses of the reality of the typing judgements. In Subsection 9.3.1, we define the interpretation of a Böhm tree by taking the filtered colimit of the denotations of its finite approximants, which is available in the bicategory Dist.

The results presented in this chapter have been published in [Kerinec et al., 2023]. Our line of thought follows a well-established tradition ([Olimpieri, 2020, Olimpieri, 2021]) that is rooted in De Carvalho's type theoretic presentation of relational semantics ([de Carvalho, 2007]) and, ultimately, in the pioneering work on filter models from [Barendregt et al., 1983].

9.1. Categorified Graph Models

In this section we will introduce categorified graph models, a generalisation of relational graph models (see [Manzonetto and Ruoppolo, 2014]) which are themselves inspired from graph models (see [Engeler, 1981]).

Definition 9.1.1 (Categorified graph pre-models)**.** *A* categorified graph pre-model *consists of a non-empty small category $D \in \mathrm{Cat}$ equipped with a full embedding $\iota : !D^{\mathrm{op}} \times D \hookrightarrow D$ (see Definition 7.1.1).*

Theorem 9.1.2. *Let $\langle D, \iota\rangle$ be a categorified graph pre-model, the canonical pair of symmetric categorical sequences $\langle \iota^\star, \iota_\star\rangle$ induces a pseudoreflexive object structure on D in the bicategory* CatSym *(Definition 7.4.11).*

If moreover ι is essentially surjective on objects, then $\langle \iota^\star, \iota_\star\rangle$ is an adjoint equivalence.

Proof. We use the following notations:

- A, B, C for objects in $!D^{\mathrm{op}} \times D$: $A ::= (\vec{a}, a)$ and $\vec{a} ::= \langle a_1, \ldots, a_n\rangle$,
- Γ, Δ are objects in $!(!D^{\mathrm{op}} \times D)$, $\Gamma ::= \langle A_1, \ldots, A_n\rangle$,
- $\iota\langle A_1, \ldots, A_n\rangle = \langle \iota(A_1), \ldots, \iota(A_n)\rangle$,
- $\vec{a} \multimap a$ corresponds to $\iota(\vec{a}, a)$.

$$\iota^\star(\Gamma, a) = \sum_{B \ s.t. \ \Gamma = \langle B\rangle} D(\iota(B), a) \qquad\qquad \iota_\star(\vec{a}, A) = !D(\vec{a}, \langle \iota(A)\rangle)$$

$$\begin{aligned}
(\iota_\star \circ \iota^\star)(\Delta, A) &= \int^{\vec{c}} !D(\vec{c}, \langle \iota(A) \rangle) \times \int^{\Delta_1 \ldots \Delta_{\mathsf{len}(\vec{c})}} !(!D^{\mathrm{op}} \times D)(\Delta, \textstyle\sum_i \Delta_i) \\
&\qquad\qquad \times \textstyle\prod_i \iota^\star(\Delta_i, c_i) \\
&\cong \int^{\vec{c}} !D(\vec{c}, \langle \iota(A) \rangle) \times \int^{A_1 \ldots A_{\mathsf{len}(\vec{c})}} !(!D^{\mathrm{op}} \times D)(\Delta, \langle A_1, \ldots, A_{\mathsf{len}(\vec{c})} \rangle) \\
&\qquad\qquad \times \textstyle\prod_i D(\iota(A_i), c_i) \\
&\cong \int^{B} !(!D^{\mathrm{op}} \times D)(\Delta, \langle B \rangle) \times D(\iota(B), \iota(A)) \\
&\cong \int^{B} !(!D^{\mathrm{op}} \times D)(\Delta, \langle B \rangle) \times (!D^{\mathrm{op}} \times D)(B, A) \\
&\cong !(!D^{\mathrm{op}} \times D)(\Delta, \langle A \rangle) \\
&\cong 1_{!D^{\mathrm{op}} \times D}(\Delta, A).
\end{aligned}$$

□

Definition 9.1.3. *We call the bicategorical model $\langle D, \iota^\star, \iota_\star, \gamma \rangle$ obtained in Theorem 9.1.2 a* categorified graph model.

If ι is essentially surjective on objects, then the induced model is extensional (i.e. the pseudoretraction carries the structure of an adjoint equivalence, Definition 8.1.2).

9.2. System $R_{\to}$: Categorified Graph Models in Logical Form

We now show that the model induced by a categorified graph pre-model can be presented as a non-idempotent intersection type system. We fix an arbitrary categorified graph pre-model $\langle D, \iota \rangle$.

The syntactic presentation of categorified graph models is based on the intuition that, given a simple type A, the elements of $![\![A]\!]$ can be seen as *resource approximations* of the type $!A$.

Now, while $!A$ represents the type of a resource that can be used *ad libitum*, a list $\langle a_1, \ldots, a_k \rangle \in ![\![A]\!]$ should be thought of as a choice of *exactly* k copies of resources of type A. In fact, the list $\langle a_1, \ldots, a_k \rangle$ corresponds to a type itself, in the form of an *intersection type* where the intersection operator is not idempotent: $a \cap a \neq a$. The intersection constructor $a \cap b$ is indeed given by the tensor product of $!A$, that is, list concatenation (denoted here by $\oplus$):

$$a_1 \cap \cdots \cap a_k := \langle a_1 \rangle \oplus \cdots \oplus \langle a_k \rangle = \langle a_1, \ldots, a_k \rangle.$$

Similarly, the elements populating $[\![!A \multimap B]\!] = ![\![A]\!]^{\mathrm{op}} \times [\![B]\!]$ can be seen as *arrow types* $\vec{a} \multimap b$.

We shall prove that this type-theoretic correspondence is more than just an analogy: the interpretation of a λ-term in a categorified graph model living in CatSym actually corresponds to the collection of its type derivations in the associated intersection type system (cf. Theorem 9.3.5). Such a type system is *strict* in the sense of [Bakel, 2011], hence the intersections only appear on the left hand-side of an arrow—not as independent types. This reflects the position of the promotion $!(-)$ in the linear logic translation of intuitionistic arrow $A \to B = !A \multimap B$ in [Girard, 1987]. Strictness is also needed to obtain a syntax-directed type system, as λ-calculus does not have a syntactic constructor corresponding to the introduction of an intersection type.

This line of thought can be extended to the untyped setting, by looking at categorified graph models as *categories of types.* Indeed, we can understand the embedding $\iota : !D \times D \hookrightarrow D$ as a way of defining "arrow types" in D, by simply letting $\langle a_1, \dots, a_k \rangle \multimap a := \iota(\langle a_1, \dots, a_k \rangle, a)$. The intersection type constructor will be given again by the tensor product of $!D$.

Standard intersection type systems usually come equipped with a *subtyping preorder* $\preceq$, which in our setting becomes a *category.* Our categorical subtyping is given by morphisms between elements of D, thus we prefer the notation $\to$, rather than $\preceq$. These morphisms are *witnesses* of the subtyping relation. Our approach gives then a sort of *operational* subtyping: morphisms in the category of types D specify which operations are allowed on a list of resources.

Definition 9.2.1. *We define System $R^D_\to$, which is parametric on a categorified graph pre-model D:*

- *The objects of D are seen as* intersection types *and given $\langle \vec{a}, a \rangle \in !D^{\mathrm{op}} \times D$, we set $\vec{a} \multimap a = \iota(\langle \vec{a}, a \rangle)$.*

  As usual, we assume that the operation $\multimap$ is right-associative:

  $$a \multimap b \multimap c = a \multimap (b \multimap c).$$

- *Subtyping in System $R^D_\to$ is given by morphisms in the category of types D.*

- *Finite lists of intersection types are called* (type) environments *and denoted by Γ, Δ. Formally, type environments of length n are objects of the category $!D^n$, the n-fold product of $!D$.*

- *Since $!D$ is monoidal, the category $!D^n$ of type environments has a* tensor product*:*

  $$\langle \vec{a}_1, \dots, \vec{a}_n \rangle \otimes \langle \vec{b}_1, \dots, \vec{b}_n \rangle = \langle \vec{a}_1 \oplus \vec{b}_1, \dots, \vec{a}_n \oplus \vec{b}_n \rangle$$

  This tensor product inherits all the structure from $\oplus$: it is symmetric strict.

- *A* derivation π *of System $R^D_\to$, denoted $\pi \in R^D_\to$, is constructed via the inference rules given in Figure 9.1. The λ-terms are not included in the definition of a derivation (therefore we use a different color for them in the figure) we add them here for a better understanding.*

- *Actions of morphisms on derivations are defined in Figure 9.2.*

$$\frac{f : a' \to a}{x_1 : \langle\rangle, \dots, x_i : \langle a' \rangle, \dots, x_n : \langle\rangle \vdash x_i : a}\ ax \qquad \frac{\Delta, x : \vec{a} \vdash M : a \qquad f : (\vec{a} \multimap a) \to b}{\Delta \vdash \lambda x.M : b}\ lam$$

$$\frac{\Gamma_0 \vdash M : \langle a_1, \dots, a_k \rangle \multimap a \quad (\Gamma_i \vdash N : a_i)_{i=1}^{k} \quad \eta : \Delta \to \bigotimes_{i=0}^{k} \Gamma_i}{\Delta \vdash MN : a}\ app$$

Figure 9.1.: Derivations and Typing of System $R^D_{\to}$.

Notation 9.2.2.

- *We will often keep the parameter D implicit and just write $R_{\to}$ for $R^D_{\to}$.*
- *If π is a derivation of $\Gamma \vdash a$ we write $\pi \triangleright \Gamma \vdash a$ (similarly for $\Gamma \vdash M : a$ we write $\pi \triangleright \Gamma \vdash M : a$).*

Reminder of [Olimpieri, 2021]. Given a non-empty small category A, we will construct D_A as the free-algebra over A for the endofunctor $!-^{\mathrm{op}} \times - : \mathrm{Cat} \to \mathrm{Cat}$ (see Definition 7.3.1).

Let us denote by G_A the multigraph where nodes are given by elements of the set Ty_A, inductively defined by the grammar:

$$(\mathsf{Ty}_A) \qquad a, b, c \ ::= \ o \in A \mid \langle a_1, \dots, a_k \rangle \multimap a,$$

and arrows are inductively generated as shown in Figure 9.3 (a multigraph is composed by a set of vertices and sets of edges for each pair of vertices).

We construct D_A by taking as objects the elements of Ty_A and as morphisms the arrows of G_A (in Figure 9.3). We define the *composition* by:

- if $f : o \to o'$ and $g : o' \to o''$ then $g \circ f : o \to o''$,
- if $\langle \sigma, \vec{f} \rangle \multimap f : (\vec{a} \multimap a) \to (\vec{a}' \multimap a')$ and $\langle \theta, \vec{g} \rangle \multimap g : (\vec{a}' \multimap a') \to (\vec{a}'' \multimap a'')$ then $(\langle \sigma, \vec{f} \rangle \multimap f) \circ (\langle \theta, \vec{g} \rangle \multimap g) : (\vec{a} \multimap a) \to (\vec{a}'' \multimap a'')$.

We denote by $i_A : A \hookrightarrow D_A$ the canonical inclusion. We also have a canonical full embedding $\iota_A : !D_A^{\mathrm{op}} \times D_A \hookrightarrow D_A$ defined by the map $\langle \vec{a}, a \rangle \mapsto \vec{a} \multimap a$.

The free algebra construction and the associated full embedding determine a pseudoreflexive object in CatSym – i.e. a categorified graph model – as detailed in [Olimpieri, 2021].

$$\left(\frac{f : a' \to a}{\langle\rangle, \dots, \langle a'\rangle, \dots, \langle\rangle \vdash a}\right)\{g : b \to a'\} \quad = \quad \frac{f \circ g : b \to a}{\langle\rangle, \dots, \langle b\rangle, \dots, \langle\rangle \vdash a}$$

$$\left(\frac{\begin{array}{c}\pi \\ \vdots \\ \Delta, \vec{a} \vdash a\end{array} \qquad f : (\vec{a} \multimap a) \to b}{\Delta \vdash b}\right)\{\eta\} \quad = \quad \frac{\begin{array}{c}\pi\{\eta \oplus \langle 1\rangle\} \\ \vdots \\ \Delta', \vec{a} \vdash a\end{array} \qquad f : (\vec{a} \multimap a) \to b}{\Delta' \vdash \vec{a} \multimap a}$$

$$\left(\frac{\begin{array}{c}\pi_0 \\ \vdots \\ \Gamma_0 \vdash \vec{a} \multimap a\end{array} \qquad \left(\begin{array}{c}\pi_i \\ \vdots \\ \Gamma_i \vdash a_i\end{array}\right)_{i=1}^{k} \qquad \theta : \Delta \to \bigotimes_{j=0}^{k} \Gamma_j}{\Delta \vdash a}\right)\{\eta\} \quad = \quad \frac{\begin{array}{c}\pi_0 \\ \vdots \\ \Gamma_0 \vdash \vec{a} \multimap a\end{array} \qquad \left(\begin{array}{c}\pi_i \\ \vdots \\ \Gamma_i \vdash a_i\end{array}\right)_{i=1}^{k} \qquad \theta \circ \eta}{\Delta' \vdash a}$$

where $\vec{a} = \langle a_1, \dots, a_k\rangle$ and $\eta : \Delta' \to \Delta$.

(a) Right action on derivations.

$$[g : a \to b]\left(\frac{f : a' \to a}{\langle\rangle, \dots, \langle a'\rangle, \dots, \langle\rangle \vdash a}\right) = \frac{g \circ f : a' \to b}{\langle\rangle, \dots, \langle a'\rangle, \dots, \langle\rangle \vdash b}$$

$$[g : a \to b]\left(\frac{\begin{array}{c}\pi \\ \vdots \\ \Delta, \vec{a} \vdash a'\end{array} \qquad f : (\vec{a} \multimap a) \to a'}{\Delta \vdash a}\right) \quad = \quad \frac{\begin{array}{c}\pi \\ \vdots \\ \Delta, \vec{a} \vdash a'\end{array} \qquad g \circ f : (\vec{a} \multimap a' \to b)}{\Delta \vdash b}$$

$$[g : a \to b]\left(\frac{\begin{array}{c}\pi_0 \\ \vdots \\ \Gamma_0 \vdash \vec{a} \multimap a\end{array} \qquad \left(\begin{array}{c}\pi_i \\ \vdots \\ \Gamma_i \vdash a_i\end{array}\right)_{i=1}^{k} \qquad \eta : \Delta \to \bigotimes_{0}^{k} \Gamma_j}{\Delta \vdash a}\right) = \frac{\begin{array}{c}[1 \multimap g]\pi_0 \\ \vdots \\ \Gamma_0 \vdash \vec{a} \multimap b\end{array} \qquad \left(\begin{array}{c}\pi_i \\ \vdots \\ \Gamma_i \vdash a_i\end{array}\right)_{i=1}^{k} \qquad \eta}{\Delta \vdash b}$$

where $\vec{a} = \langle a_1, \dots, a_k\rangle$.

(b) Left action on derivations.

Figure 9.2.: Actions on derivations.

$$\frac{f \in A(o,o')}{f : o \to o'} \qquad\qquad \frac{\langle \sigma, \vec{f} \rangle : \vec{a}' \to \vec{a} \qquad f : a \to a'}{\langle \sigma, \vec{f} \rangle \multimap f : (\vec{a} \multimap a) \to (\vec{a}' \multimap a')}$$

$$\frac{\sigma \in \mathfrak{S}_k \quad f_1 : a_1 \to a'_{\sigma(1)} \quad \cdots \quad f_k : a_k \to a'_{\sigma(k)}}{\langle \sigma, f_1, \dots, f_k \rangle : \langle a_1, \dots, a_k \rangle \to \langle a'_1, \dots, a'_k \rangle}$$

Figure 9.3.: Multigraph of Intersection Types G_A.

Remark 9.2.3.

- *The rules of our system are induced by a fine-grained analysis of the λ-terms interpretations in* CatSym. *In contrast to what happens in standard intersection type systems, type derivations of variables in an environment are not unique in the bicategorical setting. Indeed, if we consider the set $\mathsf{T}_{\langle x \rangle}(x)(\langle a \rangle, a')$, for types $a, a' \in D$, it contains as many derivations for x as there are morphisms between a and a'. Hence, a type derivation of a variable corresponds to a particular witness of subtyping.*

- *Every derivation rule incorporates a subtyping inference. This differs from what happens in the systems presented in [Olimpieri, 2021], where the abstraction rule did not contain any additional subtyping. As the models under consideration are not just the free categories of intersection types, subtyping is needed also at the abstraction level now. We chose not to separate the subtyping rule from the other rules in order to keep our system syntax-directed and closer to the semantics.*

9.2.1. Completion of partial algebras

By mimicking the free completion of a partial pair which is often used to generate a graph model (see [Berline, 2000]), we show how to complete a partial $(!\, -^{\mathrm{op}} \times -)$-algebra by lifting it to an appropriate algebra (see Definition 7.3.1). We call the resulting algebra its *completion*.

Let us consider a partial $(!\, -^{\mathrm{op}} \times -)$-algebra $A \overset{F}{\hookleftarrow} H \overset{G}{\hookrightarrow} !A^{\mathrm{op}} \times A$. We denote by $G_A^{F,G}$ the multigraph whose nodes are elements of Ty_A and arrows are the ones from Figure 9.3, plus a family of invertible arrows:

$$\mathsf{e}_x : (i_A \circ F)(x) \cong (\iota_A \circ (!i_A^{\mathrm{op}} \times i_A) \circ G)(x),$$

for $x \in \mathrm{ob}(H)$, where we recall that $i_A : A \hookrightarrow D_A$ and $\iota_A : !D_A^{\mathrm{op}} \times D_A \hookrightarrow D_A$.

Definition 9.2.4 (Completion of Partial $(!\, -^{\mathrm{op}} \times -)$-Algebras)**.** *The* completion *of $A \overset{F}{\hookleftarrow} H \overset{G}{\hookrightarrow} !A^{\mathrm{op}} \times A$ is the category $D_A^{F,G}$ defined as the categorical quotient of the* *free category over $G_A^{F,G}$ by the following coherence on morphisms:*

$$\begin{tikzcd}
F(a) \arrow[r, "\mathsf{e}_a"] \arrow[dr, "F(f)"'] & G(a) \arrow[r, "G(f)"] & G(b) \\
& F(b) \arrow[ur, "\mathsf{e}_b"'] &
\end{tikzcd}$$

for any $f : a \to b$ in the category H.

We remark that we have a canonical functor $\iota^{F,G} : !(D^{F,G})^{\mathrm{op}} \times (D^{F,G}) \to D^{F,G}$ defined again by the map $\langle \vec{a}, a \rangle \mapsto \vec{a} \multimap a$.

Examples.

Definition 9.2.5. *We construct some partial $(!-^{\mathrm{op}} \times -)$-algebras together with their completions.*

- *We observe that, given a non-empty small category A, we have a canonical partial algebra over A defined by $A \supseteq \emptyset \subseteq !A^{\mathrm{op}} \times A$. Then the completion of that pair is exactly D_A.*

- *Let $A = \{*\}$, then we have the following two full embeddings:*

$$\begin{aligned}
&\mathtt{k}_A^+ : A^+ \hookrightarrow A, \qquad && \langle\langle * \rangle, * \rangle \mapsto *, \qquad && \text{with } A^+ = \{\langle\langle * \rangle, * \rangle\}, \\
&\mathtt{k}_A^* : A^* \hookrightarrow A, \qquad && \langle\langle \rangle, * \rangle \mapsto *, \qquad && \text{with } A^* = \{\langle\langle \rangle, * \rangle\}.
\end{aligned}$$

- *Given $n > 0$, we consider the set $[n] = \{1, \ldots, n\}$ equipped with its linear order structure. We see $[n]$ as a posetal category (the category associated with the partial order: objects are elements of the set and morphisms correspond to "less than or equal to" relation). Now, consider the full subcategory of $![n]^{\mathrm{op}} \times [n]$ induced by the family $[n]^+ = \langle\langle n - (i-1) \rangle, i \rangle_{i \in [n]}$. We define a functor $\mathtt{k}^{[n]} : [n]^+ \hookrightarrow [n]$ as follows:*

$$\mathtt{k}^{[n]}(\langle\langle n - (i-1) \rangle, i \rangle) = i.$$

  By construction, if there exists a morphism $\langle\langle n - (i-1) \rangle, i \rangle \to \langle\langle n - (j-1) \rangle, j \rangle$ then $i \leq_n j$. It is easy to verify that $\mathtt{k}^{[n]}$ is a full embedding.

- *We set*

$$\begin{aligned}
D^+ &= D^{\mathtt{k}^+, \mathtt{in}_{!A^{\mathrm{op}} \times A}}, \\
D^* &= D^{\mathtt{k}^*, \mathtt{in}_{!A^{\mathrm{op}} \times A}}, \\
D^{[n]} &= D^{\mathtt{k}^{[n]}, \mathtt{in}_{![n]^{\mathrm{op}} \times [n]}},
\end{aligned}$$

  and write $\iota^{\spadesuit}$, with $\spadesuit \in \{+, *\} \cup \mathbb{N}$, for the respective algebra maps. Notice that $D^{[1]} = D^+$.

$$\dfrac{\begin{array}{c}\pi_0\\ \vdots\\ \Gamma_0 \vdash \vec{b} \multimap a\end{array} \quad \left(\begin{array}{c}[f_i]\pi_{\sigma^{-1}(i)}\\ \vdots\\ \Gamma_{\sigma^{-1}(i)} \vdash b_i\end{array}\right)^k_{i=1} \quad (1 \otimes (\sigma^{-1})^\star) \circ \eta}{\Delta \vdash a} \sim \dfrac{\begin{array}{c}[\langle \sigma, \vec{f} \rangle \multimap 1]\pi_0\\ \vdots\\ \Gamma_0 \vdash \vec{a} \multimap a\end{array} \quad \left(\begin{array}{c}\pi_i\\ \vdots\\ \Gamma_i \vdash a_i\end{array}\right)^k_{i=1} \quad \eta}{\Delta \vdash a}$$

$$\dfrac{\begin{array}{c}\pi_0\{\theta_0\}\\ \vdots\\ \Gamma_0 \vdash \vec{a} \multimap a\end{array} \quad \left(\begin{array}{c}\pi_i\{\theta_i\}\\ \vdots\\ \Gamma_i \vdash a_i\end{array}\right)^k_{i=1} \quad \eta : \Delta \to \bigotimes_{j=0}^k \Gamma_j}{\Delta \vdash a} \sim \dfrac{\begin{array}{c}\pi_0\\ \vdots\\ \Gamma'_0 \vdash \vec{a} \multimap a\end{array} \quad \left(\begin{array}{c}\pi_i\\ \vdots\\ \Gamma'_i \vdash a_i\end{array}\right)^k_{i=1} \quad (\bigotimes_{j=0}^k \theta_j) \circ \eta}{\Delta \vdash a}$$

$$\dfrac{\begin{array}{c}[g]\pi\{1 \oplus \langle \sigma, \vec{g} \rangle\}\\ \vdots\\ \Delta, \vec{a}' \vdash a'\end{array} \qquad f : (\vec{a} \multimap a) \to b}{\Delta \vdash b} \sim \dfrac{\begin{array}{c}\pi\\ \vdots\\ \Delta, \vec{a} \vdash a\end{array} \quad f \circ (\langle \sigma, \vec{g} \rangle \multimap g) : (\vec{a}' \multimap a') \to b}{\Delta \vdash b}$$

where $\langle \sigma, f_1, \ldots, f_k \rangle : \vec{a} = \langle a_1, \ldots, a_k \rangle \to \vec{b} = \langle b_1, \ldots, b_k \rangle$, $\langle \sigma, \vec{g} \rangle : \vec{a}' \to \vec{a}$, $g : a \to a'$ and $\theta_i : \Gamma_i \to \Gamma'_i$. For $(\sigma^{-1})^\star$, see Definition 8.2.4.

Figure 9.4.: Congruence on derivations.

Theorem 9.2.6. *The functor $\iota^\spadesuit : !(D^\spadesuit)^{\mathrm{op}} \times (D^\spadesuit) \to D^\spadesuit$ for $\spadesuit \in \{+, *\} \cup \mathbb{N}$ is an equivalence of categories (Definition 7.1.6).*

Proof. Faithfulness is immediate by definition of $\iota^\spadesuit$. Moreover $\iota^\spadesuit$ is essentially surjective on objects by construction, since each atomic type of $D^\spadesuit$ is isomorphic to some arrow type. The proof of fullness consists of a fine-grained analysis of morphisms between arrow types. □

Remark 9.2.7. *The categories D^+ and D^* are categorifications of extensional graph models living in the relational semantics of λ-calculus [Breuvart et al., 2018]. Intuitively, they are given by the category D of types, where we add isomorphisms between atomic types in A and appropriate arrow types.*

For instance, in D^+ we obtain $\star \cong \iota^+(\langle \star \rangle, \star) = \langle \star \rangle \multimap \star$, while in D^* we have $\star \cong \iota^*(\langle \rangle, \star) = \langle \rangle \multimap \star$.

In this way, every λ-term which is typed with an atomic type can always be seen as a "function" and—as a consequence—one obtains extensionality. The category $D^{[2]}$ is a categorification of Coppo-Dezani-Zacchi's model, first appeared in [Coppo et al., 1987].

9.3. Intersection Type Distributor

In this section, given a pre-model $\langle D, \iota : !D^{\mathrm{op}} \times D \hookrightarrow D \rangle$, we will introduce the intersection type distributor $\mathsf{T}_{\vec{x}}(M) : \; !D^{\mathsf{len}(\vec{x})} \nrightarrow D$ whose elements are the derivations of intersection types associated to the λ-term M. We will observe that it may be considered as the interpretation in our bicategorical models.

First, we define a congruence on derivations $\sim\, \subseteq R_{\to} \times R_{\to}$ as the least congruence generated by the rules given in Figure 9.4. This congruence is the syntactic counterpart of the one generated by coends in the composition of distributors. It can be seen as the congruence equating derivations up to permutations that do not affect their computational information.

Notation 9.3.1. *Let $\pi \in R_{\to}$ be a derivation: the $\sim$-equivalence class of π is denoted by $\tilde{\pi} = \{\pi' \in R_{\to} \mid \pi \sim \pi'\} \in R_{\to}/\sim$.*

Example 9.3.2. *Let $k \in \mathbb{N}$, $\sigma \in \mathfrak{S}_k$ and*

$$\pi = \quad \cfrac{\begin{array}{c}\pi_0\\ \vdots\\ \Gamma_0 \vdash \langle a_1, \ldots, a_k\rangle \multimap a\end{array} \qquad \left(\begin{array}{c}\pi_i\\ \vdots\\ \Gamma_i \vdash a_i\end{array}\right)_{i=1}^{k} \qquad \eta}{\Delta \vdash a}$$

moreover, let $\eta' = (1 \otimes (\sigma)^{\star}) \circ \eta$ and

$$\pi' = \quad \cfrac{\begin{array}{c}\pi_0[\sigma \multimap a]\\ \vdots\\ \Gamma_0 \vdash \langle a_{\sigma(1)}, \ldots, a_{\sigma(k)}\rangle \multimap a\end{array} \qquad \left(\begin{array}{c}\pi_{\sigma(i)}\\ \vdots\\ \Gamma_{\sigma(i)} \vdash a_{\sigma(i)}\end{array}\right)_{i=1}^{k} \qquad \eta'}{\Delta \vdash a}$$

then $\pi \sim \pi'$ by the first rule of Figure 9.4. In fact, writing π_0' for $\pi_0[\sigma \multimap a]$, we obtain $\pi_0 = \pi_0'[\sigma^{-1} \multimap a]$. The two derivations have indeed the same computational meaning i.e. they only differ by performing the same permutation on inputs and on the list of types in the implication.

Remark 9.3.3. *Left and right actions on derivations are preserved under congruence:*

$$[f]\tilde{\pi} = \widetilde{[f]\pi} \quad \text{and} \quad \tilde{\pi}\{\eta\} = \widetilde{\pi\{\eta\}}.$$

The above construction naturally leads to the definition of the *intersection type distributors*, that will be the syntactic presentation of our bicategorical semantics.

Definition 9.3.4. *Let $M \in \Lambda^o(\vec{x})$. Define the $R_{\to}$-*intersection type distributor of *M, written $\mathsf{T}_{\vec{x}}(M) : !D^{\mathsf{len}(\vec{x})} \nrightarrow D$, as follows:*

1. *on objects:*
$$\mathsf{T}_{\vec{x}}(M)(\Delta, a) = \{\tilde{\pi} \in R_{\to}/_{\sim} \mid \pi \triangleright \Delta \vdash M : a\};$$

2. *on morphisms:*
$$\begin{array}{rrcl}\mathsf{T}_{\vec{x}}(M)(f, \eta) : & \mathsf{T}_{\vec{x}}(M)(\Delta, a) & \to & \mathsf{T}_{\vec{x}}(M)(\Delta', a')\\ & \tilde{\pi} & \mapsto & \widetilde{[f]\pi\{\eta\}}.\end{array}$$

The bicategorical semantics previously introduced can be presented syntactically, up to Seely equivalence (Proposition 8.2.8)—via intersection type distributors.

Recall that $\mu_1 : !D \times \cdots \times !D \to !(D \sqcup \cdots \sqcup D)$ is a component of Seely's equivalence (see Proposition 8.2.8), thus $\overline{\mu}_1 : !D \otimes \cdots \otimes !D \nrightarrow !(D \,\&\, \cdots \,\&\, D)$ by Definition 8.2.21. Also, since CatSym (Definition 8.2.6) is a full subcategory of Dist, the interpretation of a λ-term can be seen as a distributor:

$$[\![M]\!]_{\vec{x}} : !(\underbrace{D \,\&\, \cdots \,\&\, D}_{\mathsf{len}(\vec{x})\ \textit{times}}) \nrightarrow D.$$

Theorem 9.3.5. *For all $M \in \Lambda$, there is a natural isomorphism:*

$$\mathsf{itd}^{M}_{\vec{x}} : \mathsf{T}_{\vec{x}}(M) \cong [\![M]\!]_{\vec{x}} \circ_{\mathrm{Dist}} \overline{\mu}_1.$$

Proof. By structural induction on M, via lengthy but straightforward coend manipulations. □

By Theorem 8.1.5 we also get a natural isomorphism whenever $M \to_\beta N$:

$$[\![M \to_\beta N]\!]_{\vec{x}} \circ_{\mathrm{Dist}} \overline{\mu}_1 : [\![M]\!]_{\vec{x}} \circ_{\mathrm{Dist}} \overline{\mu}_1 \cong [\![N]\!]_{\vec{x}} \circ_{\mathrm{Dist}} \overline{\mu}_1.$$

This straightforwardly induces an isomorphism:

$$\mathsf{T}_{\vec{x}}(M \to_\beta N) : \mathsf{T}_{\vec{x}}(M) \cong \mathsf{T}_{\vec{x}}(N).$$

If D is an extensional model, then we have analogous isomorphisms in the case that $M \to_\eta N$.

9.3.1. Intersection Type Distributors of Böhm Trees

Now, note that the type assignment system generalises to $\lambda_\perp$-terms (see Definition 3.1.5) without adding any rule. It follows that $\perp$ is not typable.

We also extend the notion of intersection type distributor to $\lambda_\perp$-terms in the natural way, by setting $\mathsf{T}_{\vec{x}}(\perp) = \emptyset_{!D^n,D}$.

Lemma 9.3.6. *Let $M, N \in \Lambda_\perp$. If $M \sqsubseteq_\beta N$ then $[\![M]\!]_{\vec{x}} \subseteq [\![N]\!]_{\vec{x}}$ (and $\mathsf{T}_{\vec{x}}(M) \subseteq \mathsf{T}_{\vec{x}}(N)$).*

Proof. By an easy induction on the structure of M. □

Let us consider $\langle \mathcal{A}_\beta(M), \sqsubseteq_\beta \rangle$ as a preorder category (see Example 7.1.3). By applying the preceding lemma, for every $M \in \Lambda^o(x_1, \ldots, x_n)$ there exists an evident functor

$$\begin{array}{rcl} [\![-]\!]_{\{x_1,\ldots,x_n\}} : \mathcal{A}_\beta(M) & \to & \mathrm{Dist}(!(D^n), D), \\ P & \mapsto & [\![P]\!]_{\{x_1,\ldots,x_n\}}, \\ P \sqsubseteq_\beta Q & \mapsto & [\![P]\!]_{\{x_1,\ldots,x_n\}} \subseteq [\![Q]\!]_{\{x_1,\ldots,x_n\}}. \end{array}$$

Definition 9.3.7. *Let $M \in \Lambda^o(\vec{x})$.*

- *Since* Dist *has cocomplete hom-categories, we can define the* interpretation of the Böhm tree of M *(Definition 3.1.1) as the following filtered colimit:*

$$[\![\mathrm{BT}_\beta(M)]\!]_{\vec{x}} = \varinjlim_{P \in \mathcal{A}_\beta(M)} [\![P]\!]_{\vec{x}}.$$

- *We define the* $R_\to$-intersection type distributor of a Böhm tree*:*

$$\mathsf{T}_{\vec{x}}(\mathrm{BT}_\beta(M)) : !D^{\mathsf{len}(\vec{x})} \nrightarrow D$$

  in the following natural way:

  - *on objects:*

$$\mathsf{T}_{\vec{x}}(\mathrm{BT}_\beta(M))(\Delta, a) = \bigcup_{P \in \mathcal{A}_\beta(M)} \mathsf{T}_{\vec{x}}(P)(\Delta, a);$$

  - *on morphisms: for all $\eta : \Delta' \to \Delta, f : a \to a'$:*

$$\mathsf{T}_{\vec{x}}(\mathrm{BT}_\beta(M))(\eta, f)(\tilde{\pi}) = \widetilde{[f]\pi\{\eta\}}.$$

Theorem 9.3.8. *Let $M \in \Lambda$. we have a natural isomorphism:*

$$[\![\mathrm{BT}_\beta(M)]\!]_{\vec{x}} \circ_{\mathrm{Dist}} \overline{\mu}_1 \cong \mathsf{T}_{\vec{x}}(\mathrm{BT}_\beta(M)).$$

Proof. It follows from an inspection of the definitions and basic category theory, showing that $\mathsf{T}_{\vec{x}}(\mathrm{BT}_\beta(M))$ is a presentation of the filtered colimit of the intersection type distributors of the finite approximants of M:

$$\begin{array}{rcll}
[\![\mathrm{BT}_\beta(M)]\!]_{\vec{x}} \circ \overline{\mu}_1 & = & (colim_P [\![P]\!]_{\vec{x}}) \circ \overline{\mu}_1 & \textit{by definition,} \\
 & = & colim_P([\![P]\!]_{\vec{x}} \circ \overline{\mu}_1) & \textit{by exchange of colimits,} \\
 & = & colim_P(\mathsf{T}_{\vec{x}}(P)) & \textit{by Theorem 9.3.5,} \\
 & = & \mathsf{T}_{\vec{x}}(\mathrm{BT}_\beta(M)) & \textit{by definition.}
\end{array}$$

□

10. A Semantic Approximation Theorem

We have introduced in previous chapters a bicategorical model of λ-calculus (see Chapter 8). We have seen that this model can be represented as a type assignment system where the intersection is non-idempotent. We have also seen that the interpretation of a λ-term contains not only its typings but the whole type derivations (see Chapter 9). Now we want to study the importance of this new additional information. As previously, results presented in this chapter have been published in [Kerinec et al., 2023].

In Section 10.1 we will study the behaviour of intersection type distributors under reduction. In general, in a derivation π of $\Delta \vdash M : a$, only some of the subterms of M need to be typed. Therefore only some redexes of M are typed, and contracting them leads to a derivation π' having a strictly smaller size than π. Since the size of the derivation decreases strictly this process terminates in a finite number of steps, giving the normal form $\mathrm{NF}(\pi)$ of π.

We then define the *normal form* of the interpretation of a λ-term M. Theorem 10.2.8 is a *commutation theorem* stating that the normal form of the denotation of a λ-term coincides with the denotation of its Böhm tree. The theory of normalisation for our bicategorical semantics implicitly builds on techniques introduced in [Ehrhard and Regnier, 2008] in the setting of the Taylor expansion of λ-terms (see Section 4.2). In this context, Theorem 10.2.8 recalls a classical and crucial result: the normal form of the Taylor expansion of a λ-term coincides with the Taylor expansion of its Böhm tree. The underlying intuition is indeed that the intersection type derivations can be seen as (typed) *linear approximations* of λ-terms. From this perspective, our work can be also seen as a generalisation to the untyped case of the approach to Böhm trees semantics in [Tsukada et al., 2017].

Moreover in Section 10.2, we show that the normal form of a derivation π, such that $\pi \rhd \Delta \vdash M : a$, allows to reconstruct a finite approximant A_π of M such that $\mathrm{NF}(\pi)$ is a derivation of $\Delta \vdash A_\pi : a$. The technique for reconstructing an approximant from any derivation in the associated type system has been introduced in [Bucciarelli et al., 2014].

By combining these properties, we provide a combinatorial proof of the fact that every categorified graph model satisfies the Approximation Theorem 10.2.9 stating that the interpretation of a λ-term is isomorphic to the interpretation of its Böhm tree.
As in the relational case, the quantitative nature of our models allows to prove this property via a simple induction. In relational graph models the relevance of the system and

the lack of idempotency allow to extract from a typing an upper bound to the number of head reductions from a λ-term to its head normal form ([de Carvalho, 2018]). In [Breuvart et al., 2018], the authors exploited this quantitative information to give the first combinatorial proof of the Approximation Theorem for relational graph models. The proof is traditionally much harder, via Tait's computability predicates, or equivalently with Girard's reducibility candidates or with Krivine's saturated sets (respectively in [Tait, 1966],[Girard, 1989] and [Krivine, 1993]).

10.1. Typed Reductions

In the following we will work on a given categorified graph model, D.

In this section we observe that contracting redexes typed in a derivation leads to a derivation of strictly smaller size. A definition of normal forms of derivations follows naturally and then a definition of normal forms of λ-terms interpretations. The following technique originates in [Bucciarelli et al., 2014].

Definition 10.1.1.

- *Given a derivation $\pi \in R_{\rightarrow}$, we call* β-redex of π *a subderivation of π of shape:*

$$\dfrac{\dfrac{\dfrac{}{\Gamma_0, \langle c_1, \dots, c_k\rangle \vdash c} \quad g : (\vec{c} \multimap c) \to (\vec{b} \multimap a)}{\Gamma_0 \vdash \langle b_1, \dots, b_k\rangle \multimap a} \qquad \dfrac{}{(\Gamma_i \vdash b_i)_{i=1}^{k}} \quad \eta : \Delta \to \bigotimes_{i=0}^{k} \Gamma_i}{\Delta \vdash a}$$

- *Assume that $\pi \triangleright \Delta \vdash M : a$ (for $M \in \Lambda_{\perp}$). We say that a β-redex of M is* informative *in π if it is typed by a redex of π.*

- *A derivation π is* in β-normal form *if it has no β-redexes as subderivations.*

For simplicity reason we will say normal form instead of β-normal form.

Remark 10.1.2. *We can decompose $g : (\vec{c} \multimap c) \to (\vec{b} \multimap a)$ as $g = \langle \alpha, \vec{g_1}\rangle \multimap g_2$, with $\langle \alpha, \vec{g_1}\rangle : \vec{b} \to \vec{c}$ and $g_2 : c \to a$. And in such a case $[g]\pi = [g_2]\pi\{\langle \alpha, \vec{g_1}\rangle\}$.*

$$\dfrac{\dfrac{}{x : \langle\langle a, a\rangle \multimap a\rangle \vdash x : \langle a, a\rangle \multimap a} \quad \dfrac{\dfrac{}{y : \langle\langle\rangle \multimap a\rangle \vdash y : \langle\rangle \multimap a}}{y : \langle\langle\rangle \multimap a\rangle \vdash yz : a} \quad \dfrac{\dfrac{}{y : \langle\langle a\rangle \multimap a\rangle \vdash y : \langle a\rangle \multimap a} \quad \dfrac{}{z : \langle a\rangle \vdash z : a}}{y : \langle\langle a\rangle \multimap a\rangle, z : \langle a\rangle \vdash yz : a}}{x : \langle\langle a, a\rangle \multimap a\rangle, y : \langle\langle a\rangle \multimap a, \langle\rangle \multimap a\rangle, z : \langle a\rangle \vdash x(yz) : a}$$

Figure 10.1.: Example of a normal derivation in $R_{\rightarrow}$.

Example 10.1.3. *In Figure 10.1 we show a derivation in normal form.*

Given a $\lambda_\bot$-term M, a subterm occurrence N of M is uniquely identified by a single-hole context $C\llparenthesis - \rrparenthesis$ satisfying $M = C\llparenthesis N \rrparenthesis$ (see Definition 2.2.6(3)).

Definition 10.1.4. *Let $\pi \in R_\to$ be such that $\tilde{\pi} \in |\mathsf{T}_{\vec{x}}(M)|$ (the web of $\mathsf{T}_{\vec{x}}(M)$ see Definition 8.2.2(2)).*

- *Define a measure $\mathsf{tsize}()$, such that $\mathsf{tsize}(\pi) = n$ if and only if the derivation π contains exactly n applications of the rule (app).*
- *The set* $\mathrm{tocc}(\pi, M)$ *of* subterm occurrences of M that are typed in π *is defined by induction on π, splitting into cases depending on the last rule applied:*

  (ax) $\mathrm{tocc}(\pi, x) = \{\llparenthesis - \rrparenthesis\}$;

  (abs) $\mathrm{tocc}(\pi, \lambda x.M') = \{\llparenthesis - \rrparenthesis\} \cup \{\lambda x.C\llparenthesis - \rrparenthesis \mid C\llparenthesis - \rrparenthesis \in \mathrm{tocc}(\pi', M')\}$

  where $M = \lambda x.M'$ and the derivation π' is the premise of the rule;

  (app) $\mathrm{tocc}(\pi, M_0M_1) = \{\llparenthesis - \rrparenthesis\} \cup \{C\llparenthesis - \rrparenthesis M_1 \mid C\llparenthesis - \rrparenthesis \in \mathrm{tocc}(\pi_0, M_0)\}$
  $\cup\{M_0(C\llparenthesis - \rrparenthesis) \mid C\llparenthesis - \rrparenthesis \in \bigcup_{i=1}^{k} \mathrm{tocc}(\pi_i, M_1)\}$

  where $M = M_0M_1$, the derivation π_0 is the premise corresponding to M_0, and $\pi_1, \ldots, \pi_k$ are those corresponding to M_1 (if any).
- *We say that a subterm N of M is* typed *in π whenever $M = C\llparenthesis N \rrparenthesis$, for some $C\llparenthesis - \rrparenthesis \in \mathrm{tocc}(\pi, M)$.*

Example 10.1.5. *The redex $\mathbf{II} = (\lambda x.x)(\lambda x.x)$ is not typed in the following derivation*

$$\pi = \begin{array}{c} f : \langle\rangle \multimap a \to \langle\rangle \multimap a' \\ \hline x : \langle\langle\rangle \multimap a\rangle \vdash x : \langle\rangle \multimap a' \\ \hline x : \langle\langle\rangle \multimap a\rangle \vdash x(\mathbf{II}) : a' \end{array} \qquad \textit{thus, } \mathrm{tocc}(\pi, x(\mathbf{II})) = \{\llparenthesis - \rrparenthesis, \llparenthesis - \rrparenthesis(\mathbf{II})\}.$$

Remark 10.1.6. *The redex occurrences of a $\lambda_\bot$-term M that are typed in a derivation π correspond to the informative redexes of π (Definition 10.1.1). Therefore, π is in normal form exactly when none of the β-redexes of M is typed in π.*

Lemma 10.1.7 (Derivations of Approximants)**.** *Let $A \in \mathcal{A}_\beta$. If $\tilde{\pi} \in |\mathsf{T}_{\vec{x}}(A)|$ then π is a normal form.*

Proof. Immediate, since A does not contain any redex. □

Definition 10.1.8. *Given a derivation π such that $\pi \rhd \Delta_1, \vec{b}, \Delta_2 \vdash a$ with $\vec{b} = b_1, \ldots, b_k$ and derivations $\langle \pi_1, \ldots, \pi_k\rangle = \vec{\pi}$ where for each i ($1 \le i \le k$): $\pi_i \rhd \Gamma_i \vdash b_i$, the* substitution *of the types $\vec{b}$ by the derivations $\vec{\pi}$ in π is the derivation $\pi[\vec{\pi}/\vec{b}] \rhd (\Delta_1, \Delta_2) \otimes (\bigotimes_{i=0}^{k} \Gamma_i) \vdash a$, defined by induction on the last rule of π:*

- *if* $\pi = \dfrac{f : b \to a}{\langle b\rangle \vdash a}\, ax$ *then we should have $\vec{\pi} = \langle \pi_1 \rangle$ with $b_1 = b$ and $\pi[\vec{\pi}/\langle b\rangle] = [f]\pi_1$.*

- *if* $\pi = \dfrac{\dfrac{\pi_0}{\Delta_1, \vec{b}, \Delta_2, \vec{a} \vdash a} \qquad f : (\vec{a} \multimap a) \to b}{\Delta_1, \vec{b}, \Delta_2 \vdash b}\, lam$ *then by induction hypothesis we know how to define the derivation* $\dfrac{\pi_0[\vec{\pi}/\vec{b}]}{(\Delta_1, \Delta_2) \otimes \vec{\Gamma}, \vec{a} \vdash a}$ *so we can define*

$$\pi[\vec{\pi}/\vec{b}] = \dfrac{\dfrac{\pi_0[\vec{\pi}/\vec{b}]}{(\Delta_1, \Delta_2) \otimes \vec{\Gamma}, \vec{a} \vdash a} \qquad f : (\vec{a} \multimap a) \to b}{(\Delta_1, \Delta_2) \otimes \vec{\Gamma} \vdash b}\, lam$$

- *if* $\pi =$

$$\dfrac{\dfrac{\pi'_0}{\Sigma'_0, \vec{b'_0}, \Delta'_0 \vdash \langle a_1, \ldots, a_l \rangle \multimap a} \quad \left(\dfrac{\pi'_i}{\Sigma'_i, \vec{b'_i}, \Delta'_i \vdash a_i}\right)_{i=1}^{l} \quad \eta, f, \theta : \Sigma, \vec{b}, \Delta \to \Sigma', \vec{b'}, \Delta'}{\Sigma, \vec{b}, \Delta \vdash a}\, app$$

With $f : \vec{b} \to \bigotimes_{i=0}^{l} \vec{b'_i} = \vec{b'}$.

By induction hypothesis we have $\pi'_i[([f]\vec{\pi})_i/\vec{b'_i}]$ *for* $i = 0, \ldots, l$.

We can define
$\pi[\vec{\pi}/\vec{b}] =$

$$\dfrac{\dfrac{\pi'_0[([f]\vec{\pi})_0/\vec{b'_0}]}{(\Sigma'_0 \otimes \vec{\Gamma}^0_0) \otimes (\Delta'_0 \otimes \vec{\Gamma}^1_0) \vdash \langle a_1, \ldots, a_l \rangle \multimap a} \quad \left(\dfrac{(\pi'_i[([f]\vec{\pi})_i/\vec{b'_i}])}{(\Sigma'_i \otimes \vec{\Gamma}^0_i) \otimes (\Delta'_i \otimes \vec{\Gamma}^1_i) \vdash a_i}\right)_{i=1}^{l} \quad \tau((\eta, \theta) \otimes 1)}{(\Sigma, \Delta) \otimes \vec{\Gamma} \vdash a}\, app$$

τ *being the permutation that (on each component) does*

$$(\vec{a}_0 \otimes \vec{a}_1) \otimes (\vec{b}_0 \otimes \vec{b}_1) \mapsto (\vec{a}_0 \otimes \vec{b}_0) \otimes (\vec{a}_1 \otimes \vec{b}_1)$$

so

$$\tau((\eta, \theta) \otimes 1) : (\Sigma, \Delta) \otimes \vec{\Gamma} \to (\Sigma' \otimes \vec{\Gamma}^0) \otimes (\Delta' \otimes \vec{\Gamma}^1).$$

Definition 10.1.9. *Let a derivation* π *contains a* β*-redex*

$$\pi' = \dfrac{\dfrac{\dfrac{\pi_0}{\Gamma_0, \langle c_1, \ldots, c_k \rangle \vdash c} \quad g : (\vec{c} \multimap c) \to (\vec{b} \multimap a)}{\Gamma_0 \vdash \langle b_1, \ldots, b_k \rangle \multimap a} \quad \left(\dfrac{\pi_i}{\Gamma_i \vdash b_i}\right)_{i=1}^{k} \quad \eta : \Delta \to \bigotimes_{i=0}^{k} \Gamma_i}{\Delta \vdash a}$$

The contraction *of this* β*-redex is the derivation* π *where we replaced the subderivation* π' *by* $(([g]\pi_0)[\langle \pi_1, \ldots, \pi_k \rangle / \langle b_1, \ldots, b_k \rangle])\{\eta\}$. *We name this new derivation a* β-reduct *of* π.

Example 10.1.10. *Given $M = (\lambda x.x)y$ and $N = y$, we have $M \to_\beta N$.*

Consider

$$\pi = \dfrac{\dfrac{\dfrac{1}{x : \langle a\rangle \vdash x : a} \quad g : (\langle a\rangle \multimap a) \to (\langle c\rangle \multimap b)}{\vdash \lambda x.x : \langle c\rangle \multimap b} \quad \dfrac{1}{y : \langle c\rangle \vdash y : c} \quad 1}{y : \langle c\rangle \vdash (\lambda x.x)y : b}$$

we want to contract the last rule, corresponding to $(\lambda x.x)y \to_\beta y$. So we took the subderivation $\pi_0 = \dfrac{1}{x : \langle a\rangle \vdash x : a}$ wich is a rule ax.

And we have $\mathsf{len}(\vec{\pi}) = 1$, $\pi_1 = \dfrac{1}{y : \langle c\rangle \vdash y : c}$.

So $([g]\pi_0)[\vec{\pi}/\langle b\rangle]\{1\} = \dfrac{c \to b}{x : \langle c\rangle \vdash x : b}[\vec{\pi}/\langle b\rangle] = [c \to b]\pi_1 = \dfrac{c \to b}{y : \langle c\rangle \vdash y : b}$.

Notation 10.1.11. *We use the notation $\pi[\pi_2/\pi_1]$ for the derivation π where we replaced the subderivation π_1 by the subderivation π_2 (admitting that π_1 and π_2 have the same typing).*

Theorem 10.1.12. *The β-reduction on derivations is strongly normalising (see Definition 2.1.3).*

Proof. Given a derivation π containing a β-redex π' (for the same notation than in Definition 10.1.9). By contracting π', we have $\pi \to_\beta \pi[([g]\pi_0)[\langle \pi_1, \ldots, \pi_k\rangle/\langle b_1, \ldots, b_k\rangle]\{\eta\}/\pi']$. Since one rule *app* is deleted and no duplication is done during the substitution, we have that $\mathsf{tsize}(([g]\pi_0)[\langle \pi_1, \ldots, \pi_k\rangle/\langle b_1, \ldots, b_k\rangle])\{\eta\} < \mathsf{tsize}(\pi')$. We then deduce that $\mathsf{tsize}(\pi[([g]\pi_0)[\langle \pi_1, \ldots, \pi_k\rangle/\langle b_1, \ldots, b_k\rangle]\{\eta\}/\pi']) < \mathsf{tsize}(\pi)$ and that the size of the derivations decreases strictly during their β-reduction. □

Theorem 10.1.13. *The β-reduction on derivations is locally confluent (see Definition 2.1.3).*

Proof. The proof corresponds to the observation of a derivation π containing two different β-redexes:

$$\pi_1 = \dfrac{\dfrac{\dfrac{\pi_0^1}{\Gamma_0^1, \vec{c_1} \vdash c_1} \quad g_1 : (\vec{c_1} \multimap c_1) \to (\vec{b_1} \multimap a_1)}{\Gamma_0^1 \vdash \langle b_1^1, \ldots, b_{k_1}^1\rangle \multimap a_1} \quad \left(\dfrac{\pi_i^1}{\Gamma_i^1 \vdash b_i^1}\right)_{i=1}^{k_1} \quad \eta_1 : \Delta_1 \to \bigotimes_{i=0}^{k_1} \Gamma_i^1}{\Delta_1 \vdash a_1}\, app.$$

and

$$\pi_2 = \dfrac{\dfrac{\dfrac{\pi_0^2}{\Gamma_0^2, \vec{c_2} \vdash c_2} \quad g_2 : (\vec{c_2} \multimap c_2) \to (\vec{b_2} \multimap a_2)}{\Gamma_0^2 \vdash \langle b_1^2, \ldots, b_{k_2}^2\rangle \multimap a_2} \quad \left(\dfrac{\pi_i^2}{\Gamma_i^2 \vdash b_i^2}\right)_{i=1}^{k_2} \quad \eta_2 : \Delta_2 \to \bigotimes_{i=0}^{k_2} \Gamma_i^2}{\Delta_2 \vdash a_2}\, app.$$

We will observe that the order in which we contract those β-redexes does not matter.

To begin with, if π_1 and π_2 appear in two different premises of another *app* rule, the order in wich the substitutions are made obviously does not matter.

Otherwise wlog we can assume π_1 appears in one of the premises of π_2. It is sufficient to observe the substitutions made in the two orders in π_2 since the order doesn't matter in other subderivations of π. There are two options for the position of π_1 in the derivation π_2:

- π_1 appears in π_l^2 for a $l \in [1, k_2]$:
  - Contacting π_1 leads to π_1 being replaced by $([g_1]\pi_0^1)[\vec{\pi_1}/\vec{b_1}]\{\eta_1\} = \pi_1'$ and π_l^2 by $\pi_l^2[\pi_1'/\pi_1]$, π_0^2 is not altered as well as the others π_i^2's. Then if we contract π_2 we replaced it by $([g_2]\pi_0^2)[\langle \pi_1^2, \ldots, \pi_l^2[\pi_1'/\pi_1], \ldots, \pi_{k_2}^2\rangle/\vec{b_2}]\{\eta_2\}$.
  - On the other way: if we contract π_2 we replaced it by $([g_2]\pi_0^2)[\vec{\pi_2}/\vec{b_2}]\{\eta_2\}$. During a substitution there is no duplication or deletion of the π_i^2's, π_1 still apears in the subderivation π_l^2, contracting it still leads to $\pi_l^2[\pi_1'/\pi_1]$ and we replaced π_2 by $([g_2]\pi_0^2)[\langle \pi_1^2, \ldots, \pi_l^2[\pi_1'/\pi_1], \ldots, \pi_{k_2}^2\rangle/\vec{b_2}]\{\eta_2\}$.
- π_1 appears in π_0^2, we proceed by induction on π_0^1:
  - $\pi_0^1 = \dfrac{l : b \to c_1}{\langle b \rangle \vdash c_1}\, ax$
    - ∗ if $b = c_h \in \vec{c_1}$ then
      - · if we contract π_1 before then we replaced π_1 by $[g_1 \circ l]\pi_{\alpha_1(h)}^1\{\eta_1\}$ (given $g_1 = \langle \alpha_1, \vec{g_1}\rangle \multimap g_1$), so π_0^2 become $\pi_0^2[[g_1 \circ l]\pi_{\alpha_1(h)}^1\{\eta_1\}/\pi_1]$, and then by contracting π_2, we get $([g_2]\pi_0^2[[g_1 \circ l]\pi_{\alpha_1(h)}^1\{\eta_1\}/\pi_1])[\vec{\pi_2}/\vec{b_2}]\{\eta_2\}$.
      - · otherwise when we contract π_2, π_0^1 is not altered since it does not used any c_2, but $\pi_{\alpha_1(h)}^1$ is replaced by $([g_2]\pi_{\alpha_1(h)}^1)[\vec{\pi_2}/\vec{b_2}]$ and π_2 become $([g_2]\pi_0^2)[\vec{\pi_2}/\vec{b_2}]\{\eta_2\}$. Then contracting π_1 leads π_1 to be replaced by $([g_1 \circ l](([g_2]\pi_{\alpha_1(h)}^1)[\vec{\pi_2}/\vec{b_2}]))\{\eta_1\}$ and we get $([g_2]\pi_2^0)[\vec{\pi_2}/\vec{b_2}]\{\eta_2\}[([g_1 \circ l](([g_2]\pi_{\alpha_1(h)}^1)[\vec{\pi_2}/\vec{b_2}]))\{\eta_1\}/\pi_1] = ([g_2]\pi_0^2[[g_1 \circ l]\pi_{\alpha_1(h)}^1\{\eta_1\}/\pi_1])[\vec{\pi_2}/\vec{b_2}]\{\eta_2\}$.
    - ∗ if $b = c_h^2 \in \vec{c_2}$ then
      - · if we contract π_1 before then it become $\pi_0^1\{\eta_1\}$ and π_h^2 is not altered, when we contract π_2 we replace the initial π_0^1 by $[l \circ g_h^2]\pi_{\alpha_2(h)}^2\{\eta_1\}$ (given $g_2 = \langle \alpha_2, \vec{g_2}\rangle \multimap g_2$).
      - · otherwise we obtain a similar result since the contaction of π_1 does not afect π_h^2.
    - ∗ if $b \notin \vec{c_1} \cup \vec{c_2}$ the order of the contractions obviously does not matter.
- the two other cases follow easily by induction hypothesis. □

Theorem 10.1.14. *The β-reduction on derivations is confluent (see Definition 2.1.3).*

Proof. Using Theorems 10.1.12 and 10.1.13 and by Lemma 2.1.4. □

We will now focus on congruences defined in Figure 9.4.

Lemma 10.1.15. *Given $\pi_1, \pi_2 \rhd \Delta \vdash (\lambda x.M)N : a$, such that $\pi_1 \to_\beta \pi'_1 \rhd \Delta \vdash M[N/x] : a$, $\pi_2 \to_\beta \pi'_2 \rhd \Delta \vdash M[N/x] : a$ if $\pi'_1 \sim \pi'_2$ then $\pi_1 \sim \pi_2$.*

Proof. We proceed by induction on M:

- if $M = x$ then for $i = 1, 2$: $\quad \pi_i =$

$$\dfrac{\dfrac{\dfrac{l_i : c_1^i \to c_i}{\vec{c_i} = \langle c_1^i \rangle \vdash c_i} \qquad \langle 1, \vec{f_i} \rangle \multimap f_i : (\vec{c_i} \multimap c_i) \to (\vec{a_i} \multimap a)}{\vdash \langle a_1^i \rangle \multimap a} \qquad \dfrac{\pi_1^i}{\Gamma_1^i \vdash a_1^i} \qquad \eta_i : \Delta \to \bigotimes_{j=0}^{k_i} \Gamma_j^i}{\Delta \vdash a}$$

$M[N/x] = N$ and $\pi'_i = ([\langle 1, \vec{f_i} \rangle \multimap f_i]\pi_0^i)[\vec{\pi_i}/\vec{a_i}]\{\eta_i\} = ([((\langle 1, \vec{f_i} \rangle \multimap f_i)l_i]\pi_1^i)\{\eta_i\}$. So $([((\langle 1, \vec{f_1} \rangle \multimap f_1)l_1]\pi_1^1)\{\eta_1\} \sim ([((\langle 1, \vec{f_2} \rangle \multimap f_2)l_2]\pi_1^2)\{\eta_2\}$.

$\pi_1 \sim$

$$\dfrac{\dfrac{\dfrac{[f_1]l_1\{\langle 1, \vec{f_1} \rangle\}}{\langle a_1^1 \rangle \vdash a} \qquad 1}{\vdash \langle a_1^1 \rangle \multimap a} \qquad \dfrac{\pi_1^1}{\Gamma_1^1 \vdash a_1^1} \qquad \eta_1 : \Delta \to \Gamma_1^1}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{1}{\langle a \rangle \vdash a} \qquad ((\langle 1, \vec{f_1} \rangle \multimap f_1)l_1 \multimap 1}{\vdash \langle a_1^1 \rangle \multimap a} \qquad \dfrac{\pi_1^1}{\Gamma_1^1 \vdash a_1^1} \qquad \eta_1 : \Delta \to \Gamma_1^1}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{1}{\langle a \rangle \vdash a} \qquad 1}{\vdash \langle a \rangle \multimap a} \qquad \dfrac{[((\langle 1, \vec{f_1} \rangle \multimap f_1)l_1]\pi_1^1}{\Gamma_1^1 \vdash a} \qquad \eta_1 : \Delta \to \Gamma_1^1}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{1}{\langle a \rangle \vdash a} \qquad 1}{\vdash \langle a \rangle \multimap a} \qquad \dfrac{[((\langle 1, \vec{f_1} \rangle \multimap f_1)l_1]\pi_1^1\{\eta_1\}}{\Delta \vdash a} \qquad 1}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{1}{\langle a \rangle \vdash a} \qquad 1}{\vdash \langle a \rangle \multimap a} \qquad \dfrac{[((\langle 1, \vec{f_2} \rangle \multimap f_2)l_2]\pi_1^2\{\eta_2\}}{\Delta \vdash a} \qquad 1}{\Delta \vdash a}$$

$\sim \cdots \sim \pi_2$

- if $M = y$ (with $y \neq x$) then for $i = 1, 2$

$$\pi_i = \dfrac{\dfrac{\dfrac{l_i : c_1^i \to c_i}{\langle c_1^i \rangle \vdash c_i} \quad \langle\rangle \multimap f_i : (\langle\rangle \multimap c_i) \to (\langle\rangle \multimap a)}{\langle c_1^i \rangle \vdash \langle\rangle \multimap a} \quad \eta_i : \Delta \to \langle c_1^i \rangle}{\Delta \vdash a}$$

$M[N/x] = M$

$$\pi'_i = \dfrac{(f_i \circ l_i)\eta_i : \Delta \to a}{\Delta \vdash a}$$

Since $\pi'_1 \sim \pi'_2$, we get $(f_1 \circ l_1)\eta_1 = (f_2 \circ l_2)\eta_2$ and
$\pi_1 \sim$

$$\dfrac{\dfrac{\dfrac{l_1 \circ f_1 : c_1^1 \to a}{\langle c_1^1 \rangle \vdash a} \quad \langle\rangle \multimap 1}{\langle c_1^1 \rangle \vdash \langle\rangle \multimap a} \quad \eta_1 : \Delta \to \langle c_1^1 \rangle}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{(l_1 \circ f_1)\eta_1 : \Delta \to a}{\Delta \vdash a} \quad \langle\rangle \multimap 1}{\Delta \vdash \langle\rangle \multimap a} \quad 1}{\Delta \vdash a}$$

$=$

$$\dfrac{\dfrac{\dfrac{(l_2 \circ f_2)\eta_2 : \Delta \to a}{\Delta \vdash a} \quad \langle\rangle \multimap 1}{\Delta \vdash \langle\rangle \multimap a} \quad 1}{\Delta \vdash a}$$

$\sim \cdots \sim \pi_2$

- if $M = \lambda y.M'$ then for $i = 1, 2$: $\pi_i =$

$$\dfrac{\dfrac{\dfrac{\dfrac{\rho_i}{\Gamma_0^i, \vec{c_i}, \vec{d_i} \vdash d_i} \quad \langle \beta_i, \vec{g_i} \rangle \multimap g_i : (\vec{d_i} \multimap d_i) \to c_i}{\Gamma_0^i, \vec{c_i} \vdash c_i} \quad \langle \alpha_i, \vec{f_i} \rangle \multimap f_i}{\Gamma_0^i \vdash \langle a_1^i, \ldots, a_{n_i}^i \rangle \multimap a} \quad \left(\dfrac{\pi_j^i}{\Gamma_j^i \vdash a_j^i} \right)_{j=1}^{n_i} \quad \eta_i}{\Delta \vdash a}$$

$$\pi'_i = \dfrac{\dfrac{\rho_i\{1 \oplus \langle \alpha_i, \vec{f_i} \rangle \oplus 1\}[\vec{\pi_i}/\vec{a_i}]\{\eta_i \oplus 1\}}{\Delta, \vec{d_i} \vdash d_i} \quad \langle \beta_i, \vec{g_i} \rangle \multimap f_i \circ g_i : (\vec{d_i} \multimap d_i) \to a}{\Delta \vdash a}$$

and $\pi'_1 \sim \pi'_2$ so without loss of generality there exists k and $\langle \delta, \vec{k} \rangle$ such that $(\langle \beta_1, \vec{g_1} \rangle \multimap f_1 \circ g_1) \circ (\langle \delta, \vec{k} \rangle \multimap k) = \langle \beta_2, \vec{g_2} \rangle \multimap f_2 \circ g_2$ and
$\rho_1\{1 \oplus \langle \alpha_1, \vec{f_1} \rangle \oplus 1\}[\vec{\pi_1}/\vec{a_1}]\{\eta_1 \oplus 1\} \sim [k](\rho_2\{1 \oplus \langle \alpha_2, \vec{f_2} \rangle \oplus 1\}[\vec{\pi_2}/\vec{a_2}])\{\eta_2 \oplus \langle \delta, \vec{k} \rangle\}$.

We consider $\rho_1' = \rho_1\{D_1\}$ with $D_1 : \Gamma_0^1, \vec{d_1}, \vec{c_1} \to \Gamma_0^1, \vec{c_1}, \vec{d_1}$ and ρ_2' defined in the same way. By definition we have $\rho_1\{1 \oplus \langle \alpha_1, \vec{f_1}\rangle \oplus 1\}[\vec{\pi_1}/\vec{a_1}] = \rho_1'\{1 \oplus 1 \oplus \langle \alpha_1, \vec{f_1}\rangle\}[\vec{\pi_1}/\vec{a_1}]$ (resp. ρ_2').

By induction hypothesis we get:

$$R_1 = \dfrac{\dfrac{\dfrac{\rho_1'}{\Gamma_0^1, \vec{d_1}, \vec{c_1} \vdash d_1} \quad \langle \alpha_1, \vec{f_1}\rangle \multimap 1 : \vec{c_1} \multimap d_1 \to \vec{a_1} \multimap d_1}{\Gamma_0^1, \vec{d_1} \vdash \vec{a_1} \multimap d_1} \quad \left(\dfrac{\pi_j^1}{\Gamma_j^1 \vdash a_j^1}\right)_{j=1}^{n_1} \quad \eta_1 \oplus 1}{\Delta, \vec{d_1} \vdash d_1}$$

and

$R_2 =$

$$\dfrac{\dfrac{\dfrac{[k]\rho_2'\{1 \oplus \langle \delta, \vec{k}\rangle \oplus 1\}}{\Gamma_0^2, \vec{d_1}, \vec{c_2} \vdash d_1} \quad \langle \alpha_2, \vec{f_2}\rangle \multimap 1 : \vec{c_2} \multimap d_1 \to \vec{a_2} \multimap d_1}{\Gamma_0^2, \vec{d_1} \vdash \vec{a_2} \multimap d_1} \quad \left(\dfrac{\pi_j^2}{\Gamma_j^2 \vdash a_j^2}\right)_{j=1}^{n_2} \quad \eta_2 \oplus 1}{\Delta, \vec{d_1} \vdash d_1}$$

with $R_1 \sim R_2$. So $n_1 = n_2$ we name it n.

By an inspection of the congruence rules, we can consider the existence of $\langle \sigma, \vec{h}\rangle$, θ and $\langle \gamma, \vec{l}\rangle$ such that:

- $\rho_1' \sim [k]\rho_2'\{(1 \oplus \langle \delta, \vec{k}\rangle \oplus 1) \circ (\theta_0 \oplus 1 \oplus 1) \circ (1 \oplus 1 \oplus \langle \gamma, \vec{l}\rangle)\}$;
- $\forall j = 1, \ldots, n : \pi_j^1 \sim [h_j]\pi^2_{\sigma^{-1}(j)}\{\theta_j\}$;
- $(1 \otimes (\sigma^{-1})^\star) \circ \eta_2 = \bigotimes_{j=0}^n \theta_j \circ \eta_1$;
- $\langle \alpha_2, \vec{f_2}\rangle = \langle \sigma, \vec{h}\rangle \circ \langle \alpha_1, \vec{f_1}\rangle \circ \langle \gamma, \vec{l}\rangle$.

(or such that

- $\rho_1'\{(\theta_0 \oplus 1 \oplus 1) \circ (1 \oplus 1 \oplus \langle \gamma, \vec{l}\rangle)\} \sim [k]\rho_2'\{1 \oplus \langle \delta, \vec{k}\rangle \oplus 1\}$;
- $\forall j = 1, \ldots, n : \pi_j^2 \sim [h_j]\pi^1_{\sigma(j)}\{\theta_j\}$;
- $(1 \otimes (\sigma^{-1})^\star) \circ \eta_1 = \bigotimes_{j=0}^n \theta_j \circ \eta_2$;
- $\langle \alpha_1, \vec{f_1}\rangle = \langle \sigma, \vec{h}\rangle \circ \langle \alpha_2, \vec{f_2}\rangle \circ \langle \gamma, \vec{l}\rangle$.

or

- $\rho_1'\{\theta_0 \oplus 1 \oplus 1\} \sim [k]\rho_2'\{(1 \oplus \langle \delta, \vec{k}\rangle \oplus 1) \circ (1 \oplus 1 \oplus \langle \gamma, \vec{l}\rangle)\}$;
- $\forall j = 1, \ldots, n : \pi_j^1\{\theta_j\} \sim [h_j]\pi^2_{\sigma(j)}$;
- $\bigotimes_{j=0}^n \theta_j \circ (1 \otimes (\sigma^{-1})^\star) \circ \eta_2 = \eta_1$;

– $\langle\alpha_2,\vec{f_2}\rangle = \langle\sigma,\vec{h}\rangle \circ \langle\alpha_1,\vec{f_1}\rangle \circ \langle\gamma,\vec{l}\rangle$.

... or equivalent configurations using congruence rules symmetrically.)

We deduce that $\rho_1 \sim [k]\rho_2\{(1\oplus 1\oplus\langle\delta,\vec{k}\rangle)\circ(\theta_0\oplus 1\oplus 1)\circ(1\oplus\langle\gamma,\vec{l}\rangle\oplus 1)\}$

We can now use this knowlegde into $\pi_1 \sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[k]\rho_2\{\theta_0 \oplus \langle\gamma,\vec{l}\rangle \oplus \langle\delta,\vec{k}\rangle\}}{\Gamma^1_0,\vec{c_1},\vec{d_1}\vdash d_1} \qquad \langle\beta_1,\vec{g_1}\rangle \multimap g_1}{\Gamma^1_0,\vec{c_1}\vdash c_1} \qquad \langle\alpha_1,\vec{f_1}\rangle \multimap f_1}{\Gamma^1_0\vdash\vec{a_1}\multimap a} \quad \left(\dfrac{[h_j]\pi^2_{\sigma^{-1}(j)}\{\theta_j\}}{\Gamma^1_{\sigma^{-1}(j)}\vdash a^1_j}\right)^n_{j=1} \quad \eta_1}{\Delta\vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[k]\rho_2\{\theta_0 \oplus \langle\gamma,\vec{l}\rangle \oplus \langle\delta,\vec{k}\rangle\}}{\Gamma^1_0,\vec{c_1},\vec{d_1}\vdash d_1} \qquad \langle\beta_1,\vec{g_1}\rangle \multimap f_1\circ g_1}{\Gamma^1_0,\vec{c_1}\vdash a} \qquad \langle\alpha_1,\vec{f_1}\rangle \multimap 1}{\Gamma^1_0\vdash\vec{a_1}\multimap a} \quad \left(\dfrac{[h_j]\pi^2_{\sigma^{-1}(j)}\{\theta_j\}}{\Gamma^1_{\sigma^{-1}(j)}\vdash a^1_j}\right)^n_{j=1} \quad \eta_1}{\Delta\vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[k]\rho_2\{1 \oplus \langle\gamma,\vec{l}\rangle \oplus \langle\delta,\vec{k}\rangle\}}{\Gamma^2_0,\vec{c_1},\vec{d_1}\vdash d_1} \qquad \langle\beta_1,\vec{g_1}\rangle \multimap f_1\circ g_1}{\Gamma^2_0,\vec{c_1}\vdash a} \qquad \langle\alpha_1,\vec{f_1}\rangle \multimap 1}{\Gamma^2_0\vdash\vec{a_1}\multimap a} \quad \left(\dfrac{[h_j]\pi^2_{\sigma^{-1}(j)}}{\Gamma^2_{\sigma^{-1}(j)}\vdash a^1_j}\right)^n_{j=1} \quad (1\otimes(\sigma^{-1})^\star)\circ\eta_2}{\Delta\vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[k]\rho_2\{1 \oplus \langle\gamma,\vec{l}\rangle \oplus \langle\delta,\vec{k}\rangle\}}{\Gamma^2_0,\vec{c_1},\vec{d_1}\vdash d_1} \qquad \langle\beta_1,\vec{g_1}\rangle \multimap f_1\circ g_1}{\Gamma^2_0,\vec{c_1}\vdash a} \qquad \langle\sigma,\vec{h}\rangle\circ\langle\alpha_1,\vec{f_1}\rangle \multimap 1}{\Gamma^2_0\vdash\vec{a_2}\multimap a} \quad \left(\dfrac{\pi^2_j}{\Gamma^2_j\vdash a^2_j}\right)^n_{j=1} \quad \eta_2}{\Delta\vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[k]\rho_2\{1 \oplus 1 \oplus \langle\delta,\vec{k}\rangle\}}{\Gamma^2_0,\vec{c_2},\vec{d_1}\vdash d_1} \qquad \langle\beta_1,\vec{g_1}\rangle \multimap f_1\circ g_1}{\Gamma^2_0,\vec{c_2}\vdash a} \qquad \langle\alpha_2,\vec{f_2}\rangle \multimap 1}{\Gamma^2_0\vdash\vec{a_2}\multimap a} \quad \left(\dfrac{\pi^2_j}{\Gamma^2_j\vdash a^2_j}\right)^n_{j=1} \quad \eta_2}{\Delta\vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{\rho_2}{\Gamma_0^2, \vec{c_2}, \vec{d_2} \vdash d_2} \quad \langle \beta_2, \vec{g_2} \rangle \multimap f_2 \circ g_2}{\Gamma_0^2, \vec{c_2} \vdash a} \qquad \langle \alpha_2, \vec{f_2} \rangle \multimap 1}{\Gamma_0^2 \vdash \vec{a_2} \multimap a} \qquad \left(\dfrac{\pi_j^2}{\Gamma_j^2 \vdash a_j^2} \right)_{j=1}^{n} \qquad \eta_2}{\Delta \vdash a}$$

$\sim \pi_2$

(for other configurations depending of $\langle \sigma, \vec{h} \rangle$ and $\langle \gamma, \vec{l} \rangle$ we proceed in the same way).

- if $M = M_1 M_2$ then for $i = 1, 2$: $\pi_i =$

$$\dfrac{\dfrac{\dfrac{\dfrac{\rho_0^i}{T_0^i, \vec{b_0^i} \vdash \vec{d_i} \multimap c_i} \qquad \left(\dfrac{\rho_j^i}{T_j^i, \vec{b_j^i} \vdash d_j^i} \right)_{j=1}^{o^i} \qquad \Theta_i, h_i}{\Gamma_0^i, \vec{c_i} = \langle c_1^i, \dots, c_{n_i}^i \rangle \vdash c_i} \qquad \langle \alpha_i, \vec{f_i} \rangle \multimap f_i}{\Gamma_0^i \vdash \vec{a_i} \multimap a} \qquad \left(\dfrac{\pi_j^i}{\Gamma_j^i \vdash a_j^i} \right)_{j=1}^{n_i} \qquad \eta_i}{\Delta \vdash a}$$

with $\langle \sigma_i, \vec{h_i} \rangle = h_i$ and $\Theta_i, \langle \sigma_i, \vec{h_i} \rangle : \Gamma_0^i, \vec{c_i} \to \bigotimes_{j=0}^{o^i} (T_j^i, \vec{b_j^i})$ and $\forall j = 0, \dots, o^i : \vec{b_j^i} = \langle b_{j,1}^i, \dots, b_{j,l_j^i}^i \rangle$. We set $\vec{\Gamma}_j^i$ for $([h_i \circ \langle \alpha_i, \vec{f_i} \rangle] \vec{\Gamma}_i)_j$.

We have $M[N/x] = M_1[N/x] M_2[N/x]$ and $\pi_i' =$

$$\dfrac{\dfrac{[1 \multimap f_i] \rho_0^i [([h_i \circ \langle \alpha_i, \vec{f_i} \rangle] \vec{\pi_i})_0 / \vec{b_0^i}]}{T_0^i \otimes \vec{\Gamma}_0^i \vdash \vec{d_i} \multimap a} \qquad \left(\dfrac{\rho_j^i [([h_i \circ \langle \alpha_i, \vec{f_i} \rangle] \vec{\pi_i})_j / \vec{b_j^i}]}{T_j^i \otimes \vec{\Gamma}_j^i \vdash d_j^i} \right)_{j=1}^{o^i} \qquad \tau_i(\Theta_i \otimes (\sigma_i^{-1} \circ \alpha_i^{-1})^\star) \circ \eta_i}{\Delta \vdash a}$$

with $\tau_i(\Theta_i \otimes (\sigma_i^{-1} \circ \alpha_i^{-1})^\star) : \bigotimes_{j=0}^{n_i} \Gamma_j^i \to \bigotimes_{j=0}^{o^i} T_j^i \otimes \vec{\Gamma}_j^i$.

Since $\pi_1' \sim \pi_2'$ we deduce that $o^1 = o^2$ named o, and that there exists $\langle \delta, \vec{k} \rangle$ and $\vec{L}$ such that :

- $[(\langle \delta, \vec{k} \rangle \multimap f_1)] \rho_0^1 [(h_1 \circ \langle \alpha_1, \vec{f_1} \rangle] \vec{\pi_1})_0 / \vec{b_0^1}] \{L_0\} \sim [1 \multimap f_2] \rho_0^2 [(h_2 \circ \langle \alpha_2, \vec{f_2} \rangle] \vec{\pi_2})_0 / \vec{b_0^2}]$,
- $\forall j = 1, \dots, o$:
  $\rho_j^1 [(h_1 \circ \langle \alpha_1, \vec{f_1} \rangle] \vec{\pi_1})_j / \vec{b_j^1}] \{L_j\} \sim [k_j] \rho_{\delta^{-1}(j)}^2 [(h_2 \circ \langle \alpha_2, \vec{f_2} \rangle] \vec{\pi_2})_{\delta^{-1}(j)} / \vec{b}_{\delta^{-1}(j)}^2]$,
- $(\bigotimes_{j=0}^{o} L_j) \circ (1 \otimes (\delta^{-1})^\star) \circ \tau_2(\Theta_2 \otimes (\sigma_2^{-1} \circ \alpha_2^{-1})^\star) \circ \eta_2 = \tau_1(\Theta_1 \otimes (\sigma_1^{-1} \circ \alpha_1^{-1})^\star) \circ \eta_1$.

(or equivalent configurations depending on $\langle \delta, \vec{k} \rangle$, $\vec{L}$ and using congruence rules symmetrically.)

We apply the induction hypothesis on $[(\langle\delta,\vec{k}\rangle \multimap f_1)]\rho_0^1[(h_1\circ\langle\alpha_1,\vec{f_1}\rangle]\vec{\pi_1})_0/\vec{b_0^1}]\{L_0\}$ and $[1\multimap f_2]\rho_0^2[(h_2\circ\langle\alpha_2,\vec{f_2}\rangle]\vec{\pi_2})_0/\vec{b_0^2}]$

$$R_0^1 = \dfrac{\dfrac{\dfrac{\rho_0^1}{T_0^1,\vec{b_0^1}\vdash\vec{d_1}\multimap c_1}\qquad 1\multimap(\langle\delta,\vec{k}\rangle\multimap f_1)}{T_0^1\vdash \vec{b_0^1}\multimap(\vec{d_2}\multimap a)} \qquad \left(\dfrac{(([h_1\circ\langle\alpha_1,\vec{f_1}\rangle]\vec{\pi_1})_0)_j}{(\vec{\Gamma}_0^1)_j\vdash b_{0,j}^1}\right)_{j=1}^{l_0^1} \qquad L_0}{T_0^2\otimes\vec{\Gamma}_0^2\vdash\vec{d_2}\multimap a}$$

$$R_0^2 = \dfrac{\dfrac{\dfrac{\rho_0^2}{T_0^2,\vec{b_0^2}\vdash\vec{d_2}\multimap c_2}\qquad 1\multimap(1\multimap f_2)}{T_0^2\vdash \vec{b_0^2}\multimap(\vec{d_2}\multimap a)} \qquad \left(\dfrac{(([h_2\circ\langle\alpha_2,\vec{f_2}\rangle]\vec{\pi_2})_0)_j}{(\vec{\Gamma}_0^2)_j\vdash b_{0,j}^2}\right)_{j=1}^{l_0^2} \qquad 1}{T_0^2\otimes\vec{\Gamma}_0^2\vdash\vec{d_2}\multimap a}$$

and we have $R_0^1\sim R_0^2$. So $l_0^2=l_0^1$ named l_0.

Due to congruence rules Figure 9.4 we can say that there exists $\vec{g_0}:\vec{b_0^2}\to\vec{b_0^1}$, $L_0^0:T_0^2\to T_0^1$ and $\langle L_1^0,\dots,L_{l_0}^0\rangle:\vec{\Gamma}_0^2\to\vec{\Gamma}_0^1$ with $L_0=\bigotimes_{j=0}^{l_0}L_j^0$ such that:

- $[\langle\delta,\vec{k}\rangle\multimap f_1]\rho_0^1\{L_0^0\oplus\vec{g_0}\}\sim[1\multimap f_2]\rho_0^2$,
- $\forall j=1,\dots,l_0$: $(([h_1\circ\langle\alpha_1,\vec{f_1}\rangle]\vec{\pi_1})_0)_j\{L_j^0\}\sim[g_j^0]\bigl(([h_2\circ\langle\alpha_2,\vec{f_2}\rangle]\vec{\pi_2})_0\bigr)_j$.

(or equivalent configurations using congruence rules symmetrically)

Similarly for $j=1,\dots,o$:

$$R_j^1 = \dfrac{\dfrac{\dfrac{\rho_j^1}{T_j^1,\vec{b_j^1}\vdash d_j^1}\qquad 1\multimap 1}{T_j^1\vdash\vec{b_j^1}\multimap d_j^1}\qquad \left(\dfrac{(([h_1\circ\langle\alpha_1,\vec{f_1}\rangle]\vec{\pi_1})_j)_m}{(\vec{\Gamma}_j^1)_m\vdash b_{j,m}^1}\right)_{m=1}^{l_j^1}\qquad L_j}{T_{\delta^{-1}(j)}^2\otimes\vec{\Gamma}_{\delta^{-1}(j)}^2\vdash d_j^1}$$

$$R_j^2 = \dfrac{\dfrac{\dfrac{\rho_{\delta^{-1}(j)}^2}{T_{\delta^{-1}(j)}^2,\vec{b}_{\delta^{-1}(j)}^2\vdash d_{\delta^{-1}(j)}^2}\qquad 1\multimap k_j}{T_{\delta^{-1}(j)}^2\vdash\vec{b}_{\delta^{-1}(j)}^2\multimap d_j^1}\qquad \left(\dfrac{(([h_2\circ\langle\alpha_2,\vec{f_2}\rangle]\vec{\pi_2})_{\delta^{-1}(j)})_m}{(\vec{\Gamma}_{\delta^{-1}(j)}^2)_m\vdash b_{\delta^{-1}(j),m}^2}\right)_{m=1}^{l_{\delta^{-1}(j)}^2}\qquad 1}{T_{\delta^{-1}(j)}^2\otimes\vec{\Gamma}_{\delta^{-1}(j)}^2\vdash d_j^1}$$

with $R_j^1\sim R_j^2$ so $l_j^1=l_{\delta^{-1}(j)}^2$ named l_j and we can consider $\vec{g_j}$ and L_m^js such that:

- $\rho_j^1\{L_0^j\oplus\vec{g_j}\}\sim[k_j]\rho_{\delta^{-1}(j)}^2$,
- $\forall m=1,\dots,l_j$: $(([h_1\circ\langle\alpha_1,\vec{f_1}\rangle]\vec{\pi_1})_j)_m\{L_m^j\}\sim[g_m^j](([h_2\circ\langle\alpha_2,\vec{f_2}\rangle]\vec{\pi_2})_{\delta^{-1}(j)})_m$.

(or equivalent configurations using congruence rules symmetrically)

We have $(\bigotimes_{j=0}^{o} L_j) \circ ((1 \otimes \delta^{-1})^\star \otimes (1 \otimes \delta^{-1})^\star) \circ (\Theta_2 \otimes (\sigma_2^{-1} \circ \alpha_2^{-1})^\star) \circ \eta_2 = (\Theta_1 \otimes (\sigma_1^{-1} \circ \alpha_1^{-1})^\star) \circ \eta_1$.

We use those informations in $\pi_1 \sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{\rho_0^1}{T_0^1, \vec{b_0^1} \vdash \vec{d_1} \multimap c_1} \qquad \left(\dfrac{\rho_j^1}{T_j^1, \vec{b_j^1} \vdash d_j^1}\right)_{j=1}^{o} \quad \Theta_1, h_1}{\Gamma_0^1, \vec{c_1} \vdash c_1} \qquad \langle \alpha_1, \vec{f_1} \rangle \multimap f_1}{\Gamma_0^1 \vdash \vec{a_1} \multimap a} \quad \left(\dfrac{\pi_j^1}{\Gamma_j^1 \vdash a_j^1}\right)_{j=1}^{n} \quad \eta_1}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{\rho_0^1}{T_0^1, \vec{b_0^1} \vdash \vec{d_1} \multimap c_1} \quad \left(\dfrac{\rho_j^1}{T_j^1, \vec{b_j^1} \vdash d_j^1}\right)_{j=1}^{o} \quad 1, h_1}{\bigotimes_{j=0}^{o} T_j^1, c_j^1 \vdash c_1} \qquad \langle \alpha_1, \vec{f_1} \rangle \multimap f_1}{\bigotimes_{j=0}^{o} T_j^1 \vdash \vec{a_1} \multimap a} \quad \left(\dfrac{\pi_j^1}{\Gamma_j^1 \vdash a_j^1}\right)_{j=1}^{n} \quad (\Theta_1 \otimes 1) \circ \eta_1}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_1]\rho_0^1}{T_0^1, \vec{b_0^1} \vdash \vec{d_1} \multimap a} \quad \left(\dfrac{\rho_j^1}{T_j^1, \vec{b_j^1} \vdash d_j^1}\right)_{j=1}^{o} \quad (1, h_1) \circ (1, \langle \alpha_1, \vec{f_1} \rangle)}{\bigotimes_{j=0}^{o} T_j^1, a_j^1 \vdash a} \quad 1}{\bigotimes_{j=0}^{o} T_j^1 \vdash \vec{a_1} \multimap a} \quad \left(\dfrac{\pi_j^1}{\Gamma_j^1 \vdash a_j^1}\right)_{j=1}^{n} \quad (\Theta_1 \otimes 1) \circ \eta_1}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_1]\rho_0^1}{T_0^1, \vec{b_0^1} \vdash \vec{d_1} \multimap a} \quad \left(\dfrac{\rho_j^1}{T_j^1, \vec{b_j^1} \vdash d_j^1}\right)_{j=1}^{o} \quad 1}{\bigotimes_{j=0}^{o} T_j^1, b_j^1 \vdash a} \quad 1}{\bigotimes_{j=0}^{o} T_j^1 \vdash \vec{b_1} \multimap a} \quad \left(\left(\dfrac{[h_j^1 \circ f_{j,m}^1]\pi_{j,m}^1}{\Gamma_{j,m}^1 \vdash b_{j,m}^1}\right)_{j=0}^{o}\right)_{m=1}^{l_j} \quad (\Theta_1 \otimes (\sigma_1^{-1} \circ \alpha_1^{-1})^\star) \circ \eta_1}{\Delta \vdash a}$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_1]\rho_0^1}{T_0^1, \vec{b_0^1} \vdash \vec{d_1} \multimap a} \quad \left(\dfrac{\rho_j^1}{T_j^1, \vec{b_j^1} \vdash d_j^1}\right)_{j=1}^{o}}{\bigotimes_{j=0}^{o} T_j^1, b_j^1 \vdash a}\;1}{\bigotimes_{j=0}^{o} T_j^1 \vdash \vec{b_1} \multimap a}\;1 \quad \left(\left(\dfrac{[h_j^1 \circ f_{j,m}^1]\pi_{j,m}^1}{\Gamma_{j,m}^1 \vdash b_{j,m}^1}\right)_{j=0}^{o}\right)_{m=1}^{l_j}}{\Delta \vdash a}\;B$$

with $A = (\Theta_2 \otimes (\sigma_2^{-1} \circ \alpha_2^{-1})^\star) \circ \eta_2$ and $B = (\bigotimes_{j=0}^{o} \bigotimes_{m=0}^{l_j} L_m^j) \circ (1 \otimes (\delta^{-1})^\star) \otimes (1 \otimes (\delta^{-1})^\star) \circ A$.

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_1]\rho_0^1\{L_0^0 \oplus 1\}}{T_0^2, \vec{b_0^1} \vdash \vec{d_1} \multimap a} \quad \left(\dfrac{\rho_j^1\{L_j^0 \oplus 1\}}{T_{\delta^{-1}(j)}^2, \vec{b_j^1} \vdash d_j^1}\right)_{j=1}^{o}}{T_0^2 \otimes \bigotimes_{j=1}^{o} T_{\delta^{-1}(j)}^2, \vec{b_1} \vdash a}\;1}{T_0^2 \otimes \bigotimes_{j=1}^{o} T_{\delta^{-1}(j)}^2 \vdash \vec{b_1} \multimap a}\;1 \quad \left(\left(\dfrac{[h_j^1 \circ f_{j,m}^1]\pi_{j,m}^1\{L_m^j\}}{\Gamma_{\delta^{-1}(j),m}^2 \vdash b_{j,m}^1}\right)_{j=0}^{o}\right)_{m=1}^{l_j}}{\Delta \vdash a}\;C$$

with $C = (((1 \otimes \delta^{-1})^\star) \otimes (1 \otimes \delta^{-1})^\star)) \circ A$ for simplicity reason we use $\delta^{-1}(0) = 0$ for $\Gamma_{\delta^{-1}(j),m}^2$.

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_1]\rho_0^1\{L_0^0 \oplus 1\}}{T_0^2, \vec{b_0^1} \vdash \vec{d_1} \multimap a} \quad \left(\dfrac{\rho_j^1\{L_j^0 \oplus 1\}}{T_{\delta^{-1}(j)}^2, \vec{b_j^1} \vdash d_j^1}\right)_{j=1}^{o}}{T_0^2 \otimes \bigotimes_{j=1}^{o} T_{\delta^{-1}(j)}^2, \vec{b_1} \vdash a}\;1}{T_0^2 \otimes \bigotimes_{j=1}^{o} T_{\delta^{-1}(j)}^2 \vdash \vec{b_1} \multimap a}\;1 \quad \left(\left(\dfrac{[g_m^j]((([h_2 \circ \langle \alpha_2, \vec{f_2} \rangle]\vec{\pi_2})_{\delta^{-1}(j)})_m}{\Gamma_{\delta^{-1}(j),m}^2 \vdash b_{j,m}^1}\right)_{j=0}^{o}\right)_{m=1}^{l_j}}{\Delta \vdash a}\;C$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_1]\rho_0^1\{L_0^0 \oplus 1\}}{T_0^2, \vec{b_0^1} \vdash \vec{d_1} \multimap a} \quad \left(\dfrac{\rho_j^1\{L_j^0 \oplus 1\}}{T_{\delta^{-1}(j)}^2, \vec{b_j^1} \vdash d_j^1}\right)_{j=1}^{o}}{(T_0^2, \vec{b_0^2}) \otimes \bigotimes_{j=1}^{o} T_{\delta^{-1}(j)}^2, \vec{b_1} \vdash a}\;1}{T_0^2 \otimes \bigotimes_{j=1}^{o} T_{\delta^{-1}(j)}^2 \vdash \vec{b_2} \multimap a}\;\langle (1 \otimes \delta), \vec{g} \rangle \multimap 1 \quad \left(\left(\dfrac{(([h_2 \circ \langle \alpha_2, \vec{f_2} \rangle]\vec{\pi_2})_j)_m}{\Gamma_{j,m}^2 \vdash b_{j,m}^2}\right)_{j=0}^{o}\right)_{m=1}^{l_j}}{\Delta \vdash a}\;D$$

with $D = (((1 \otimes \delta^{-1})^\star) \otimes 1) \circ A$.

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_1]\rho^1_0\{L^0_0 \oplus \langle 1, \vec{g_0}\rangle\}}{T^2_0, \vec{b^1_0} \vdash \vec{d_1} \multimap a} \quad \left(\dfrac{\rho^1_j\{L^0_j \oplus \langle 1, \vec{g_j}\rangle\}}{T^2_{\delta^{-1}(j)}, \vec{b^2_{\delta^{-1}(j)}} \vdash d^1_j}\right)^o_{j=1}}{(T^2_0, \vec{b^2_0}) \otimes \bigotimes^o_{j=1} T^2_{\delta^{-1}(j)}, \vec{b_2} \vdash a}\; 1 \oplus (1 \otimes (\delta^{-1}))^\star}{T^2_0 \otimes \bigotimes^o_{j=1} T^2_{\delta^{-1}(j)} \vdash \vec{b_2} \multimap a}\; 1 \quad \left(\left(\dfrac{(([h_2 \circ \langle \alpha_2, \vec{f_2}\rangle]\vec{\pi_2})_j)_m}{\Gamma^2_{j,m} \vdash b^2_{j,m}}\right)^o_{j=0}\right)^{l_j}_{m=1}}{\Delta \vdash a}\; D$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_1]\rho^1_0\{L^0_0 \oplus \langle 1, \vec{g_0}\rangle\}}{T^2_0, \vec{b^2_0} \vdash \vec{d_1} \multimap a} \quad \left(\dfrac{[k_j]\rho^2_{\delta^{-1}(j)}}{T^2_{\delta^{-1}(j)}, \vec{b^2_{\delta^{-1}(j)}} \vdash d^1_j}\right)^o_{j=1}}{T^2_0 \otimes \bigotimes^o_{j=1} T^2_{\delta^{-1}(j)}, \vec{b_2} \vdash a}\; 1 \otimes (\delta^{-1})^\star}{T^2_0 \otimes \bigotimes^o_{j=1} T^2_{\delta^{-1}(j)} \vdash \vec{b_2} \multimap a}\; 1 \quad \left(\left(\dfrac{(([h_2 \circ \langle \alpha_2, \vec{f_2}\rangle]\vec{\pi_2})_j)_m}{\Gamma^2_{j,m} \vdash b^2_{j,m}}\right)^o_{j=0}\right)^{l_j}_{m=1}}{\Delta \vdash a}\; A$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[\langle \delta, \vec{k}\rangle \multimap f_1]\rho^1_0\{L^0_0 \oplus \langle 1, \vec{g_0}\rangle\}}{T^2_0, \vec{b^2_0} \vdash \vec{d_2} \multimap a} \quad \left(\dfrac{\rho^2_j}{T^2_j, \vec{b^2_j} \vdash d^2_j}\right)^o_{j=1}}{\bigotimes^o_{j=0} T^2_j, \vec{b_2} \vdash a}\; 1}{\bigotimes^o_{j=0} T^2_j \vdash \vec{b_2} \multimap a}\; 1 \quad \left(\left(\dfrac{(([h_2 \circ \langle \alpha_2, \vec{f_2}\rangle]\vec{\pi_2})_j)_m}{\Gamma^2_{j,m} \vdash b^2_{j,m}}\right)^o_{j=0}\right)^{l_j}_{m=1}}{\Delta \vdash a}\; A$$

$\sim$

$$\dfrac{\dfrac{\dfrac{\dfrac{[1 \multimap f_2]\rho^2_0}{T^2_0, \vec{b^2_0} \vdash \vec{d_2} \multimap a} \quad \left(\dfrac{\rho^2_j}{T^2_j, \vec{b^2_j} \vdash d^2_j}\right)^o_{j=1}}{\bigotimes^o_{j=0} T^2_j, \vec{b_2} \vdash a}\; 1}{\bigotimes^o_{j=0} T^2_j \vdash \vec{b_2} \multimap a}\; 1 \quad \left(\left(\dfrac{(([h_2 \circ \langle \alpha_2, \vec{f_2}\rangle]\vec{\pi_2})_j)_m}{\Gamma^2_{j,m} \vdash b^2_{j,m}}\right)^o_{j=0}\right)^{l_j}_{m=1}}{\Delta \vdash a}\; (\Theta_2 \otimes (\sigma_2^{-1} \circ \alpha_2^{-1})^\star) \circ \eta_2$$

$\sim \cdots \sim \pi_2.$ □

Theorem 10.1.16 (Injectivity of the semantics)**.** *If two derivations type the same λ-term and β-reduce to congruent derivations then they are congruent.*

Proof. Following from Lemma 10.1.15. □

Lemma 10.1.17. *If $\pi_1 \sim \pi_2$ and $\pi_1 \to_\beta \pi_1', \pi_2 \to_\beta \pi_2'$ by contracting the same redex then $\pi_1' \sim \pi_2'$.*

Proof. If the contracted redex is the subject to one of the Figure 9.4 rules, we proceed by induction on π^1_0 and π^2_0 the subderivations corresponding to the abstraction subterm. Otherwise it is sufficient to observe that for any derivation π and appropriate f and η: $([f]\pi\{\eta\})' = [f]\pi'\{\eta\}$ (the derivations obtained by contracting a same specific redex). □

We extend the β-reduction on derivations to congruence classes.

Now we will explore the link between the β-reduction on derivations and the intersection type disctributor $\mathsf{T}_{\vec{x}}()$ defined in Section 9.3.

In the following, we use $n = \mathsf{len}(\vec{x})$ and

$$\begin{aligned} Sub_{\vec{x}}^{M,x,N}(\Delta, a) \quad = \quad & \int^{\vec{a},\vec{c}\in !D}\int^{c\in D}\int^{\Gamma_0,\ldots,\Gamma_{\mathsf{len}(\vec{a})}\in !D^n} \mathsf{T}_{\vec{x}\oplus\langle x\rangle}(M)(\Gamma_0\oplus\langle\vec{c}\rangle, c) \\ & \times\Pi_{i=1}^{\mathsf{len}(\vec{a})}\mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i)\times !D^n(\Delta, \textstyle\bigotimes_{i=0}^{\mathsf{len}(\vec{a})}\Gamma_i)\times D(\vec{c}\multimap c, \vec{a}\multimap a). \end{aligned}$$

We use $\diamond_{\vec{b}}$ for the unique morphism $f : \vec{b} \to \langle\rangle$.

Lemma 10.1.18. *Let $M \in \Lambda$, $\mathrm{FV}(M) \subseteq \vec{x}$, and $x \in \mathbb{V}$, we have the following natural isomorphism:*

$$\mathsf{T}_{\vec{x}}(M)(\Delta, a) \cong \mathsf{T}_{\vec{x}\oplus\langle\langle\rangle\rangle}(M)(\Delta \oplus \langle\langle\rangle\rangle, a).$$

Lemma 10.1.19. *Let $M \in \Lambda$, $\mathrm{FV}(M) \subseteq \vec{x}$, $y \notin \vec{x}$ and $\vec{b} \in !D$, we have a natural isomorphism:*

$$\mathsf{T}_{\vec{x}}(M)(\Delta \oplus \langle\diamond_{\vec{b}}\rangle, a) : \mathsf{T}_{\vec{x}\oplus\langle y\rangle}(M)(\Delta \oplus \langle\langle\rangle\rangle, a) \cong \mathsf{T}_{\vec{x}\oplus\langle y\rangle}(M)(\Delta \oplus \langle\vec{b}\rangle, a).$$

Lemma 10.1.20. *Let $M \in \Lambda$, $\mathrm{FV}(M) \subseteq \vec{x}$ and $y \notin \vec{x}$,*

$$\mathsf{T}_{\vec{x}}(\lambda y.M)(\Delta, a) \cong \int^{b\in D}\int^{\vec{b}\in !D} \mathsf{T}_{\vec{x}\oplus\langle y\rangle}(M)(\Delta \oplus \langle\vec{b}\rangle, b) \times D(\vec{b} \multimap b, a).$$

Lemma 10.1.21. *Let $M \in \Lambda$, $\mathrm{FV}(M) \subseteq \vec{x}$ and $i, j \in [n]$, such that $\theta_{i,j} : [n] \to [n]$ is a permutation that swapes i and j,*

$$\mathsf{T}_{\vec{x}}(M)(\Delta, a) \cong \mathsf{T}_{\vec{x}\{\theta_{i,j}\}}(M)(\Delta\{\theta_{i,j}\}, a).$$

Lemma 10.1.22. *Given $M, N \in \Lambda$, $(\mathrm{FV}(M)/\{x\}) \cup \mathrm{FV}(N) \subseteq \vec{x}$ and $x \notin \vec{x}$, there is an isomorphism*

$$S_{\Delta,a}^{M,x,N} : Sub_{\vec{x}}^{M,x,N}(\Delta, a) \cong \mathsf{T}_{\vec{x}}(M[N/x])(\Delta, a).$$

Proof. By induction on M:

- if $M = x$ then $M\{N/x\} = N$ and

$$
\begin{aligned}
& Sub_{\vec{x}}^{M,x,N}(\Delta, a) \\
\cong\ & \int^{\vec{a},\vec{c}\in !D} \int^{c\in D} \int^{\vec{\Gamma}=\langle\Gamma_0,\ldots,\Gamma_{\mathsf{len}(\vec{a})}\rangle\in !D^n} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle\rangle\rangle) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \\
& \quad \times !D(\vec{c}, \langle c\rangle) \times !D^n(\Delta, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i) \times D(\vec{c} \multimap c, \vec{a} \multimap a) \\
\cong\ & \int^{\vec{a},\vec{c}} \int^{c} \int^{\vec{\Gamma}} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle\rangle\rangle) \times !D(\vec{c}, \langle c\rangle) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \\
& \quad \times !D^n(\Delta, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i) \times !D^{op}(\vec{c}, \vec{a}) \times D(c, a)
\end{aligned}
$$

applying Yoneda Lemma 7.2.3 on c we get

$$
\begin{aligned}
\cong\ & \int^{\vec{a},\vec{c}} \int^{\vec{\Gamma}} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle\rangle\rangle) \times !D(\vec{c}, \langle a\rangle) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times !D^n(\Delta, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i) \\
& \quad \times !D^{op}(\vec{c}, \vec{a})
\end{aligned}
$$

then on $\vec{c}$:

$$\cong\ \int^{\vec{a}} \int^{\vec{\Gamma}} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle\rangle\rangle) \times !D(\vec{a}, \langle a\rangle) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times !D^n(\Delta, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i)$$

then on $\vec{a}$:

$$\cong\ \int^{\Gamma_0,\Gamma_1} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle\rangle\rangle) \times \mathsf{T}_{\vec{x}}(N)(\Gamma_1, a) \times !D^n(\Delta, \Gamma_0 \otimes \Gamma_1)$$

then on Γ_0:

$$\cong\ \int^{\Gamma_1} \mathsf{T}_{\vec{x}}(N)(\Gamma_1, a) \times !D^n(\Delta, \Gamma_1)$$

finaly on Γ_1:

$$\cong\ \mathsf{T}_{\vec{x}}(N)(\Delta, a)$$

and we get

$$S_{\Delta,a}^{M,x,N} \quad : \quad \langle \vec{a}, \vec{c}, c, \langle\Gamma_i\rangle_{i=0}^{\mathsf{len}(\vec{a})}, \widetilde{g, \langle\diamond_{\Gamma_0}}, \langle h\rangle\rangle, \langle\phi_i\rangle_{i=1}^{\mathsf{len}(\vec{a})}, \eta\rangle \mapsto [g(\widetilde{h)]\phi_1}\{\eta\};$$

where the ϕ_i's are classes of congruences of derivations.

- if $M = y \neq x$ then $M\{N/x\} = M$ and

$$
\begin{aligned}
& Sub_{\vec{x}}^{M,x,N}(\Delta, a) \\
\cong\ & \int^{\vec{a},\vec{c}\in !D} \int^{c\in D} \int^{\vec{\Gamma}=\langle\Gamma_0,\ldots,\Gamma_{\mathsf{len}(\vec{a})}\rangle\in !D^n} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle c\rangle, \ldots, \langle\rangle\rangle) \times !D(\vec{c}, \langle\rangle) \\
& \quad \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times !D^n(\Delta, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i) \times D(\vec{c} \multimap c, \vec{a} \multimap a) \\
\cong\ & \int^{\vec{a},\vec{c}} \int^{c} \int^{\vec{\Gamma}} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle c\rangle, \ldots, \langle\rangle\rangle) \times !D(\vec{c}, \langle\rangle) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \\
& \quad \times !D^n(\Delta, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i) \times !D^{op}(\vec{c}, \vec{a}) \times D(c, a)
\end{aligned}
$$

applying Yoneda Lemma 7.2.3 on c we get

$$
\begin{aligned}
\cong\ & \int^{\vec{a},\vec{c}} \int^{\vec{\Gamma}} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle a\rangle, \ldots, \langle\rangle\rangle) \times !D(\vec{c}, \langle\rangle) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \\
& \quad \times !D^n(\Delta, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i) \times !D^{op}(\vec{c}, \vec{a})
\end{aligned}
$$

then on $\vec{c}$:

$$
\begin{aligned}
\cong\ & \int^{\vec{a}} \int^{\vec{\Gamma}} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle a\rangle, \ldots, \langle\rangle\rangle) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times !D^n(\Delta, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i) \\
& \quad \times !D^{op}(\langle\rangle, \vec{a})
\end{aligned}
$$

and on $\vec{a}$:

$$\cong\ \int^{\Gamma_0} !D^n(\Gamma_0, \langle\langle\rangle, \ldots, \langle a\rangle, \ldots, \langle\rangle\rangle) \times !D^n(\Delta, \Gamma_0)$$

then on Γ_0:

$$\cong\ !D^n(\Delta, \langle\langle\rangle, \ldots, \langle a\rangle, \ldots, \langle\rangle\rangle)$$

and we get

$$S^{M,x,N}_{\Delta,a} \quad : \quad \langle g, \widetilde{\Gamma, \langle h\rangle}, \eta\rangle \mapsto [g]\widetilde{\langle h\rangle\{\eta\}};$$

- if $M = \lambda y.M'$ then

$$Sub^{M,x,N}_{\vec{x}}(\Delta, a)$$

$$\cong \int^{\vec{a},\vec{c}\in !D} \int^{c\in D} \int^{\vec{\Gamma}=\langle\Gamma_0,\ldots,\Gamma_{\mathsf{len}(\vec{a})}\rangle\in !D^n} \mathsf{T}_{\vec{x}\oplus\langle x\rangle}(\lambda y.M')(\Gamma_0 \oplus \langle\vec{c}\rangle, c) \times \Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i)$$
$$\times !D^n(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a}) \times D(c, a)$$

$$\cong \int^{\vec{a},\vec{c},\vec{b}} \int^{c,b} \int^{\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle x\rangle\oplus\langle y\rangle}(M')(\Gamma_0 \oplus \langle\vec{c}\rangle \oplus \langle\vec{b}\rangle, b) \times D(\vec{b} \multimap b, c)$$
$$\times\Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i)\times !D^n(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a}) \times D(c, a)$$

using Lemma 10.1.18

$$\cong \int^{\vec{a},\vec{c},\vec{b}} \int^{c,b} \int^{\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle x\rangle\oplus\langle y\rangle}(M')(\Gamma_0 \oplus \langle\vec{c}\rangle \oplus \langle\vec{b}\rangle, b) \times D(\vec{b} \multimap b, c)$$
$$\times\Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}\oplus\langle y\rangle}(N)(\Gamma_i \oplus \langle\rangle, a_i)\times !D^{n+1}(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a}) \times D(c, a)$$

$$\cong \int^{\vec{a},\vec{c},\vec{b}} \int^{c,b} \int^{\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle x\rangle\oplus\langle y\rangle}(M')(\Gamma_0 \oplus \langle\vec{c}\rangle \oplus \langle\vec{b}\rangle, b) \times D(\vec{b} \multimap b, c)$$
$$\times\Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}\oplus\langle y\rangle}(N)(\Gamma_i \oplus \langle\rangle, a_i)\times !D^{n+1}(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a}) \times D(c, a)$$

using Lemma 10.1.21

$$\cong \int^{\vec{a},\vec{c},\vec{b}} \int^{c,b} \int^{\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle y\rangle\oplus\langle x\rangle}(M')(\Gamma_0 \oplus \langle\vec{b}\rangle \oplus \langle\vec{c}\rangle, b) \times D(\vec{b} \multimap b, c)$$
$$\times\Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}\oplus\langle y\rangle}(N)(\Gamma_i \oplus \langle\rangle, a_i)\times !D^{n}(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a}) \times D(c, a)$$

using Yoneda Lemma 7.2.3

$$\cong \int^{\vec{a},\vec{c},\vec{b},\vec{d}} \int^{c,b} \int^{\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle y\rangle\oplus\langle x\rangle}(M')(\Gamma_0 \oplus \langle\vec{d}\rangle \oplus \langle\vec{c}\rangle, b) \times D(\vec{b} \multimap b, c) \times \Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}\oplus\langle y\rangle}(N)$$
$$(\Gamma_i \oplus \langle\rangle, a_i)\times !D^{n}(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a}) \times D(c, a)\times !D(\vec{b},\vec{d})$$

using Yoneda Lemma 7.2.3 on c:

$$\cong \int^{\vec{a},\vec{c},\vec{b},\vec{d}} \int^{b} \int^{\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle y\rangle\oplus\langle x\rangle}(M')(\Gamma_0 \oplus \langle\vec{d}\rangle \oplus \langle\vec{c}\rangle, b) \times D(\vec{b} \multimap b, a)$$
$$\times\Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}\oplus\langle y\rangle}(N)(\Gamma_i \oplus \langle\rangle, a_i)\times !D^{n}(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a})\times !D(\vec{b},\vec{d})$$

On the other hand:

$$\mathsf{T}_{\vec{x}}((\lambda y.M')[N/x])(\Delta, a)$$

Using Lemma 10.1.20

$$\cong \mathsf{T}_{\vec{x}}(\lambda y.(M'[N/x]))(\Delta, a)$$

$$\cong \int^{b} \int^{\vec{b}} \mathsf{T}_{\vec{x}\oplus\langle y\rangle}(M'[N/x]))(\Delta \oplus \langle\vec{b}\rangle, b) \times D(\vec{b} \multimap b, a)$$

by induction hypothesis:

$$\cong \int^{b} \int^{\vec{b}} Sub^{M',x,N}_{\vec{x}\oplus\langle y\rangle}(\Delta \oplus \langle\vec{b}\rangle, b) \times D(\vec{b} \multimap b, a)$$

$$\cong \int^{\vec{a},\vec{b},\vec{c},\vec{b_0},\ldots,\vec{b_k},c,b,\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle y\rangle\oplus\langle x\rangle}(M')(\Gamma_0 \oplus \langle\vec{b_0}\rangle \oplus \langle\vec{c}\rangle, c) \times \Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}\oplus\langle y\rangle}(N)(\Gamma_i \oplus \langle\vec{b_i}\rangle, a_i)$$
$$\times !D^{n}(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a}) \times D(c, b)\times !D(\vec{b}, \bigoplus^{k}_{i=1} \vec{b_i}) \times D(\vec{b} \multimap b, a)$$

using Yoneda Lemma 7.2.3 on c:

$$\cong \int^{\vec{a},\vec{b},\vec{c},\vec{b_0},\ldots,\vec{b_k},b,\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle y\rangle\oplus\langle x\rangle}(M')(\Gamma_0 \oplus \langle\vec{b_0}\rangle \oplus \langle\vec{c}\rangle, b) \times \Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}\oplus\langle y\rangle}(N)(\Gamma_i \oplus \langle\vec{b_i}\rangle, a_i)$$
$$\times !D^{n}(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a})\times !D(\vec{b}, \bigoplus^{k}_{i=1} \vec{b_i}) \times D(\vec{b} \multimap b, a)$$

using Lemma 10.1.19

$$\cong \int^{\vec{a},\vec{b},\vec{c},\vec{b_0}} \int^{b} \int^{\vec{\Gamma}} \mathsf{T}_{\vec{x}\oplus\langle y\rangle\oplus\langle x\rangle}(M')(\Gamma_0 \oplus \langle\vec{b_0}\rangle \oplus \langle\vec{c}\rangle, b) \times \Pi^{\mathsf{len}(\vec{a})}_{i=1}\mathsf{T}_{\vec{x}\oplus\langle y\rangle}(N)(\Gamma_i \oplus \langle\rangle, a_i)$$
$$\times !D^{n}(\Delta, \bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)\times !D^{op}(\vec{c},\vec{a})\times !D(\vec{b},\vec{b_0}) \times D(\vec{b} \multimap b, a)$$

And we can conclude

$$\begin{aligned} S^{M,x,N}_{\Delta,a} \quad : \quad & \langle \vec{a}, \vec{c}, \vec{b}, b, c, f_1 \multimap f_2, \langle \Gamma_i \oplus \widetilde{\langle\rangle\rangle}^{\mathsf{len}(\vec{a})}_{i=0}, \phi_0, g_1 \multimap g_2, \langle \phi_i \rangle^{\mathsf{len}(\vec{a})}_{i=1}, \eta \rangle \\ & \mapsto \langle \vec{b}, b, S^{M',x,N}_{\Delta \oplus \langle \vec{b} \rangle, b}(\langle \vec{a}, \langle \Gamma_i \oplus \langle\rangle \rangle^{\mathsf{len}(\vec{a})}_{i=0} \widetilde{, \phi_0}\{f_1\}, \langle \phi_i \rangle^{\mathsf{len}(\vec{a})}_{i=1}, \eta \rangle), g_1 \multimap f_2 \circ g_2 \rangle; \end{aligned}$$

- if $M = PL$ then

$$\begin{aligned} & Sub^{M,x,N}_{\vec{x}}(\Delta, a) \\ \cong \quad & \int^{\vec{a},\vec{c} \in !D} \int^{c \in D} \int^{\vec{\Gamma} \in !D^n} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(PL)(\Gamma_0 \oplus \langle \vec{c} \rangle, c) \times \Pi^{\mathsf{len}(\vec{a})}_{i=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \\ & \qquad \times !D^n(\Delta, \textstyle\bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i) \times !D^{op}(\vec{c}, \vec{a}) \times D(c, a) \end{aligned}$$

using Yoneda Lemma 7.2.3 on $\vec{c}$ and c:

$$\cong \quad \int^{\vec{a}} \int^{\vec{\Gamma}} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(PL)(\Gamma_0 \oplus \langle \vec{a} \rangle, a) \times \Pi^{\mathsf{len}(\vec{a})}_{i=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times !D^n(\Delta, \textstyle\bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i)$$

developing $\mathsf{T}_{\vec{x} \oplus \langle x \rangle}(PL)$:

$$\begin{aligned} \cong \quad & \int^{\vec{a}, \vec{b} = \langle b_1, \ldots, b_k \rangle} \int^{\vec{\Gamma}} \int^{\Gamma'_0 \oplus \langle \vec{a_0} \rangle, \ldots, \Gamma'_k \oplus \langle \vec{a_k} \rangle \in !D^{n+1}} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(P)(\Gamma'_0 \oplus \langle \vec{a_0} \rangle, \vec{b} \multimap a) \\ & \qquad \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(L)(\Gamma'_i \oplus \langle \vec{a_i} \rangle, b_i) \times !D^{n+1}(\Gamma_0 \oplus \langle \vec{a} \rangle, \textstyle\bigotimes^l_{i=0} \Gamma'_i \oplus \langle \vec{a_i} \rangle \times !D^n(\Delta, \textstyle\bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i) \\ & \qquad \times \Pi^{\mathsf{len}(\vec{a})}_{i=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times !D^n(\Delta, \textstyle\bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i) \\ \cong \quad & \int^{\vec{a}, \vec{b}} \int^{\vec{a_0}, \ldots, \vec{a_k}} \int^{\vec{\Gamma}, \vec{\Gamma'}} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(P)(\Gamma'_0 \oplus \langle \vec{a_0} \rangle, \vec{b} \multimap a) \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(L)(\Gamma'_i \oplus \langle \vec{a_i} \rangle, b_i) \\ & \qquad \times !D^n(\Gamma_0, \textstyle\bigotimes^l_{i=0} \Gamma'_i) \times !D(\vec{a}, \textstyle\bigotimes^l_{i=0} \langle \vec{a_i} \rangle) \times !D^n(\Delta, \textstyle\bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i) \times \Pi^{\mathsf{len}(\vec{a})}_{i=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \\ & \qquad \times !D^n(\Delta, \textstyle\bigotimes^{\mathsf{len}(\vec{a})}_{i=0} \Gamma_i) \end{aligned}$$

using Yoneda Lemma 7.2.3 on Γ_0 and $\vec{a}$:

$$\begin{aligned} \cong \quad & \int^{\Gamma_1, \ldots, \Gamma_l} \int^{\vec{b}} \int^{\vec{\Gamma'}} \int^{\vec{a_0}, \ldots, \vec{a_k}} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(P)(\Gamma'_0 \oplus \langle \vec{a_0} \rangle, \vec{b} \multimap a) \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(L)(\Gamma'_i \oplus \langle \vec{a_i} \rangle, b_i) \\ & \qquad \times \Pi^l_{i=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, c_i) \times !D^n(\Delta, \textstyle\bigotimes^l_{i=1} \Gamma_i \otimes \textstyle\bigotimes^k_{i=0} \Gamma'_i) \quad \textit{with } \textstyle\bigoplus^k_{i=0} \vec{a_i} = \langle c_1, \ldots, c_l \rangle. \end{aligned}$$

we denote $\forall j \in [k], \vec{\Gamma_j} = \langle \Gamma_{j,1}, \ldots, \Gamma_{j,k_j} \rangle$ the partition of $\langle \Gamma_1, \ldots, \Gamma_l \rangle$ induced by $\vec{a_j} = \langle a_{j,1}, \ldots, a_{j,k_j} \rangle$ of $\langle c_1, \ldots, c_l \rangle$ and $\mathsf{T}_{\vec{x}}(N)(\vec{\Gamma_i}, \vec{a_i}) = \Pi^{k_i}_{j=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_{i,j}, a_{i,j})$.

$$\begin{aligned} \cong \quad & \int^{\Gamma_1, \ldots, \Gamma_l} \int^{\vec{b}} \int^{\vec{\Gamma'}} \int^{\vec{a_0}, \ldots, \vec{a_k}} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(P)(\Gamma'_0 \oplus \langle \vec{a_0} \rangle, \vec{b} \multimap a) \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(L)(\Gamma'_i \oplus \langle \vec{a_i} \rangle, b_i) \\ & \qquad \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times \mathsf{T}_{\vec{x}}(N)(\vec{\Gamma_0}, \vec{a_0}) \times !D^n(\Delta, \textstyle\bigotimes^k_{i=0} \Gamma'_i \otimes \textstyle\bigotimes^l_{i=1} \Gamma_i) \end{aligned}$$

by symmetry of the tensor product and since functors preserves isomorphisms

$$\begin{aligned} \cong \quad & \int^{\Gamma_1, \ldots, \Gamma_l} \int^{\vec{b}} \int^{\vec{\Gamma'}} \int^{\vec{a_0}, \ldots, \vec{a_k}} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(P)(\Gamma'_0 \oplus \langle \vec{a_0} \rangle, \vec{b} \multimap a) \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(L)(\Gamma'_i \oplus \langle \vec{a_i} \rangle, b_i) \\ & \qquad \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times \mathsf{T}_{\vec{x}}(N)(\vec{\Gamma_0}, \vec{a_0}) \times !D^n(\Delta, \textstyle\bigotimes^k_{i=0} \Gamma'_i \otimes \vec{\Gamma_i}) \end{aligned}$$

where we set $\Gamma'_i \otimes \vec{\Gamma_i} = \Gamma'_i \otimes (\bigotimes^{k_i}_{j=1} \Gamma_{i,j})$

using Yoneda Lemma 7.2.3 multiple times:

$$\begin{aligned} \cong \quad & \int^{\Gamma_1, \ldots, \Gamma_l} \int^{\vec{b}} \int^{\vec{\Gamma'}, \vec{\Delta} = \langle \Delta_0, \ldots, \Delta_k \rangle} \int^{\vec{a_0}, \ldots, \vec{a_k}} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(P)(\Gamma'_0 \oplus \langle \vec{a_0} \rangle, \vec{b} \multimap a) \\ & \qquad \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(L)(\Gamma'_i \oplus \langle \vec{a_i} \rangle, b_i) \quad \times \Pi^k_{i=1} \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \times \mathsf{T}_{\vec{x}}(N)(\vec{\Gamma_0}, \vec{a_0}) \\ & \qquad \times !D^n(\Delta, \textstyle\bigotimes^k_{i=1} \Delta_i) \times !D^n(\Delta_0, \Gamma'_0 \otimes \vec{\Gamma_0}) \times \cdots \times !D^n(\Delta_{k_j}, \Gamma'_{k_j} \otimes \vec{\Gamma_{k_j}}) \end{aligned}$$

by cocontinuouty and commutativity we have:

$$\begin{aligned} \cong \quad & \int^{\vec{b}, \vec{a_0}} \int^{\Delta_0, \ldots, \Delta_k, \Gamma'_0} \int^{\vec{\Gamma_0}} \mathsf{T}_{\vec{x} \oplus \langle x \rangle}(P)(\Gamma'_0 \oplus \langle \vec{a_0} \rangle, \vec{b} \multimap a) \times \mathsf{T}_{\vec{x}}(N)(\vec{\Gamma_0}, \vec{a_0}) \\ & \qquad \times !D^n(\Delta_0, \Gamma'_0 \otimes \vec{\Gamma_0}) \times \Pi^k_{i=1} (\int^{\vec{a_i}, \Gamma'_i, \vec{\Gamma'_i}} \mathsf{T}_{\vec{x}}(L)(\Gamma'_i \oplus \langle \vec{a_i} \rangle, b_i) \times \mathsf{T}_{\vec{x}}(N)(\Gamma_i, a_i) \\ & \qquad \times !D^n(\Delta_i, \Gamma'_i \otimes \vec{\Gamma_i})) \times !D^n(\Delta, \textstyle\bigotimes^k_{i=1} \Delta_i) \end{aligned}$$

by definition:

$$\cong \quad \int^{\vec{b}} \int^{\Delta_0, \ldots, \Delta_k} Sub^{P,x,N}_{\vec{x}}(\Delta_0, \vec{b} \multimap a)) \times \Pi^k_{i=1} Sub^{L,x,N}_{\vec{x}}(\Delta_i, b_i)) \times !D^n(\Delta, \textstyle\bigotimes^k_{i=0} \Delta_i)$$

By definition

$$\begin{aligned}\mathsf{T}_{\vec{x}}(PL[N/x])(\Delta, a) &= \int^{\vec{b}}\int^{\Delta_0,\dots,\Delta_k} \mathsf{T}_{\vec{x}}(P[N/x])(\Delta_0, \vec{b} \multimap a)) \\ &\quad \times \Pi_{i=1}^{k} \mathsf{T}_{\vec{x}}(L[N/x])(\Delta_i, b_i)) \times !D^n(\Delta, \bigotimes_{i=0}^{k} \Delta_i)\end{aligned}$$

Using the induction hypothesis we can conclude, since isomorphisms are preserved by the coend construction.

we get

$$\begin{aligned}S^{M,x,N}_{\Delta,a} \quad : \quad & \langle \vec{a}, \vec{c}, c, \langle \Gamma_i \rangle_{i=0}^{\mathsf{len}(\vec{a})}, \langle \vec{b}, \widetilde{\langle \Gamma'_i \oplus \langle \vec{b_i} \rangle\rangle_{i=0}^{\mathsf{len}(\vec{b})}}, f_1 \multimap f_2, \langle \phi, \vec{\xi}, \theta \oplus \langle h \rangle\rangle\rangle, \vec{\phi}, \eta \rangle \\ & \mapsto \langle \vec{b}, \langle \Gamma'_i \otimes \vec{\Gamma}_i \rangle_{i=0}^{\mathsf{len}(\vec{b})}, S^{P,x,N}_{\Gamma'_0 \otimes \vec{\Gamma}_0, \vec{b} \multimap a}(\langle \Gamma'_0 \otimes \vec{\Gamma}_0, 1 \multimap f_2, \phi, ([h \circ f_1]\vec{\phi})_0, 1 \rangle), \\ & \qquad \langle S^{L,x,N}_{\Gamma'_i \otimes \vec{\Gamma}_i, b_i}(\langle \Gamma'_i \otimes \vec{\Gamma}_i, \xi_i, (\widetilde{[h \circ f_1])\vec{\phi}})_i, 1 \rangle)\rangle_{i=1}^{\mathsf{len}(\vec{b})}, \tau \circ (\theta \otimes ((\alpha \circ \beta)^{-1})^{\star}) \circ \eta \rangle.\end{aligned}$$

with $h = \langle \alpha, \vec{h} \rangle$ and $f_1 = \langle \beta, \vec{f_1} \rangle$, $\vec{\Gamma} = \langle \Gamma_i \rangle_{i=0}^{\mathsf{len}(\vec{a})}$ and $\tau : \bigoplus_{j=0}^{\mathsf{len}(\vec{b})} \Gamma'_j \otimes \vec{\Gamma} \to \bigotimes_{j=0}^{\mathsf{len}(\vec{b})} (\Gamma'_j \otimes \vec{\Gamma}_j)$ and $\vec{\Gamma}_j = ([h]\vec{\Gamma})_j$. □

Remark 10.1.23. *The isomorphism $\mathsf{T}_{\vec{x}}(M \to_\beta N)$ can be define using the isomorphism S, by induction on the reduction step $M \to_\beta N$:*

- *if $(\lambda x.M_1)M_2 \to_\beta M_1[M_2/x]$ then by definition*

$$\mathsf{T}_{\vec{x}}((\lambda x.M_1)M_2 \to_\beta M_1[M_2/x]) = S^{M_1,x,M_2};$$

- *if $\lambda y.M_1 \to_\beta \lambda y.M'_1$ then by induction hypothesis*

$$\mathsf{T}_{\vec{x}}(\lambda y.M_1 \to_\beta \lambda y.M'_1) \quad = \quad \int^{\vec{b}}\int^{b} \mathsf{T}_{\vec{x}\oplus\langle y \rangle}(M_1 \to_\beta M'_1)(- \oplus \langle \vec{b} \rangle, b) \times D(\vec{b} \multimap b, -);$$

- *if $M_1M_2 \to_\beta M'_1M_2$ then by induction hypothesis*

$$\begin{aligned}\mathsf{T}_{\vec{x}}(M_1M_2 \to_\beta M'_1M_2) &= \int^{\vec{a},\vec{\Gamma}} \mathsf{T}_{\vec{x}}(M_1 \to_\beta M'_1)(\Gamma_0, \vec{a} \multimap -) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(M_2)(\Gamma_i, a_i) \\ &\quad \times !D^n(-, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i);\end{aligned}$$

- *if $M_1M_2 \to_\beta M_1M'_2$ then by induction hypothesis*

$$\begin{aligned}\mathsf{T}_{\vec{x}}(M_1M_2 \to_\beta M_1M'_2) &= \int^{\vec{a},\vec{\Gamma}} \mathsf{T}_{\vec{x}}(M_1)(\Gamma_0, \vec{a} \multimap -) \times \Pi_{i=1}^{\mathsf{len}(\vec{a})} \mathsf{T}_{\vec{x}}(M_2 \to_\beta M'_2)(\Gamma_i, a_i) \\ &\quad \times !D^n(-, \bigotimes_{i=0}^{\mathsf{len}(\vec{a})} \Gamma_i).\end{aligned}$$

Lemma 10.1.24 (Substitution)**.** *Let $\tilde{\pi} \in \mathsf{T}_{\vec{x}}((\lambda x.M_1)M_2)(\Delta, a)$ if $\tilde{\pi} \to_\beta \tilde{\pi}'$ with $\tilde{\pi}' \in \mathsf{T}_{\vec{x}}(M_1[M_2/x])(\Delta, a)$ then $\mathsf{T}_{\vec{x}}((\lambda x.M_1)M_2 \to_\beta M_1[M_2/x])_{\Delta,a}(\tilde{\pi}) = \tilde{\pi}'$.*

Proof. With respect Definitions 10.1.8 and 10.1.9, by induction on M_1 we observe that the substitution corresponds to $S^{M_1,x,M_2}_{\Delta,a}$. □

Theorem 10.1.25. *Let $\tilde{\pi} \in \mathsf{T}_{\vec{x}}(M)(\Delta, a)$ and $M \to_\beta N$ if $\tilde{\pi} \to_\beta \tilde{\pi}' \in \mathsf{T}_{\vec{x}}(N)(\Delta, a)$ then $\mathsf{T}_{\vec{x}}(M \to_\beta N)_{\Delta,a}(\tilde{\pi}) = \tilde{\pi}'$.*

Proof. By induction on M and using Lemma 10.1.24 as base case. □

Let $\tilde{\pi} \in \mathsf{T}_{\vec{x}}(M)(\Delta, a)$ for some $\langle \Delta, a\rangle \in\, !D^{\mathsf{len}(\vec{x})} \times D$ and $M \in \Lambda$. We say that $\tilde{\pi}$ is *normalisable along* M if there exists $N \in \Lambda$ such that $M \twoheadrightarrow_\beta N$ and $\mathsf{T}_{\vec{x}}(M \twoheadrightarrow_\beta N)_{\Delta,a}(\tilde{\pi})$ is a normal form. We get $\mathsf{T}_{\vec{x}}(M \twoheadrightarrow_\beta N)$ by composition of isomorphisms $\mathsf{T}_{\vec{x}}(M \to_\beta N)$.
The unicity of normal forms for typing derivations along a λ-term M is guaranteed by Theorem 10.1.14.

Definition 10.1.26. *For $M \in \Lambda^o(\vec{x})$, define*

$$\mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)}(\Delta, a) = \{\tilde{\pi} \in \mathrm{NF}(R_\to) \mid \exists N \in \Lambda,\ \ M \twoheadrightarrow_\beta N \textit{ and } \tilde{\pi} \in \mathsf{T}_{\vec{x}}(N)(\Delta, a)\}.$$

The previous construction naturally extends to a distributor that we shall call $\mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)}$.

Notice that, by definition of normalisation, $\tilde{\pi} \in \mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)(\Delta,a)}$ whenever there exists a λ-term N such that $M \twoheadrightarrow_\beta N$ and $\tilde{\pi} \in \mathsf{T}_{\vec{x}}(N)(\Delta, a)$.
For $\tilde{\pi} \in |\mathsf{T}_{\vec{x}}(M)|$ (the web of $\mathsf{T}_{\vec{x}}(M)$, Definition 8.2.2(2)) we denote its normal form as $\mathrm{NF}(\tilde{\pi})_M$. In what follows, we shall keep the parameter M implicit, writing just $\mathrm{NF}(\tilde{\pi})$. In particular,

$$\mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)}(\Delta, a) = \{\mathrm{NF}(\tilde{\pi}) \in R_\to \mid \tilde{\pi} \in \mathsf{T}_{\vec{x}}(M)(\Delta, a)\}. \tag{10.1}$$

Theorem 10.1.27. *For $M \in \Lambda^o(\vec{x})$, there is a canonical natural isomorphism*

$$\mathsf{Norm}_{\vec{x}}(M) : \mathsf{T}_{\vec{x}}(M) \cong \mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)}$$

given by normalisation $\tilde{\pi} \mapsto \mathrm{NF}(\tilde{\pi})$.

Proof. Injectivity and naturality of the map follow from Theorems 8.1.5 and 10.1.14. □

10.2. Reconstructing Approximants

Consider a derivation $\pi \triangleright \Delta \vdash M : a$. We have seen that not all subterms of M need to be typed in a subderivation of π. Thus we might have $\pi \triangleright \Delta \vdash N : a$ also for $\lambda_\perp$-terms $N \neq M$, as untyped subterms of M can be replaced by anything (even $\perp$) without affecting the derivation validity.

We are going to show that every derivation π contains enough information to reconstruct the minimal $\lambda_\perp$-term A_π satisfying $\pi \triangleright \Delta \vdash A_\pi : a$. For any λ-term M, such that $\pi \in \mathsf{T}_{\vec{x}}(M)$, we have $A_\pi \sqsubseteq_\beta M$.

We use the notation $\downarrow M$ for $\{A \mid A \sqsubseteq_\beta M\}$.
In a first time we will consider pairs (π, M) of a derivation and a λ-term, such that $\pi \triangleright \Delta \vdash M : a$ for some environment Δ and type a.

Definition 10.2.1. *For each pair (π, M) where $\pi \in \mathsf{T}_{\vec{x}}(M)$, we define an approximant $A^{\vec{x}}_{\pi,M} \in \ \downarrow M$ by induction on the structure of (π, M) as follows:*

- *if π is an axiom, then $M = x_i$ (for some $x_i \in \vec{x}$), define $A^{\vec{x}}_{\pi,M} = x_i$;*
- *if π is an abstraction, then $M = \lambda y.N$, define $A^{\vec{x}}_{\pi,M} = \lambda y.(A^{\vec{x},y}_{\pi',N})$, where $\pi' \in \mathsf{T}_{\vec{x},\vec{y}}(N)$ is the unique premise of π and we can assume $y \notin \vec{x}$ (wlog, due to α-conversion);*
- *if π is an application, $M = NN'$ and there are premises $\pi_0 \in \mathsf{T}_{\vec{x}}(N)$ and $\pi_1, \ldots, \pi_k \in \mathsf{T}_{\vec{x}}(N')$, for some $k \in \mathbb{N}$. By induction we get $A^{\vec{x}}_{\pi_0,N} \in\downarrow N$ and for $i = 1, \ldots, k$, $A^{\vec{x}}_{\pi_i,N'} \in\downarrow N'$. We can take $\bigvee_{i=1}^{k} A^{\vec{x}}_{\pi_i,N'}$ (since the elements are in $\mathcal{A}_\beta(N')$). We define $A^{\vec{x}}_{\pi} = A^{\vec{x}}_{\pi_0,N}(\bigvee_{i=1}^{k} A^{\vec{x}}_{\pi_i,N'})$.*

Remark 10.2.2. *Note that in the last case when $k = 0$, we have $\bigvee_{i=1}^{k} A^{\vec{x}}_{\pi_i,N'} = \bot$.*

Theorem 10.2.3. *For a given $\pi \in R_{\to}$ corresponding to a λ-term, all (π, M) (such that $\pi \in \mathsf{T}_{\vec{x}}(M)$) have the same $A^{\vec{x}}_{\pi,M}$, we name it $A^{\vec{x}}_{\pi}$.*

Proof. By induction on the last rule of π:

- if π is an axiom, then for any suitable M, $A^{\vec{x}}_{\pi,M} = x_i$, where i is the index of the only type appearing in the type environment of π;
- if π is an abstraction, there is an unique premise $\pi' \in R_{\to}$ and by induction hypothesis there is $A^{\vec{x},y}_{\pi'}$ (we can assume $y \notin \vec{x}$ due to α-conversion). For any suitable M, $A^{\vec{x}}_{\pi,M} = \lambda y.(A^{\vec{x},y}_{\pi'})$;
- if π is an application, then there are premises $\pi_0 \in R_{\to}$ and $\pi_1, \ldots, \pi_k \in R_{\to}$, for some $k \in \mathbb{N}$. By induction hypothesis we get $A^{\vec{x}}_{\pi_0}$ and for $i = 1 \ldots k$, $A^{\vec{x}}_{\pi_i}$. For any suitable M, $M = NN'$ with $A^{\vec{x}}_{\pi_0,N} = A^{\vec{x}}_{\pi_0}$ and for $i = 1 \ldots k$, $A^{\vec{x}}_{\pi_i,N'} = A^{\vec{x}}_{\pi_i}$. In such a case, $A^{\vec{x}}_{\pi,M} = A^{\vec{x}}_{\pi_0}(\bigvee_{i=1}^{k} A^{\vec{x}}_{\pi_i})$. □

Notation 10.2.4. *For a given π, all M such that $\pi \in \mathsf{T}_{\vec{x}}(M)$ have the same $A^{\vec{x}}_{\pi,M}$. Since this approximant only depends on the derivation π, we name it $A^{\vec{x}}_{\pi}$.*

Example 10.2.5.

- *If π is* $$\dfrac{\dfrac{f : a' \to a}{x : \langle\langle\rangle \multimap a'\rangle \vdash x : \langle\rangle \multimap a}}{x : \langle\langle\rangle \multimap a'\rangle \vdash x\Omega : a}$$ *then we have* $\mathrm{tocc}(\pi) = \{\llparenthesis - \rrparenthesis, \llparenthesis - \rrparenthesis\Omega\}$ *and* $A^{x}_{\pi} = x\bot$.
- *Consider the derivation π in Figure 10.1, then we have that:*

  $\mathrm{tocc}(\pi) = \{\llparenthesis - \rrparenthesis, \llparenthesis - \rrparenthesis(yz), x\llparenthesis - \rrparenthesis, x(\llparenthesis - \rrparenthesis z), x(y\llparenthesis - \rrparenthesis)\}$ *and the associated approximant is* $A^{\langle x,y,z\rangle}_{\pi} = x(yz)$ *since* $x(y\bot \vee yz) = x(yz)$.

Remark 10.2.6. *By definition, we have $A^{\vec{x}}_{\pi} = A^{\vec{x}}_{\pi\{\eta\}}$ and $A^{\vec{x}}_{[f]\pi} = A^{\vec{x}}_{\pi}$. Also, $\pi \sim \pi'$ implies $A^{\vec{x}}_{\pi} = A^{\vec{x}}_{\pi'}$. Thus, we can extend $A^{\vec{x}}_{\_}$ to equivalence classes $\tilde{\pi}$ and write $A^{\vec{x}}_{\tilde{\pi}}$ for the corresponding approximant.*

Proposition 10.2.7. *Let $M \in \Lambda^{o}(\vec{x})$ and $\pi \in R_{\to}(M)$:*

(i) $\pi \in R_{\to}(A^{\vec{x}}_{\pi})$ and $A^{\vec{x}}_{\pi} \sqsubseteq_{\beta} M$.

(ii) If π is a normal form then $A^{\vec{x}}_{\pi} \in \mathcal{A}_{\beta}$, whence $A^{\vec{x}}_{\pi} \in \mathcal{A}_{\beta}(M)$.

Proof.

(i) By a straightforward induction on the structure of M.

(ii) By structural induction on M, using the fact that π has no β-redexes. □

We prove a semantic analogue of Ehrhard's theorem in [Ehrhard and Regnier, 2006a] stating that the normal form of the Taylor expansion of a λ-term coincide with the Taylor expansion of its Böhm tree.

Theorem 10.2.8 (Commutation Theorem)**.** *For all $M \in \Lambda^{o}(\vec{x})$,*

$$\mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)} = \mathsf{T}_{\vec{x}}(\mathrm{BT}_{\beta}(M)).$$

Proof.

(⊆) Let $\tilde{\pi} \in \mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)}(\Delta, a)$.

By definition of normalisation along M, there exist $\tilde{\rho} \in \mathsf{T}_{\vec{x}}(M)(\Delta, a)$ and $N \in \Lambda$ such that $\tilde{\pi} = \mathrm{NF}(\tilde{\rho})$ and $\tilde{\pi} \in \mathsf{T}_{\vec{x}}(N)(\Delta, a)$ with $M \twoheadrightarrow_{\beta} N$.

By Proposition 10.2.7, we get $\tilde{\pi} \in \mathsf{T}_{\vec{x}}(A^{\vec{x}}_{\pi})$ and $A^{\vec{x}}_{\pi} \sqsubseteq_{\beta} N$ is a $\beta\bot$-normal form.

Thus we have $A^{\vec{x}}_{\pi} \in \mathcal{A}_{\beta}(N)$, so we conclude $\tilde{\pi} \in \mathsf{T}_{\vec{x}}(\mathrm{BT}_{\beta}(M))(\Delta, a)$.

(⊇) Let $\tilde{\pi} \in \mathrm{BT}_{\beta}(M)(\Delta, a)$.

By definition, there exists a $A \in \mathcal{A}_{\beta}(M)$ such that $\tilde{\pi} \in \mathsf{T}_{\vec{x}}(A)(\Delta, a)$.

By Lemma 10.1.7, such a $\tilde{\pi}$ is a normal form.

From Lemma 9.3.6 and the definition of $\mathcal{A}_{\beta}(M)$, we get $\mathsf{T}_{\vec{x}}(A) \subseteq \mathsf{T}_{\vec{x}}(N)$ for some λ-term N such that $M \twoheadrightarrow_{\beta} N$.

By Theorem 8.1.5, we conclude that there exists $\tilde{\rho} \in \mathsf{T}_{\vec{x}}(M)$ such that $\tilde{\pi}$ is the normal form of $\tilde{\rho}$. □

The following result is a generalisation of the Approximation Theorem for relational graph models [Breuvart et al., 2018] to categorified graph models.

Theorem 10.2.9 (Bicategorical Approximation Theorem)**.**
Let $M \in \Lambda^{o}(\vec{x})$. We have a natural isomorphism

$$\mathsf{appr}_{\vec{x}}(M) : \mathsf{T}_{\vec{x}}(M) \cong \mathsf{T}_{\vec{x}}(\mathrm{BT}_{\beta}(M)).$$

Proof. We compose the isomorphisms obtained in Theorems 10.1.27 and 10.2.8. □

A model is called consistent if it does not equate all λ-terms, and sensible if it is consistent and equates all unsolvables. From the above Approximation Theorem it follows the sensibility of the bicategorical model.

Corollary 10.2.10. *For all $M \in \Lambda^o(\vec{x})$, we have:*

$$\mathrm{BT}_\beta(M) = \bot \iff \mathsf{T}_{\vec{x}}(M) \cong \emptyset_{!D^{\vec{x}},D}.$$

Proof.

($\Rightarrow$) If $\mathrm{BT}_\beta(M) = \bot$, then $\mathcal{A}_\beta(M) = \{\bot\}$. By the Approximation Theorem 10.2.9, we have $\mathsf{T}_{\vec{x}}(M) \cong \mathsf{T}_{\vec{x}}(\{\bot\}) = \emptyset_{!D^{\vec{x}},D}$.

($\Leftarrow$) Assume by contradiction that $\mathrm{BT}_\beta(M) \neq \bot$.

Then, there is $P = \lambda y_1 \ldots y_m.xP_1 \cdots P_k \in \mathcal{A}_\beta(M)$. Suppose that $x \in \mathrm{FV}(M)$, i.e. $x = x_i$ for some $i\,(i \in [n])$, otherwise the argument can be easily adapted. For every type $a = \langle\rangle^k \multimap b$ with $b \in D$, we can construct the derivation π_a as:

$$\dfrac{\dfrac{\dfrac{1_a}{x_1 : \langle\rangle, \ldots, x_i : \langle a\rangle, \ldots, x_n : \langle\rangle, y_1 : \langle\rangle, \ldots, y_m : \langle\rangle \vdash x_i : a}}{x_1 : \langle\rangle, \ldots, x_i : \langle a\rangle, \ldots, x_n : \langle\rangle, y_1 : \langle\rangle, \ldots, y_m : \langle\rangle \vdash x_iP_1 \cdots P_k : b}}{x_1 : \langle\rangle, \ldots, x_i : \langle a\rangle, \ldots, x_n : \langle\rangle \vdash \lambda\vec{y}.x_iP_1 \cdots P_k : \langle\rangle^m \multimap b}$$

By Theorem 10.2.9, we obtain $\tilde{\pi}_a \in \mathsf{T}_{\vec{x}}(M)$. Contradiction. □

11. Characterisation of the Theory

The equational theories of λ-calculus: λ-theories are defined as congruences on λ-terms containing β-conversions (and α ones). They are key objects for the study of the equivalence between terms rather than the study of their computational process.

The λ-theories can come from both the semantic and the syntactic world:

- Semantically: from a denotational model $\mathcal{D}$ of λ-calculus we can define λ-theories by taking the kernel $\mathrm{Th}(\mathcal{D})$ of its interpretation function:

$$\mathrm{Th}(\mathcal{D}) = \{(M, N) \mid [\![M]\!]_{\mathcal{D}} = [\![N]\!]_{\mathcal{D}}\}.$$

- Syntactically: we defined them by equating λ-terms with a same given operational behaviour.

  Among them, well known theories are:

  - $\mathcal{H}$ where all unsolvable λ-terms are collapsed together;
  - $\mathcal{B}$ equating two λ-terms exactly when they have the same Böhm tree;
  - extensional theory $\mathcal{H}^*$ equating all observationally indistinguishable λ-terms.

Some classical results are that Plotkin's model $\mathcal{P}_\omega$ has theory $\mathrm{Th}(\mathcal{P}_\omega) = \mathcal{B}$ and Scott's $\mathcal{D}_\infty$ has theory $\mathrm{Th}(\mathcal{D}_\infty) = \mathcal{H}^*$ ([Hyland, 1976, Wadsworth, 1976]).
In both cases the inclusion $\mathcal{B} \subseteq \mathrm{Th}(\mathcal{D})$ follows from the fact that $\mathcal{D}$ satisfies an Approximation Theorem stating that the interpretation of a λ-term in the model $\mathcal{D}$ is given by the supremum of the denotations of its finite approximants.

In 1-categorical semantics, the fact that a model $\mathcal{D}$ satisfies the Approximation Theorem just allows to conclude that $\mathcal{B} \subseteq \mathrm{Th}(\mathcal{D})$.
For instance, since the relational interpretation of a λ-term M is given by the set of its typings (Γ, a), and many derivations π of $\Gamma \vdash M : a$ may exist, one cannot univocally reconstruct an approximant $A_\pi \in \mathcal{A}_\beta(M)$ as we have done in the 2-dimensional case in Section 10. Indeed since the interpretation of a λ-term in this settings is given by the "collection" of all its type derivations (see Chapter 9) it contains much more information.

In the present chapter we will characterise the λ-theory of the bicategorical model introduced and studied in previous chapters. In Section 11.2, we show that the additional information of type derivations is sufficient to obtain the characterisation of the λ-theory associated with our model as an easy corollary (Corollary 11.2.5) of the Bicategorical

Approximation Theorem (see Theorem 10.2.9).

We demonstrate that any derivation π living in the interpretation of a λ-term M, but not in the interpretation of a λ-term N, implies that A_π (obtained with the map in Definition 10.2.3) is an approximant of M but not of N. This leads to a characterisation of the theory of $\mathcal{D}$, since it allows to conclude in Theorem 11.2.4 that $\mathrm{Th}(\mathcal{D}) = \mathcal{B}$.

This technique to characterise the theory of a model is original, and the same reasoning cannot be performed in the relational semantics as typings do not carry enough information to uniquely identify an approximant, in general. As previously the results have been published in [Kerinec et al., 2023].

11.1. Some Formal Definitions

Definition 11.1.1. *A λ-theory is any congruence on Λ (that is, an equivalence relation compatible with abstraction and application) containing the β-conversion.*

A λ-theory is called:

- *extensional* if it contains the η-conversion as well;
- *consistent* if it does not equate all λ-terms;
- *sensible* if it does equate all unsolvables.

Notation 11.1.2. *Given a λ-theory $\mathcal{T}$, and $M, N \in \Lambda$ we will write $\mathcal{T} \vdash M = N$, or simply $M =_{\mathcal{T}} N$, to express the fact that M and N are equal in $\mathcal{T}$.*

Definition 11.1.3. *The equivalence $\mathcal{B}$ is obtained by equating all λ-terms having the same Böhm tree:*

$$\mathcal{B} = \{(M, N) \mid \mathrm{BT}_\beta(M) = \mathrm{BT}_\beta(N)\} \subseteq \Lambda^2.$$

Example 11.1.4.

- *$\mathcal{B} \vdash \Omega = M$, for all M unsolvable;*
- *$\mathcal{B} \vdash \lambda x.x\Omega = \lambda x.x(\mathbf{YI})$, since $\mathbf{YI}$ is unsolvable;*
- *$\mathcal{B} \vdash \mathbf{Y} = Z$, for any fixed point combinator Z.*

The next proposition follows directly:

Property 11.1.5. *The λ-theory $\mathcal{B}$ is consistent and sensible.*

Proof. The λ-theory $\mathcal{B}$ equates all unsolvable λ-terms (by definition of Böhm trees), but not all the λ-terms. □

11.2. The Induced Theory

In order to define the theory of a model, we focus on isomorphisms that "behave well" w.r.t. $\to_\beta$.

We say that a natural isomorphism $\gamma : [\![M]\!]_{\vec{x}} \cong [\![N]\!]_{\vec{x}}$ is *coherent w.r.t. β-normalisation* when the induced natural isomorphism $\gamma : \mathsf{T}_{\vec{x}}(M) \cong \mathsf{T}_{\vec{x}}(N)$ satisfies the following property:

$$\textit{for all } \tilde{\pi} \in \mathsf{T}_{\vec{x}}(M)(\Delta, a) \textit{ we have } \mathrm{NF}(\tilde{\pi}) = \mathrm{NF}(\gamma_{\Delta,a}(\tilde{\pi})).$$

Definition 11.2.1. *The* non-extensional theory of a bicategorical model $\mathcal{D}$ in CatSym *is defined by*

$$\mathrm{Th}(\mathcal{D}) = \{(M,N) \mid M,N \in \Lambda^o(\vec{x}) \textit{ and } \gamma : [\![M]\!]_{\vec{x}} \cong [\![N]\!]_{\vec{x}} \textit{ with } \gamma \in \mathsf{ISO}^\beta_{M,N}\},$$

where $\mathsf{ISO}^\beta_{M,N}$ *is the set of natural isomorphisms* $[\![M]\!]_{\vec{x}} \cong [\![N]\!]_{\vec{x}}$ *coherent with respect to β-normalisation.*

It is readily proved that $\mathrm{Th}(\mathcal{D})$ is a λ-theory. We now show that all categorified graph models have the same non-extensional theory, namely $\mathcal{B}$.

Remark 11.2.2. *Note that the definition of theory induced by a model depends on an appropriate choice of isomorphisms. This was not the case for the analogous notion in the 1-categorical setting, since the only possible choice of isomorphisms in that case is the equality.*

Lemma 11.2.3. *Let* $M \in \Lambda^o(\vec{x})$ *and* $A \in \mathcal{A}_\beta(M)$*. If* $A \neq \bot$ *then there exists* $\tilde{\pi}$ *belonging to the web* $|\mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)}|$ *(see Definition 8.2.22 and Equation* (10.1)*) such that* $A = A^{\vec{x}}_{\tilde{\pi}}$*.*

Proof. By structural induction on A.
Since $A \neq \bot$, we must have $A = \lambda y_1 \dots y_m.xA_1 \cdots A_k$ such that for $i = 1,\dots,k$, $\mathrm{FV}(A_i) \subseteq \vec{x},\vec{y}$ for $\vec{y} = \{y_1,\dots,y_m\}$. Without loss of generality, assume $\mathsf{len}(\vec{x}) > 0$ and $x = x_1 \in \vec{x}$.
By definition, $M \twoheadrightarrow_\beta \lambda y_1 \dots y_m.x_1M_1 \cdots M_k$ with $A_j \in \mathcal{A}_\beta(M_j)$, for $j = 1,\dots,k$.
By induction hypothesis, for all such j, $A_j \neq \bot$ entails the existence of a derivation $\tilde{\pi}_j \in \mathrm{NF}_{\mathsf{T}_{\vec{x},\vec{y}}(M_j)}(\Delta_j, a_j)$ such that $A_j = A^{\vec{x},\vec{y}}_{\tilde{\pi}_j}$.

We define

$$\mu_j = \begin{cases} \langle\rangle & \text{if } A_j = \bot, \\ \langle a_j \rangle & \text{otherwise,} \end{cases} \qquad\qquad \Gamma_j = \begin{cases} \langle\langle\rangle,\dots,\langle\rangle\rangle & \text{if } A_j = \bot, \\ \Delta_j & \text{otherwise,} \end{cases}$$

and $b = \mu_1 \multimap \cdots \multimap \mu_k \multimap c$ with $c \in D$.
From the $\tilde{\pi}_j$s, using k times the rule (app), it is easy to construct

$$\tilde{\pi}' \in \mathrm{NF}_{\mathsf{T}_{\vec{x},\vec{y}}(x_1M_1\cdots M_k)}(\langle\langle b\rangle, \underbrace{\langle\rangle,\dots,\langle\rangle}_{\mathsf{len}(\vec{x})+m}\rangle \otimes \big(\textstyle\bigotimes_{j=1}^k \Gamma_j\big), c)$$

and therefore, by applying m-times the rule (abs) we get the derivation $\tilde{\pi}$ we are looking for, by construction $\tilde{\pi} \in |\mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)}|$ and $A^{\vec{x}}_{\pi} = A$. □

Theorem 11.2.4. $\mathsf{T}_{\vec{x}}(M) \cong \mathsf{T}_{\vec{x}}(N) \iff \mathrm{BT}_\beta(M) = \mathrm{BT}_\beta(N)$.

Proof.

(⇒) Assume $\mathsf{T}_{\vec{x}}(M) \cong \mathsf{T}_{\vec{x}}(N)$. By definition, this entails $\mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)} = \mathrm{NF}_{\mathsf{T}_{\vec{x}}(N)}$. Assume $\mathrm{BT}_\beta(M) \neq \mathrm{BT}_\beta(N)$ towards a contradiction.

Say, there is $A \in \mathcal{A}_\beta(M) \setminus \mathcal{A}_\beta(N)$.

By Lemma 11.2.3 there is $\tilde{\pi} \in |\mathrm{NF}_{\mathsf{T}_{\vec{x}}(M)}| = |\mathrm{NF}_{\mathsf{T}_{\vec{x}}(N)}|$ such that $A^{\vec{x}}_{\tilde{\pi}} = P$. By definition of normalisation along N, we have $\tilde{\pi} \in |\mathsf{T}_{\vec{x}}(N')|$ for some N' such that $N \twoheadrightarrow_\beta N'$. By Proposition 10.2.7 we obtain $P \sqsubseteq_\beta N'$, from which it follows $P \in \mathcal{A}_\beta(N)$. Contradiction.

(⇐) Assume $\mathrm{BT}_\beta(M) = \mathrm{BT}_\beta(N)$. Then

$$\begin{aligned} \mathsf{T}_{\vec{x}}(M) &\cong \mathsf{T}_{\vec{x}}(\mathrm{BT}_\beta(M)) && \text{by Theorem 10.2.9,} \\ &= \mathsf{T}_{\vec{x}}(\mathrm{BT}_\beta(N)) && \text{by the assumption,} \\ &\cong \mathsf{T}_{\vec{x}}(N) && \text{by Theorem 10.2.9.} \end{aligned}$$

This concludes the proof. □

Corollary 11.2.5. $\mathrm{Th}(\mathcal{D}) = \mathcal{B}$.

Remark 11.2.6. *The reader could be surprised by the* prima facie *paradoxical result of Corollary 11.2.5. Our result works for arbitrary categorified graph models, while it is well-known that in the 1-dimensional case no extensional model can have theory $\mathcal{B}$, since $\mathcal{B}$ is not an extensional theory. However, the 2-dimensional aspect of our semantics considerably refines the situation. At the beginning of the section we restricted our attention to isomorphisms preserving β-normalisation of type derivations. It is easy to see that, if D is extensional, the canonical natural isomorphism*

$$[\![M \to_\eta N]\!]_{\vec{x}} \colon [\![M]\!]_{\vec{x}} \cong [\![N]\!]_{\vec{x}}$$

does not preserve β-normalisation of type derivations. Indeed, take $D = D^+$, $M = \lambda x.yx$ and $N = y$. Now, consider

$$\pi = \cfrac{\cfrac{\cfrac{\mathsf{e}^{-1}_{\langle*\rangle\multimap*} : * \to (\langle*\rangle \multimap *)}{y : \langle*\rangle \vdash y : \langle*\rangle \multimap *} \qquad x : \langle*\rangle \vdash x : *}{y : \langle*\rangle, x : \langle*\rangle \vdash yx : *} \qquad \mathsf{e}_{\langle*\rangle\multimap*} : (\langle*\rangle \multimap *) \to *}{y : \langle*\rangle \vdash \lambda x.yx : *}$$

and $$\pi' = \cfrac{}{y : \langle*\rangle \vdash y : *}$$

We have that $\mathsf{T}_{\vec{x}}(M \to_\eta N)(\langle*\rangle, *)(\tilde{\pi}) = \tilde{\pi}'$. Clearly, there is no β-reduction chain that produces $\mathrm{NF}(\pi) = \mathrm{NF}(\pi')$ so, by λ-abstracting y on both sides, we get that the model distinguishes $[\![\mathbf{1}]\!]$ and $[\![\mathbf{I}]\!]$. In fact, our choice of isomorphisms automatically discards the isomorphisms induced by extensionality.

12. Decategorification of the Semantics

In this chapter we translate in 1-dimension our previous results. From the perspective of enriched category theory, we perform a *change of base* [Kelly, 1982].

We start, in Section 12.1, by defining the target category Polr of preorders and monotonic relations [Ehrhard, 2012, Ehrhard, 2016]. All the constructions in Polr are generalisations of the ones for Rel, notice that Rel is a full subcategory of Polr, considering sets as discrete preorders.

In Section 12.2, we define a decategorification pseudofunctor forgetting the bicategorical structure which is present in the model $\mathcal{D}$ and retrieving a *relational graph model* $\mathcal{U}$ living in the coKleisli of the comonad of finite multisets on the category Polr. We provide a type-theoretic description of the relational graph models living in Polr (Definition 12.1.5).

We then prove that the Approximation Theorem, for those relational graph models arising from the decategorification, follows directly from its bicategorical analogue (corollary 12.2.7).

We conclude that the theory of the categorified graph model is included in the theory of its decategorification (corollary 12.2.8).

12.1. The Category Polr

We shall work within the category Polr of preorders and monotonic relations, already studied in [Ehrhard, 2012, Ehrhard, 2016].

Definition 12.1.1. Polr *category of preorders and monotonic relations:*

- *objects of* Polr *are preorders;*
- *a morphism $f : \mathcal{X} \to \mathcal{Y}$ from $\mathcal{X} = \langle |\mathcal{X}|, \leq_{\mathcal{X}} \rangle$ to $\mathcal{Y} = \langle |\mathcal{Y}|, \leq_{\mathcal{Y}} \rangle$ is a* monotonic *relation from $|\mathcal{X}|$ to $|\mathcal{Y}|$, namely a relation $f \subseteq |\mathcal{X}| \times |\mathcal{Y}|$ such that $\langle x, y \rangle \in f$ entails $\langle x', y' \rangle \in f$, for all $x' \leq_{\mathcal{X}} x$ and $y \leq_{\mathcal{Y}} y'$;*
- *composition is given by relational composition;*
- *in* Polr *the product $\mathcal{X}_1 \,\&\, \mathcal{X}_2$ is the disjoint union of sets $|\mathcal{X}_1| \sqcup |\mathcal{X}_2|$ with the preorder $\leq_{\mathcal{X}_1} \sqcup \leq_{\mathcal{X}_2}$ defined as $\langle i, x \rangle \leq_{\mathcal{X}_1 \& \mathcal{X}_2} \langle j, y \rangle$ whenever $i = j$ and $x \leq_{\mathcal{X}_i} y$;*
- *the terminal object is $\emptyset$ with the empty order.*

  Projections $\pi_i \colon \mathcal{X}_1 \,\&\, \mathcal{X}_2 \to \mathcal{X}_i$ are $\pi_i = \{\langle \langle i, x \rangle, x' \rangle \mid x \leq_{\mathcal{X}_i} x'\}$;

- Polr *has a symmetric monoidal structure. The tensor $\mathcal{X}_1 \otimes \mathcal{X}_2$ is the cartesian product of sets with the product preorder. The endofunctor $\mathcal{X} \otimes (-)$ admits a right adjoint $(-) \multimap \mathcal{Y}$ defined as follows: $|\mathcal{X} \multimap \mathcal{Y}| = |\mathcal{X}| \times |\mathcal{Y}|$ and $\langle x, y\rangle \leq_{\mathcal{X}\multimap\mathcal{Y}} \langle x', y'\rangle$ if $x' \leq_{\mathcal{X}} x$ and $y \leq_{\mathcal{Y}} y'$.*

The following remark is crucial to properly establish the decategorification.

Remark 12.1.2. *The definition above could be equivalently rephrased by taking the* characteristic function *point of view: considering a monotonic relation $R : \mathcal{X} \to \mathcal{Y}$ as a monotonic function $f_R : \mathcal{X}^{\mathrm{op}} \times \mathcal{Y} \to \{0, 1\}$.*

The category Polr extends naturally to a bicategory by considering inclusions $f \subseteq g$ as 2-cells.

12.1.1. Exponential Modality

The exponential modality of Linear Logic is interpreted by exploiting the free commutative monoid construction over a set. What happens here is again a direct generalisation of the well-known relational case, where the multiset construction is considered.

Notation 12.1.3.

1. *We denote by $\mathcal{M}_{\mathrm{f}}(X)$ the* free commutative monoid *of finite multisets over a set X. A finite multiset $\overline{a}$ over X is denoted as an unordered list $[a_1, \ldots, a_k]$, possibly with repetitions.*

2. *Given finite multisets $\overline{a} = [a_1, \ldots, a_k], \overline{b} = [b_1, \ldots, b_n] \in \mathcal{M}_{\mathrm{f}}(X)$, their union is $\overline{a} + \overline{b} = [a_1, \ldots, a_k, b_1, \ldots, b_n]$.*

We now detail the action on objects of the comonadic endofunctor that gives the interpretation of the ! modality.

Definition 12.1.4.

(i) The endofunctor $!\colon \mathrm{Polr} \to \mathrm{Polr}$ *is given by $!\mathcal{X} = \langle \mathcal{M}_{\mathrm{f}}(|\mathcal{X}|), \leq_{\mathcal{X}}\rangle$, where*

$$[x_1, \ldots, x_n] \leq_{!\mathcal{X}} [x'_1, \ldots, x'_{n'}]$$

holds if and only if $n = n'$ and there exists $\sigma \in \mathfrak{S}_n$ such that $x_i \leq_{\mathcal{X}} x'_{\sigma(i)}$, for all $i \in [n]$.

(ii) We denote by MPolr *the Kleisli category of the comonad defined in (i) (see Definition 7.1.22).*

It is worth noting that the construction above strongly recalls the one considered in Section 8.2. Such a construction can indeed be seen as the decategorification of the free monoidal completion, as we will detail in the next subsection.

12.1.2. Relational Graph Models and Their Type-Theoretic Presentation

In this section we extend the concept of *relational graph model* introduced in [Breuvart et al., 2018] to the preordered setting.

Definition 12.1.5 (Relational graph pre-models)**.** *A* relational graph pre-model *consists of a preorder $\mathcal{U}$ equipped with a monotonic injection $\iota : !\mathcal{U}^{\mathrm{op}} \times \mathcal{U} \hookrightarrow \mathcal{U}$.*

It is easy to see that a relational graph pre-model canonically induces a reflexive object in MPolr (Definition 7.1.18). We call this object a *relational graph model.*

Notation 12.1.6. *A relational graph model $\mathcal{U}$ can be presented as a non-idempotent intersection type system $\mathcal{R}_{\leq}$ (see Fig. 12.1), depending on a preordered set $\mathcal{X} = \langle |\mathcal{X}|, \leq_{\mathcal{X}} \rangle$ of* atoms *(ground types). The* types over $|\mathcal{X}|$ *correspond to the elements of $|\mathcal{U}|$. We let $\overline{a} \multimap a$ be another notation for $\iota(\langle \overline{a}, a \rangle)$.*

- *In this context the "non-idempotent intersection" is assumed to be commutative, therefore it is represented by finite multisets rather than ordered lists. The preorder $\leq_{\mathcal{U}}$ associated with $\mathcal{U}$ is obtained by lifting $\leq_{\mathcal{X}}$ from atoms to multisets and to higher types as expected.*
- *The elements of $!\mathcal{U}^n$ are called* (type) environments *(of length n) and are denoted by Γ, Δ.*
- *The* tensor product *of two type environments is defined by applying multiset union, denoted by $+$, componentwise: $\langle \overline{a}_1, \dots, \overline{a}_n \rangle \otimes \langle \overline{b}_1, \dots, \overline{b}_n \rangle = \langle \overline{a}_1 + \overline{b}_1, \dots, \overline{a}_n + \overline{b}_n \rangle$.*
- *We write $\vdash_{\mathrm{MPolr}}$ to denote judgements in the associated type assignment system $\mathcal{R}_{\leq}$ (Figure 12.1b).*

Remark 12.1.7. *Figure 12.1a actually describes a family of reflexive objects $\mathcal{U}_{\mathcal{X}}$ in* MPolr*, since $\mathcal{U}$ is parametric over a preordered set $\mathcal{X}$. This construction has been already explicitly considered in the context of bang calculus [Guerrieri and Olimpieri, 2021]. It derives from the type theoretic presentation of the Scott semantics in [Ehrhard, 2012].*

The underlying set of $\mathcal{U}$ is populated by non-idempotent intersection types over the set $|\mathcal{X}|$ of atoms. As in the categorified setting of distributors, $\mathcal{U}$ is given by a free algebra construction, which determines an inclusion of preorders $!\mathcal{U}^{\mathrm{op}} \times \mathcal{U} \subseteq \mathcal{U}$.

Definition 12.1.8.

- *The interpretation of a λ-term $M \in \Lambda^o(x_1, \dots, x_n)$ in a relational graph model $\mathcal{U}$ living in* MPolr *is given by a monotonic relation*

$$\llbracket M \rrbracket^{\mathrm{MPolr}}_{\vec{x}} : !\mathcal{U}^n \to \mathcal{U},$$
$$\llbracket M \rrbracket^{\mathrm{MPolr}}_{\vec{x}}(\Delta, a) = \begin{cases} 1 & \textit{if } \Delta \vdash_{\mathrm{MPolr}} M : a, \\ 0 & \textit{otherwise.} \end{cases}$$

We also write $(\Delta, a) \in \llbracket M \rrbracket^{\mathrm{MPolr}}_{\vec{x}}$ for $\llbracket M \rrbracket^{\mathrm{MPolr}}_{\vec{x}}(\Delta, a) = 1$.

- *The interpretation above is extended to approximants $A \in \mathcal{A}_\beta$ in the obvious way, and to Böhm trees by setting:*

$$(\Delta, a) \in [\![\mathrm{BT}_\beta(M)]\!]^{\mathrm{MPolr}}_{\vec{x}} \iff \exists A \in \mathcal{A}_\beta(M)\,,\ \ (\Delta, a) \in [\![A]\!]^{\mathrm{MPolr}}_{\vec{x}}.$$

We remark that, if $\mathcal{X}$ is discretely ordered by $=$, then the construction boils down to the standard one performed in the context of relational semantics [de Carvalho, 2007].

Types (for $x \in |\mathcal{X}|$):

$$\mathtt{Ty}_{\mathcal{X}} \ni a := x \mid [a_1, \dots, a_k] \multimap a$$

Free construction of $\leq_{\mathcal{U}}$ depending on $\mathcal{X}$:

$$\frac{x \leq_{\mathcal{X}} x'}{x \leq_{\mathcal{U}} x'} \qquad\qquad \frac{\overline{a}' \leq_{\mathcal{U}} \overline{a} \qquad a \leq_{\mathcal{U}} a'}{(\overline{a} \multimap a) \leq_{\mathcal{U}} (\overline{a}' \multimap a')}$$

$$\frac{\sigma \in \mathfrak{S}_k \quad a_1 \leq_{\mathcal{U}} a'_{\sigma(1)} \quad \overset{k \in \mathbb{N}}{\dots} \quad a_k \leq_{\mathcal{U}} a'_{\sigma(k)}}{[a_1, \dots, a_k] \leq_{\mathcal{U}} [a'_1, \dots, a'_k]}$$

(a) Graph of intersection types $G_{\mathcal{X}}$.

Derivation rules:

$$\frac{a' \leq_{\mathcal{U}} a}{x_1 : [\,], \dots, x_i : [a'], \dots x_n : [\,] \vdash x_i : a}$$

$$\frac{\Gamma_0 \vdash M : [a_1, \dots, a_k] \multimap a \quad (\Gamma_i \vdash N : a_i)_{i \in [k]} \quad \Delta \leq_{\mathcal{U}^n} \bigotimes_{j=0}^{k} \Gamma_j}{\Delta \vdash MN : a}$$

$$\frac{\Delta, x : \overline{a} \vdash M : a \qquad \overline{a} \multimap a \leq_{\mathcal{U}} b}{\Delta \vdash \lambda x.M : b}$$

(b) Non-idempotent intersection type system $\mathcal{R}_{\leq}$.

Figure 12.1.: Type theoretic presentation of a relational graph model living in Polr.

12.2. Decategorification Pseudofunctor

We want to define a pseudofunctor $\mathsf{Dec} : \mathrm{Dist} \to \mathrm{Polr}$. We now take the angle of characteristic function on monotonic relations. The construction that we shall present corresponds to a *change of base* in the sense of enriched category theory [Laird, 2017, Galal, 2020], passing from Set-enriched distributors to $\{0, 1\}$-enriched distributors.

Definition 12.2.1.

- *The* preorder collapse $\mathsf{Dec}(A)$ of a small category A *is defined by setting* $|\mathsf{Dec}(A)| = \mathrm{ob}(A)$ *and* $a \leq_{\mathsf{Dec}(A)} b$ *whenever* $A(a, b) \neq \emptyset$.
- *Given small categories A and B, define a functor*

$$\mathsf{Dec}_{A,B} : \mathrm{Dist}(A, B) \to \mathrm{Polr}(\mathsf{Dec}(A), \mathsf{Dec}(B))$$

by setting, for all $F : A \nrightarrow B$,

$$\mathsf{Dec}_{A,B}(F) = \{\langle a, b\rangle \mid \langle a, b\rangle \in |\mathsf{Dec}(A)^{\mathrm{op}} \times \mathsf{Dec}(B)| \wedge F(a, b) \neq \emptyset\}.$$

The data above naturally define a pseudofunctor $\mathsf{Dec} : \mathrm{Dist} \to \mathrm{Polr}$, called the *decategorification of* Dist *to* Polr, which also preserves the linear logic structure [Galal, 2020].

Proposition 12.2.2. *Let $A \in \mathrm{Cat}$. We have an equivalence of categories:* $\mathsf{D}_! : \mathsf{Dec}(!A) \simeq !(\mathsf{Dec}(A))$ *given by the map* $\langle a_1, \dots, a_k\rangle \mapsto [a_1, \dots, a_k]$.

Proof. $\mathsf{D}_!$ is a fully faithful functor (functors between preorders are just monotonic functions) by remarking that $\langle a_1, \dots, a_k\rangle \leq_{!A} \langle a'_1, \dots, a'_k\rangle$ whenever there exists $\sigma \in \mathfrak{S}_k$ and $a_i \leq_A a'_{\sigma(i)}$. Hence $\vec{a} \leq \vec{a}'$ iff $\mathsf{D}_!(\vec{a}) \leq \mathsf{D}_!(\vec{a}')$. It is also surjective on objects: by taking an arbitrary presentation of a multiset $[a_1, \dots, a_k]$, *e.g.* $\langle a_1, \dots, a_k\rangle$ we trivially have $\mathsf{D}_!(\langle a_1, \dots, a_k\rangle) = [a_1, \dots, a_k]$. Hence the equivalence class of $\langle a_1, \dots, a_k\rangle$ is sent to the multiset $[\widetilde{a_1}, \dots, \widetilde{a_k}]$. □

We work modulo the equivalence above, so we identify $\mathsf{Dec}(!A)$ with the multiset construction over $\mathsf{Dec}(A)$. Remark that this equivalence extends to $!D^n$, with $n \in \mathbb{N}$, in the natural way.

We show that the decategorification of the free category of intersection types D_A (as described in section 9.1) is exactly the free preorder on intersection types $\mathcal{U}_{\mathsf{Dec}(A)}$.

Lemma 12.2.3. *Let D_A be a categorified graph model. Then $\mathsf{Dec}(D_A) = \mathcal{U}_{\mathsf{Dec}(A)}$ is a relational graph model living in* MPolr.

Proof. By exploiting the fact that the decategorification pseudofunctor preserve the linear logic structure. By applying the former proposition, it is easy to see that

$$\mathsf{Dec}(D_A) = (!\mathsf{Dec}(D_A)^{\mathrm{op}} \times \mathsf{Dec}(D_A)) \sqcup \mathsf{Dec}(A)$$

thus, $\mathsf{Dec}(D_A)$ is the free algebra construction. □

Remark 12.2.4. *Note that, if A is a set, we recover the standard construction of non-extensional models used in the relational setting [de Carvalho, 2007]. The decategorifications of D^* and D^+ correspond to two extensional models in* MRel, *studied in [Breuvart et al., 2018], which can be seen as a relational counterpart of classical filter models of λ-calculus.*

Lemma 12.2.5. *Let $M \in \Lambda_\perp$.*

- *If $\Delta \vdash_{\text{CatSym}} M : a$ then $\mathsf{D}_!(\Delta) \vdash_{\text{MPolr}} M : \mathsf{D}_!(a)$.*
- *Considering $\Delta \vdash_{\text{CatSym}} M : a$, $\eta : \Delta' \to \Delta$ and $f : a \to a'$. We have $\mathsf{D}_!(\Delta') \vdash_{\text{MPolr}} M : \mathsf{D}_!(a')$.*

Proof. Both items follow easily by induction on a derivation of $\Delta \vdash_{\text{CatSym}} M : a$. □

Theorem 12.2.6. *Let $M \in \Lambda_\perp$, we have $\mathsf{Dec}(\mathsf{T}_{\vec{x}}(M)) = [\![M]\!]_{\vec{x}}^{\text{MPolr}}$.*

Proof. By direct application of Lemma 12.2.5. □

We show that the Approximation Theorem for $\mathcal{U}_{\mathsf{Dec}(A)}$ is a direct consequence of the result above and of the Bicategorical Approximation Theorem 10.2.9.

Corollary 12.2.7 (Approximation Theorem for MPolr)**.**
For all $M \in \Lambda^o(\vec{x})$, we have $[\![M]\!]_{\vec{x}}^{\text{MPolr}} = [\![\mathrm{BT}_\beta(M)]\!]_{\vec{x}}^{\text{MPolr}}$, i.e.

$$(\Delta, a) \in [\![M]\!]_{\vec{x}}^{\text{MPolr}} \iff \exists P \in \mathcal{A}_\beta(M), (\Delta, a) \in [\![P]\!]_{\vec{x}}^{\text{MPolr}}.$$

Proof. Corollary of Theorem 10.2.9 and Theorem 12.2.6. The central point of the proof is the remark that, by Proposition 12.2.2, $(\Delta, a) = \mathsf{D}_!(\Delta', a')$ for some context and type of the system $R_\to$. Then, one derives that $(\Delta, a) \in [\![M]\!]_{\vec{x}}^{\text{MPolr}}$ if and only if $\mathsf{T}_{\vec{x}}(M)(\Delta', a') \neq \emptyset$. We can therefore conclude by applying the bicategorical Approximation Theorem. □

Note that the theory of the reflexive object $\mathsf{Dec}(D)$, for D categorified graph model, is the standard 1-categorical notion defined as:

$$\mathrm{Th}(\mathsf{Dec}(D)) = \{(M, N) \mid [\![M]\!]_{\vec{x}}^{\text{MPolr}} = [\![N]\!]_{\vec{x}}^{\text{MPolr}}\}.$$

Corollary 12.2.8. *For all $M, N \in \Lambda^o(\vec{x})$, we have*

$$\mathsf{T}_{\vec{x}}(M) \cong \mathsf{T}_{\vec{x}}(N) \Rightarrow [\![M]\!]_{\vec{x}}^{\text{MPolr}} = [\![N]\!]_{\vec{x}}^{\text{MPolr}}.$$

Therefore

$$\mathcal{B} = \mathrm{Th}(D) \subseteq \mathrm{Th}(\mathsf{Dec}(D)).$$

13. Conclusion

In this second part, we left the realm of Call-by-Value to explore that of classic Call-by-Name. We introduced a type-theoretic bicategorical semantics of λ-calculus, building upon the work presented in [Olimpieri, 2021, Olimpieri, 2020], and extending it. In the aforementionned articles Olimpieri considered a bicategorical semantics of λ-calculus, where the models are free-algebra constructions for an appropriate endofunctor, while we introduced the more general class of categorified graph models (in Section 9.1). The free-algebra models are then just particular (non-extensional) instances of our construction. Those categorified graphs models are a generalisation of relational graph models from [Manzonetto and Ruoppolo, 2014] and graph models from [Engeler, 1981]. Our models live in a cartesian closed bicategory of distributors: the bicategory of symmetric categorical sequences ([Gambino and Joyal, 2017]). Following [Olimpieri, 2021], we can present them as intersection type systems, where the intersection is neither commutative nor idempotent. However commutativity is restored with permutative actions on type derivations.

We proved that the classical interpretation of a λ-term M in such a model can be seen as an intersection type distributor $\mathsf{T}_{\vec{x}}(M)$ that actually corresponds to the collection of its type derivations (see Theorem 9.3.5). We can consider the semantic as proof-relevant since the interpretation of a λ-term does not just answer the question "is this term M typable with type a in environment Γ?", but contains the derivations of $\Gamma \vdash M : a$ as witness of the statement. We then explored the importance of this additionnal information.

We observed that not all subterms of a term M are typed in a derivation π of M. Thus only some redexes are typed: they are the ones carrying information. Contracting one of those redexes leads to a term M' such that $M \to_\beta M'$ and a derivation π' of M' with strictly small size than π. We can then define a normal form of derivations in Definition 10.1.26. Due to the nature of the intersection type distributor we also define its normal form. For any λ-term M, there is an isomorphism between $\mathsf{T}_{\vec{x}}(M)$ and $\mathrm{NF}(\mathsf{T}_{\vec{x}}(M))$ (see Theorem 10.1.27).

In Section 9.3.1, we extend the interpretation (and the intersection type distributor) to approximants and Böhm trees. The untyped subterms in a derivation could as well be replaced by $\perp$ without altering the validity of it. Therefore we defined a map linking derivations to minimal compatible λ-terms extended with $\perp$ (see Definition 10.2.3). In the case of derivations in normal forms, those are approximants. Using them we proved the commutation Theorem 10.2.8: the normal form of the interpretation of a λ-term is equivalent to interpretation of its Böhm tree. This result is similar to the famous one connecting Taylor expansion and Böhm trees: $\mathrm{NF}(\mathscr{T}(M)) = \mathscr{T}(\mathrm{BT}_\beta(M))$. This highlights that our intersection type derivations can be seen as linear appoximants.

From the commutation theorem we deduced the approximation Theorem 10.2.9: the interpretation of the term is isomorphic to that of its Böhm tree.

However the interpretation being proof-relevant provides us with even more information. Indeed in the proof of Theorem 11.2.4, from a derivation π contained in the interpretation of a λ-term M but not in the interpretation of another λ-term N, we obtained an approximant A_π for M that is not an approximant of N. The theory induced by our model is then the one that equates λ-terms with same Böhm trees: $\mathrm{Th}(\mathcal{D}) = \mathcal{B}$ (see Corollary 11.2.5).

Similar techniques could not have been possible in usual models where the interpretation does not contain as much information. In the last chapter we decategorified our results and observed the consequences in the traditional 1-dimensionnal models.

The models considered are relational graph models in the coKleisli category of the comonad of finite multisets on the category Polr (the category of preorders and monotonic relations (Definition 12.1.5)).

Those models can be presented as non-idempotent intersection type systems.

We introduced a pseudofunctor going from Set-enriched distributors to $\{0,1\}$-enriched distributors, called the decategorification of Dist to Polr in Definition 12.2.1. It preserves the linear logic structure (see [Galal, 2020]). From the bicategorical approximation theorem we then deduced, in corollary 12.2.7, an approximation theorem for the relational graph models. In corollary 12.2.8), we finally showed that the theory of the categorified graph model is included in the theory of its decategorification.

13.0.1. Future Developments.

We conclude with some more speculative discussions about possible future developments.

Towards 2-Dimensional λ-Theories

In Section 11.2 we consider isomorphisms that are coherent with respect β-normalisation, giving them a suitable categorical characterisation could be an interesting step of future investigations. In order to do so, it seems natural to start from the work of Fiore and Saville on Cartesian closed bicategories in [Fiore and Saville, 2019, Fiore and Saville, 2020].

One could consider the λ-calculus $\Lambda_\perp(X)$ corresponding to the free Cartesian closed bicategory with pseudoreflexive object D on a set X, where each hom-category has an initial object that is preserved by composition and by the Cartesian closed structure in an appropriate sense. We conjecture that the isomorphisms we characterised syntactically in Section 11.2 correspond to the ones in the free cocompletion under filtered colimits of $\Lambda_\perp(X)(D^n, D)$. In this way one could define, in full generality, the free non-extensional theory of a model as the one that arises from those appropriate structural isomorphisms.

For the extensional case one could proceed analogously by taking an extensional D. In particular, this means that an extensional bicategorical model will determine both free non-extensional and extensional theories, and they will not coincide.

Besides these free constructions, one could also consider other relevant classes of isomorphisms between interpretations. Some questions immediately arise, which depend both on the choice of isomorphisms and on the particular model considered: can these isomorphisms be characterised via appropriate structural isomorphisms of some sort? What is the equational theory associated with those isomorphisms?

Second Dimension and Extensionality

One could study the possible extension of the method introduced in this paper to study the extensional theory of the models $D^+, D^{[n]}$ and D^*, individually introduced in Definition 9.2.5, and the relationship between these models and other constructions of extensional models introduced in [Fiore et al., 2008, Galal, 2020]. As an approximation theory one shall consider Lévy's extensional Böhm trees, as in [Manzonetto and Ruoppolo, 2014, Breuvart et al., 2018], or Nakajima trees as done for Scott's $\mathcal{D}_\infty$ model in [Hyland, 1976].

We conjecture that our technique can be adapted to prove that the extensional models such as D^+ and $D^{[n]}$ do satisfy an approximation theorem with respect Lévy's extensional approximants and that, as a corollary, one gets $\mathrm{Th}(\mathcal{D}) = \mathcal{H}^+$, where $\mathcal{H}^+$ is the λ-theory equating two λ-terms having the same Böhm tree up to countably many finite η-expansions.

This conjecture is motivated by analogous results available in the relational setting [Breuvart et al., 2018]. In Section 11 we presented a direct proof of $\mathrm{Th}(\mathcal{D}) = \mathcal{B}$, which constitutes the first characterisation of the λ-theory induced by a bicategorical model. In [Breuvart et al., 2018] a relational graph model $\mathcal{E}$ having theory $\mathcal{B}$ is presented, thus, by Corollary 12.2.8 all bicategorical models $\mathcal{D}$ having $\mathcal{E}$ as decategorification satisfy $\mathrm{Th}(\mathcal{D}) \subseteq \mathrm{Th}(\mathcal{E}) = \mathcal{B}$. Since $\mathcal{B} \subseteq \mathrm{Th}(\mathcal{D})$ is a corollary of the Approximation Theorem, this gives an indirect proof of $\mathrm{Th}(\mathcal{D}) = \mathcal{B}$ for these models.

In [Olimpieri, 2021] the construction of the bicategory of distributors is more parametrised and allows to obtain also Scott-continuous models by decategorification, and many theories of continuous models are known (see [Berline, 2000], for a survey). Since $\mathrm{Th}(\mathcal{D}) \subseteq \mathcal{T}$ is usually the difficult direction in proving $\mathrm{Th}(\mathcal{D}) = \mathcal{T}$, we believe that these results may be transposed from the categorical to the bicategorical setting using the decategorification and the above reasoning.

List of Figures

Index

14. Bibliography

[Accattoli and Dal Lago, 2012] Accattoli, B. and Dal Lago, U. (2012). On the invariance of the unitary cost model for head reduction (long version). *CoRR*, abs/1202.1641.

[Accattoli and Guerrieri, 2016] Accattoli, B. and Guerrieri, G. (2016). Open call-by-value. In Igarashi, A., editor, *Programming Languages and Systems - 14th Asian Symposium, APLAS 2016, Hanoi, Vietnam, November 21-23, 2016, Proceedings*, volume 10017 of *Lecture Notes in Computer Science*, pages 206–226.

[Accattoli and Guerrieri, 2017] Accattoli, B. and Guerrieri, G. (2017). Implementing open call-by-value (extended version). *CoRR*, abs/1701.08186.

[Accattoli and Guerrieri, 2018] Accattoli, B. and Guerrieri, G. (2018). Types of fireballs. In Ryu, S., editor, *Programming Languages and Systems - 16th Asian Symposium, APLAS 2018, Wellington, New Zealand, December 2-6, 2018, Proceedings*, volume 11275 of *Lecture Notes in Computer Science*, pages 45–66. Springer.

[Accattoli and Guerrieri, 2022] Accattoli, B. and Guerrieri, G. (2022). The theory of call-by-value solvability (long version).

[Accattoli and Kesner, 2012] Accattoli, B. and Kesner, D. (2012). The permutative lambda calculus. In *18th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning - LPAR-18*, Merida, Venezuela.

[Accattoli and Paolini, 2012] Accattoli, B. and Paolini, L. (2012). Call-by-value solvability, revisited. pages 4–16.

[Amadio and Curien, 1998] Amadio, R. and Curien, P.-L. (1998). *Domains and Lambda Calculi*. Cambridge tracts in Theoretical Computer Science. Cambridge University Press.

[Arrial et al., 2023] Arrial, V., Guerrieri, G., and Kesner, D. (2023). Quantitative inhabitation for different lambda calculi in a unifying framework. *Proc. ACM Program. Lang.*, 7(POPL).

[Backus et al., 1960] Backus, J. W., Bauer, F. L., Green, J., Katz, C., McCarthy, J., Perlis, A. J., Rutishauser, H., Samelson, K., Vauquois, B., Wegstein, J. H., Van Wijngaarden, A., and Woodger, M. (1960). Report on the algorithmic language ALGOL 60. *Communications of the ACM*, 3(5):299–314.

[Bakel, 2011] Bakel, S. V. (2011). Strict intersection types for the lambda calculus. *ACM Comput. Surv.*, 43(3):20:1–20:49.

[Barendregt et al., 1983] Barendregt, H., Coppo, M., and Dezani-Ciancaglini, M. (1983). A filter lambda model and the completeness of type assignment. *Journal of Symbolic Logic*, 48(4):931–940.

[Barendregt, 1971] Barendregt, H. P. (1971). *Some extensional term models for combinatory logics and λ-calculi.* Ph.D. thesis, Utrecht Universiteit, the Netherlands.

[Barendregt, 1974] Barendregt, H. P. (1974). Solvability in lambda-calculi. *Journal of Symbolic Logic - JSYML*, pages 372–372.

[Barendregt, 1977] Barendregt, H. P. (1977). The type free lambda calculus. In Barwise, J., editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, pages 1091–1132. North-Holland, Amsterdam.

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

[Berline, 2000] Berline, C. (2000). From computation to foundations via functions and application: The λ-calculus and its webbed models. *Theor. Comput. Sci.*, 249(1):81–161.

[Bernadet and Lengrand, 2013] Bernadet, A. and Lengrand, S. J. (2013). Non-idempotent intersection types and strong normalisation. *Logical Methods in Computer Science*, Volume 9, Issue 4.

[Böhm, 1968] Böhm, C. (1968). Alcune proprietà delle forme β-η-normali nel λ-K-calcolo. *INAC*, 696:1–19.

[Borceux, 1994] Borceux, F. (1994). *Handbook of Categorical Algebra*, volume 1 of *Encyclopedia of Mathematics and its Applications.* Cambridge University Press.

[Boudes et al., 2013] Boudes, P., He, F., and Pagani, M. (2013). A characterization of the Taylor expansion of lambda-terms. In Ronchi Della Rocca, S., editor, *Computer Science Logic 2013 (CSL 2013), CSL 2013, September 2-5, 2013, Torino, Italy*, volume 23 of *LIPIcs*, pages 101–115. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

[Breuvart et al., 2018] Breuvart, F., Manzonetto, G., and Ruoppolo, D. (2018). Relational graph models at work. *Log. Methods Comput. Sci.*, 14(3).

[Bucciarelli et al., 2007] Bucciarelli, A., Ehrhard, T., and Manzonetto, G. (2007). Not enough points is enough. In Duparc, J. and Henzinger, T. A., editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *Lecture Notes in Computer Science*, pages 298–312. Springer.

[Bucciarelli et al., 2014] Bucciarelli, A., Kesner, D., and Ronchi Della Rocca, S. (2014). The inhabitation problem for non-idempotent intersection types. In *IFIP TCS*, volume 8705 of *Lecture Notes in Computer Science*, pages 341–354. Springer.

[Bucciarelli et al., 2020] Bucciarelli, A., Kesner, D., Ríos, A., and Viso, A. (2020). The bang calculus revisited.

[Bucciarelli et al., 2017] Bucciarelli, A., Kesner, D., and Ventura, D. (2017). Non-idempotent intersection types for the lambda-calculus. *Log. J. IGPL*, 25(4):431–464.

[Bénabou, 1973] Bénabou, J. (1973). *Les distributeurs: d'après le cours de Questions spéciales de mathématique*. Rapport (Université catholique de Louvain (1970-). Séminaire de mathématique pure)). Institut de mathématique pure et appliquée, Université catholique de Louvain.

[Bénabou, 2000] Bénabou, J. (2000). Distributors at work. Technical report.

[Carraro and Guerrieri, 2014] Carraro, A. and Guerrieri, G. (2014). A semantical and operational account of call-by-value solvability. In Muscholl, A., editor, *Foundations of Software Science and Computation Structures - 17th International Conference, FOSSACS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, volume 8412 of *Lecture Notes in Computer Science*, pages 103–118. Springer.

[Cattani and Winskel, 2005] Cattani, G. L. and Winskel, G. (2005). Profunctors, open maps and bisimulation. *Mathematical Structures in Computer Science*, 15(3):553–614.

[Cheng et al., 2003] Cheng, E., Hyland, M., and Power, J. (2003). Pseudo-distributive laws. *Electronic Notes in Theoretical Computer Science*, 83:227–245. Proceedings of 19th Conference on the Mathematical Foundations of Programming Semantics.

[Church, 1932] Church, A. (1932). A set of postulates for the foundation of logic. *Annals of Mathematics*, 33(2):346–366.

[Church, 1941] Church, A. (1941). *The Calculi of Lambda Conversion. (AM-6)*. Princeton University Press.

[Church and Rosser, 1936] Church, A. and Rosser, J. B. (1936). Some properties of conversion. *Transactions of the American Mathematical Society*, 39(3):472–482.

[Coppo and Dezani-Ciancaglini, 1978] Coppo, M. and Dezani-Ciancaglini, M. (1978). A new type-assignment for lambda terms. In *Archiv für Mathematische Logik und Grundlagenforschung*, pages 139–156.

[Coppo and Dezani-Ciancaglini, 1980] Coppo, M. and Dezani-Ciancaglini, M. (1980). An extension of the basic functionality theory for the λ-calculus. *Notre Dame J. Formal Log.*, 21:685–693.

[Coppo et al., 1987] Coppo, M., Dezani-Ciancaglini, M., and Zacchi, M. (1987). Type theories, normal forms and $\mathcal{D}_\infty$-lambda-models. *Information and Computation*, 72(2):85–116.

[de Carvalho, 2007] de Carvalho, D. (2007). *Sémantiques de la logique linéaire.* Phd thesis, Aix-Marseille Université.

[de Carvalho, 2018] de Carvalho, D. (2018). Execution time of λ-terms via denotational semantics and intersection types. *Math. Struct. Comput. Sci.*, 28(7):1169–1203.

[Egidi et al., 1992] Egidi, L., Honsell, F., and Ronchi Della Rocca, S. (1992). Operational, denotational and logical descriptions: a case study. *Fundam. Informaticae*, 16(1):149–169.

[Ehrhard, 2002] Ehrhard, T. (2002). On köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science*, 12.

[Ehrhard, 2005] Ehrhard, T. (2005). Finiteness spaces. *Mathematical Structures in Computer Science*, 15(4).

[Ehrhard, 2012] Ehrhard, T. (2012). Collapsing non-idempotent intersection types. In Cégielski, P. and Durand, A., editors, *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the European Association for Computer Science Logic, CSL 2012, September 3-6, 2012, Fontainebleau, France*, volume 16 of *LIPIcs*, pages 259–273. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[Ehrhard, 2016] Ehrhard, T. (2016). Call-by-push-value from a linear logic point of view. In Thiemann, P., editor, *Programming Languages and Systems*, pages 202–228, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Ehrhard and Regnier, 2003] Ehrhard, T. and Regnier, L. (2003). The differential lambda-calculus. *Theor. Comput. Sci.*, 309(1-3):1–41.

[Ehrhard and Regnier, 2006a] Ehrhard, T. and Regnier, L. (2006a). Böhm trees, Krivine's machine and the Taylor expansion of lambda-terms. In Beckmann, A., Berger, U., Löwe, B., and Tucker, J. V., editors, *Logical Approaches to Computational Barriers, Second Conference on Computability in Europe, CiE 2006, Swansea, UK, June 30-July 5, 2006, Proceedings*, volume 3988 of *Lecture Notes in Computer Science*, pages 186–197. Springer.

[Ehrhard and Regnier, 2006b] Ehrhard, T. and Regnier, L. (2006b). Differential interaction nets. *Theoretical Computer Science*, 364(2).

[Ehrhard and Regnier, 2008] Ehrhard, T. and Regnier, L. (2008). Uniformity and the Taylor expansion of ordinary lambda-terms. *Theor. Comput. Sci.*, 403(2-3):347–372.

[Engeler, 1981] Engeler, E. (1981). Algebras and combinators. *Algebra Universalis*, 13(3):389–392.

[Fiore, 2005] Fiore, M. (2005). Mathematical models of computational and combinatorial structures. In Sassone, V., editor, *Foundations of Software Science and Computational Structures, 8th International Conference, FOSSACS 2005, Held as Part of the Joint*

European Conferences on Theory and Practice of Software, ETAPS 2005, Edinburgh, UK, April 4-8, 2005, Proceedings, volume 3441 of *Lecture Notes in Computer Science*, pages 25–46. Springer.

[Fiore et al., 2008] Fiore, M., Gambino, N., Hyland, M., and Winskel, G. (2008). The cartesian closed bicategory of generalised species of structures. *Journal of the London Mathematical Society*, 77(1):203—-220.

[Fiore et al., 2017] Fiore, M., Gambino, N., Hyland, M., and Winskel, G. (2017). Relative pseudomonads, kleisli bicategories, and substitution monoidal structures. *Selecta Mathematica*, 24(3):2791–2830.

[Fiore and Saville, 2019] Fiore, M. and Saville, P. (2019). A type theory for cartesian closed bicategories (extended abstract). In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–13.

[Fiore and Saville, 2020] Fiore, M. and Saville, P. (2020). Coherence and normalisation-by-evaluation for bicategorical cartesian closed structure. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '20, page 425–439, New York, NY, USA. Association for Computing Machinery.

[Galal, 2020] Galal, Z. (2020). A Profunctorial Scott Semantics. In Ariola, Z. M., editor, *5th International Conference on Formal Structures for Computation and Deduction (FSCD 2020)*, volume 167 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:18, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

[Gambino and Joyal, 2017] Gambino, N. and Joyal, A. (2017). On operads, bimodules and analytic functors. *Memoirs of the American Mathematical Society*, 249(1184).

[García-Pérez and Nogueira, 2016] García-Pérez, Á. and Nogueira, P. (2016). No solvable lambda-value term left behind. *Log. Methods Comput. Sci.*, 12(2).

[Gardner, 1994] Gardner, P. (1994). Discovering needed reductions using type theory. In *Theoretical Aspects of Computer Software*, pages 555–574, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Girard, 1987] Girard, J.-Y. (1987). Linear logic. *Theoretical Computer Science*, 50(1):1 – 101.

[Girard, 1988] Girard, J.-Y. (1988). Normal functors, power series and lambda-calculus. *Annals of Pure and Applied Logic*, 37(2):129.

[Girard, 1989] Girard, J.-Y. (1989). Geometry of interaction 1: Interpretation of system f. In Ferro, R., Bonotto, C., Valentini, S., and Zanardo, A., editors, *Logic Colloquium '88*, volume 127 of *Studies in Logic and the Foundations of Mathematics*, pages 221–260. Elsevier.

[Guerrieri and Olimpieri, 2021] Guerrieri, G. and Olimpieri, F. (2021). Categorifying Non-Idempotent Intersection Types. In Baier, C. and Goubault-Larrecq, J., editors, *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*, volume 183 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:24, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

[Guerrieri et al., 2017] Guerrieri, G., Paolini, L., and Ronchi Della Rocca, S. (2017). Standardization and conservativity of a refined call-by-value lambda-calculus. *Logical Methods in Computer Science*, 13(4).

[Herbelin and Zimmermann, 2009] Herbelin, H. and Zimmermann, S. (2009). An operational account of call-by-value minimal and classical λ-calculus in "natural deduction" form. In Curien, P.-L., editor, *Typed Lambda Calculi and Applications*, pages 142–156, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Hilken, 1996] Hilken, B. P. (1996). Towards a proof theory of rewriting: the simply typed 2λ-calculus. *Theor. Comput. Sci.*, 170(1-2):407–444.

[Hirschowitz, 2013] Hirschowitz, T. (2013). Cartesian closed 2-categories and permutation equivalence in higher-order rewriting. *Log. Methods Comput. Sci.*, 9(3).

[Honsell and Lenisa, 1993] Honsell, F. and Lenisa, M. (1993). Some results on the full abstraction problem for restricted lambda calculi. In Borzyszkowski, A. M. and Sokolowski, S., editors, *Mathematical Foundations of Computer Science 1993, 18th International Symposium, MFCS'93, Gdansk, Poland, August 30 - September 3, 1993, Proceedings*, volume 711 of *Lecture Notes in Computer Science*, pages 84–104. Springer.

[Hyland, 1975] Hyland, J. M. E. (1975). A survey of some useful partial order relations on terms of the lambda calculus. In *Lambda-Calculus and Computer Science Theory, Proceedings of the Symposium Held in Rome, March 25-27, 1975*, pages 83–95.

[Hyland, 1976] Hyland, J. M. E. (1976). A syntactic characterization of the equality in some models for the λ-calculus. *Journal London Mathematical Society (2)*, 12(3):361–370.

[Hyland, 2017] Hyland, J. M. E. (2017). Classical lambda calculus in modern dress. *Math. Struct. Comput. Sci.*, 27(5):762–781.

[Hyland et al., 2006] Hyland, J. M. E., Nagayama, M., Power, J., and Rosolini, G. (2006). A category theoretic formulation for Engeler-style models of the untyped lambda calculus. *Electron. Notes Theor. Comput. Sci.*, 161:43–57.

[Johnson and Yau, 2021] Johnson, N. and Yau, D. (2021). *2-Dimensional Categories*. Oxford University Press.

[Joyal, 1986] Joyal, A. (1986). Foncteurs analytiques et espèces de structures. In *Combinatoire énumérative*, pages 126–159, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Kelly, 1980] Kelly, G. M. (1980). A unified treatment of transfinite constructions for free algebras, free monoids, colimits, associated sheaves, and so on. *Bulletin of the Australian Mathematical Society*, 22(1):1–83.

[Kelly, 1982] Kelly, M. (1982). *Basic Concepts of Enriched Category Theory*, volume 64 of *Lecture Notes in Mathematics.* Cambridge University Press, Cambridge. Republished as: Reprints in Theory and Applications of Categories, No. 10 (2005) pp. 1–136.

[Kerinec et al., 2023] Kerinec, A., Manzonetto, G., and Olimpieri, F. (2023). Why are proofs relevant in proof-relevant models? *Proc. ACM Program. Lang.*, 7(POPL):218–248.

[Kerinec et al., 2020] Kerinec, A., Manzonetto, G., and Pagani, M. (2020). Revisiting call-by-value Böhm trees in light of their Taylor expansion. *Log. Methods Comput. Sci.*, 16(3).

[Kerinec et al., 2021] Kerinec, A., Manzonetto, G., and Ronchi Della Rocca, S. (2021). Call-by-value, again! In Kobayashi, N., editor, *6th International Conference on Formal Structures for Computation and Deduction, FSCD 2021, July 17-24, 2021, Buenos Aires, Argentina (Virtual Conference)*, volume 195 of *LIPIcs*, pages 7:1–7:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[Kesner and Peyrot, 2022] Kesner, D. and Peyrot, L. (2022). Solvability for Generalized Applications. In Felty, A. P., editor, *7th International Conference on Formal Structures for Computation and Deduction (FSCD 2022)*, volume 228 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:22, Dagstuhl, Germany. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[Kleene, 1936] Kleene, S. C. (1936). λ-definability and recursiveness. *Duke Mathematical Journal*, 2(2):340 – 353.

[Klop, 1975] Klop, J. W. (1975). On solvability by λI-terms. In Böhm, C., editor, *λ-Calculus and Computer Science Theory*, pages 342–345, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Krivine, 1993] Krivine, J.-L. (1993). Lambda-calculus, types and models. In *Ellis Horwood series in computers and their applications.*

[Lack, 2000] Lack, S. (2000). A coherent approach to pseudomonads. *Advances in Mathematics*, 152(2):179 – 202.

[Laird, 2017] Laird, J. (2017). From qualitative to quantitative semantics - by change of base. In Esparza, J. and Murawski, A. S., editors, *Foundations of Software Science and Computation Structures - 20th International Conference, FOSSACS 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*, volume 10203 of *Lecture Notes in Computer Science*, pages 36–52.

[Landin, 1964] Landin, P. J. (1964). The Mechanical Evaluation of Expressions. *The Computer Journal*, 6(4):308–320.

[Lassen, 1999] Lassen, S. B. (1999). Bisimulation in untyped lambda calculus: Böhm trees and bisimulation up to context. *Electr. Notes Theor. Comput. Sci.*, 20:346–374.

[Leinster, 1998] Leinster, T. (1998). Basic bicategories. arXiv.

[Loregian, 2021] Loregian, F. (2021). *(Co)end Calculus*. London Mathematical Society Lecture Note Series. Cambridge University Press, Cambridge.

[Manzonetto and Pagani, 2011] Manzonetto, G. and Pagani, M. (2011). Böhm's theorem for resource lambda calculus through Taylor expansion. In Ong, C. L., editor, *Typed Lambda Calculi and Applications - 10th International Conference, TLCA 2011, Novi Sad, Serbia, June 1-3, 2011. Proceedings*, volume 6690 of *Lecture Notes in Computer Science*, pages 153–168. Springer.

[Manzonetto et al., 2019a] Manzonetto, G., Pagani, M., and Ronchi Della Rocca, S. (2019a). New semantical insights into call-by-value λ-calculus. *Fundam. Informaticae*, 170(1-3):241–265.

[Manzonetto et al., 2019b] Manzonetto, G., Ronchi Della Rocca, S., and Pagani, M. (2019b). New semantical insights into call-by-value λ-calculus. *Fundam. Inform.*, 170(1-3):241–265.

[Manzonetto and Ruoppolo, 2014] Manzonetto, G. and Ruoppolo, D. (2014). Relational graph models, Taylor expansion and extensionality. In Jacobs, B., Silva, A., and Staton, S., editors, *Proceedings of the 30th Conference on the Mathematical Foundations of Programming Semantics, MFPS 2014, Ithaca, NY, USA, June 12-15, 2014*, volume 308 of *Electronic Notes in Theoretical Computer Science*, pages 245–272. Elsevier.

[Mazza et al., 2017] Mazza, D., Pellissier, L. P., and Vial, P. (2017). Polyadic approximations, fibrations and intersection types. *Proceedings of the ACM on Programming Languages*, 2:1 – 28.

[Melliès and Zeilberger, 2015] Melliès, P.-A. and Zeilberger, N. (2015). Functors are type refinement systems. *SIGPLAN Not.*, 50(1):3–16.

[Mogensen, 1992] Mogensen, T. Æ. (1992). Efficient self-interpretations in lambda calculus. *J. Funct. Program.*, 2(3):345–363.

[Olimpieri, 2020] Olimpieri, F. (2020). *Intersection Types and Resource Calculi in the Denotational Semantics of Lambda-Calculus*. PhD thesis, Aix-Marseille Université.

[Olimpieri, 2021] Olimpieri, F. (2021). Intersection type distributors. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–15. IEEE.

[Ong, 1997] Ong, L. (1997). Lambda calculus. Lecture Notes.

[Paolini, 2001] Paolini, L. (2001). Call-by-value separability and computability. In Restivo, A., Ronchi Della Rocca, S., and Roversi, L., editors, *Theoretical Computer Science, 7th Italian Conference, ICTCS 2001*, volume 2202 of *Lecture Notes in Computer Science*, pages 74–89. Springer.

[Paolini, 2008] Paolini, L. (2008). Parametric λ-theories. *Theoretical Computer Science*, 398(1):51 – 62.

[Paolini et al., 2017] Paolini, L., Piccolo, M., and Ronchi Della Rocca, S. (2017). Essential and relational models. *Math. Struct. Comput. Sci.*, 27(5):626–650.

[Paolini et al., 2005] Paolini, L., Pimentel, E., and Ronchi Della Rocca, S. (2005). Lazy strong normalization. *Electronic Notes in Theoretical Computer Science*, 136:103–116. Proceedings of the Third International Workshop on Intersection Types and Related Systems (ITRS 2004).

[Paolini and Ronchi Della Rocca, 1999] Paolini, L. and Ronchi Della Rocca, S. (1999). Call-by-value solvability. *ITA*, 33(6):507–534.

[Plotkin, 1975] Plotkin, G. D. (1975). Call-by-name, call-by-value and the lambda-calculus. *Theor. Comput. Sci.*, 1(2):125–159.

[Plotkin, 1993] Plotkin, G. D. (1993). Set-theoretical and other elementary models of the lambda-calculus. *Theoretical Computer Science*, 121(1):351–409.

[Pravato et al., 1999] Pravato, A., Ronchi Della Rocca, S., and Roversi, L. (1999). The call by value λ-calculus: a semantic investigation. *Mathematical Structures in Computer Science*, 9(5):617–650.

[Regnier, 1994] Regnier, L. (1994). Une équivalence sur les lambda-termes. *Theoretical Computer Science*, 126:281–292.

[Ronchi Della Rocca and Paolini, 2004] Ronchi Della Rocca, S. and Paolini, L. (2004). *The Parametric Lambda Calculus: A Metamodel for Computation.* Texts in Theoretical Computer Science. An EATCS Series. Springer Berlin Heidelberg.

[Saville, 2020] Saville, P. (2020). *Cartesian closed bicategories: type theory and coherence.* PhD thesis, University of Cambridge.

[Scott, 1970] Scott, D. S. (1970). Outline of a Mathematical Theory of Computation. Technical Report PRG–2, Oxford, England.

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

[Scott and Strachey, 1971] Scott, D. S. and Strachey, C. (1971). Toward a mathematical semantics for computer languages. *Oxford Programming Research Group Technical Monograph.*

[Seely, 1987] Seely, R. A. G. (1987). Modelling computations: A 2-categorical framework. In *Proceedings of the Symposium on Logic in Computer Science (LICS '87), Ithaca, New York, USA, June 22-25, 1987*, pages 65–71. IEEE Computer Society.

[Seely, 1989] Seely, R. A. G. (1989). Linear logic, -autonomous categories and cofree coalgebras.

[Tait, 1966] Tait, W. W. (1966). A nonconstructive proof of Gentzen's Hauptsatz for second order predicate logic. *Bulletin of the American Mathematical Society*, 72:980–983.

[Tranquilli, 2009] Tranquilli, P. (2009). *Nets Between Determinism and Nondeterminism.* PhD thesis, Univ. Paris 7 and Univ. Roma 3.

[Tsukada et al., 2017] Tsukada, T., Asada, K., and Ong, C.-H. L. (2017). Generalised species of rigid resource terms. In *Proceedings of the 32rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS 2017.

[Tsukada et al., 2018] Tsukada, T., Asada, K., and Ong, C.-H. L. (2018). Species, profunctors and Taylor expansion weighted by smcc: A unified framework for modelling nondeterministic, probabilistic and quantum programs. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 889–898.

[Turing, 1937] Turing, A. M. (1937). Computability and lambda-definability. *The Journal of Symbolic Logic*, 2(4):153–163.

[Wadsworth, 1971] Wadsworth, C. (1971). *Semantics and Pragmatics of the Lambda-calculus.* University of Oxford.

[Wadsworth, 1976] Wadsworth, C. P. (1976). 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.